Bridging the Training-Inference Gap for Dense Phrase Retrieval

Gyuwan Kim[1] **Jinhyuk Lee**[2] **Barlas Oğuz**[3] **Wenhan Xiong**[3]
Yizhe Zhang[4] **Yashar Mehdad**[3] **William Yang Wang**[1]

Abstract

Building dense retrievers requires a series of standard procedures, including training and validating neural models and creating indexes for efficient search. However, these procedures are often misaligned in that training objectives do not exactly reflect the retrieval scenario at inference time. In this paper, we explore how the gap between training and inference in dense retrieval can be reduced, focusing on dense phrase retrieval (Lee et al., 2021a) where billions of representations are indexed at inference. Since validating every dense retriever with a large-scale index is practically infeasible, we propose an efficient way of validating dense retrievers using a small subset of the entire corpus. This allows us to validate various training strategies including unifying contrastive loss terms and using hard negatives for phrase retrieval, which largely reduces the training-inference discrepancy. As a result, we improve phrase retrieval by 2-3% in top-1 accuracy and passage retrieval by 2-4% in top-20 accuracy for open-domain question answering. Our work urges modeling dense retrievers with careful consideration of training and inference via efficient validation while advancing phrase retrieval as a general solution for dense retrieval.

1. Introduction

Dense retrieval aims to learn effective representations of queries and documents by making representations of relevant query-document pairs to be similar (Chopra et al., 2005; Van den Oord et al., 2018). With the success of dense passage retrieval for open-domain question answering (QA) (Lee et al., 2019; Karpukhin et al., 2020), recent studies build an index for a finer granularity such as dense *phrase* retrieval (Lee et al., 2021a), which largely improves the computational efficiency of open-domain QA by replacing the retriever-reader model (Chen et al., 2017) with a retriever-only model (Seo et al., 2019; Lewis et al., 2021).

Building a dense retrieval system involves multiple steps (Figure 1) including training a dual encoder (§3), selecting the best model with validation (§2), and constructing an index (often with filtering (§A)) for an efficient search. However, these components are somewhat loosely connected to each other. For example, model training is not directly optimizing the retrieval performance on the full corpus on which models should be evaluated. In this paper, we aim to minimize the gap between training and inference of dense retrievers to achieve better retrieval performance.

However, developing a better dense retriever requires validation, which requires building large indexes from a full corpus (e.g., the entire Wikipedia for open-domain QA) for inference with a huge amount of computational resources and time. To tackle this problem, we first propose an efficient way of validating dense retrievers without building large-scale indexes. Analysis of using a smaller random corpus with different sizes for the validation reveals that the accuracy from small indexes does not necessarily correlate well with the retrieval accuracy on the full index. As an alternative, we construct a compact corpus using a pre-trained dense retriever so that validation on this corpus better correlates well with the retrieval on the full scale while keeping the size of the corpus as small as possible to perform efficient validation.

With our efficient validation, we revisit the training method of dense phrase retrieval (Lee et al., 2021a;b), a general framework for retrieving different granularities of texts such as phrases, passages, and documents. We reduce the training-inference discrepancy by unifying previous loss terms to discriminate a gold answer phrase from other negative phrases altogether instead of applying in-passage negatives (Lee et al., 2021b) and in-batch negatives separately. To better approximate the retrieval at inference where the number of negatives is extremely large, we use all available negative phrases from training passages and put more weights on negative phrases. We also leverage model-based hard negatives (Xiong et al., 2020) for phrase

[1]University of California, Santa Barbara [2]Korea University [3]Meta AI [4]Apple MLR. Correspondence to: William Yang Wang <william@cs.ucsb.edu>.

ICML workshop on Knowledge Retrieval and Language Models, Baltimore, Maryland, USA, 2022. Copyright 2022 by the author(s).

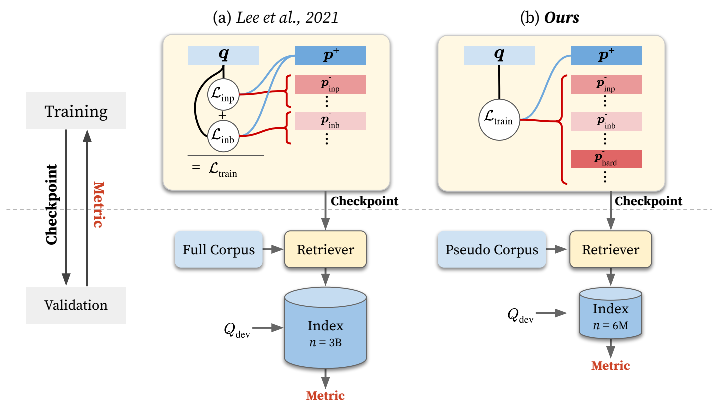


Figure 1: Comparison of the (a) original (Lee et al., 2021a) and (b) proposed procedure for DensePhrases training (top) and validation (bottom). We unify training loss terms $\mathcal{L}_{\text{inp}}$ and $\mathcal{L}_{\text{inb}}$ that enforce the representation of a question (q) similar to the representation of a positive phrase (p^+) while contrasting from representations of in-passage negative phrases (p^-_{inp}) and in-batch negative phrases (p^-_{inb}) respectively into a single term $\mathcal{L}_{\text{train}}$ and expand negatives in number and difficulty with hard negatives (p^-_{hard}). Also, we use a retrieval accuracy on the development set Q_{dev} using a smaller corpus instead of the full corpus as an efficient validation metric for selecting the best checkpoint.

retrieval, which hasn't been explored in previous studies. This enables our dense retrievers correct mistakes made at inference time.

In summary, we introduce an efficient way of validation for dense retrievers to confirm and accelerate better modeling of dense retrievers and improve dense phrase retrieval models with modified training objectives and hard negatives based on our efficient validation. Consequently, we achieve 44.0% exact match score and 83.8% top-20 passage retrieval accuracy on Natural Questions (Kwiatkowski et al., 2019) and 55.5% exact match score and 82.7% top-20 passage retrieval accuracy on TriviaQA (Joshi et al., 2017), the state-of-the-art retrieval-only open-domain QA accuracy.

2. Efficient Validation of Phrase Retrieval

Our goal is to train a dense retriever $\mathcal{M}$ that can accurately find a correct answer in the entire corpus C (in our case, Wikipedia). Careful validation is necessary to confirm whether new training methods are truly effective. It also helps finding optimal configurations induced by those techniques. However, building a large-scale index for every model makes the model development process slow and also requires huge memory. Thus, an efficient validation method could expedite modeling innovations in the correct directions. It could also allow frequent comparison of different checkpoints when updating a full index simultaneously during the training is computationally infeasible.[1]

Measuring the retrieval accuracy on an index from a smaller subset of the full corpus (denoted as $C^\star$) for model validation would be a practical choice, hoping $\text{argmax}_{\mathcal{M}\in\Omega}\,\text{acc}(D|\mathcal{M}, C^\star) \approx \text{argmax}_{\mathcal{M}\in\Omega}\,\text{acc}(D|\mathcal{M}, C)$ where Ω is a set of model candidates and acc means the retrieval accuracy on a QA dataset D. We first examine how a relative order of accuracy between modeling approaches may change with varying sizes of the random subcorpus (§2.1) and then develop a clever way to construct a compact subcorpus that maintains reasonable correlation with the retrieval accuracy in the full scale (§2.2).

[1]Although some works (Guu et al., 2020; Xiong et al., 2020) do asynchronous updates per specific number of training steps and use the intermediate index for better modeling, it requires a huge amount of computational resource.

2.1. Random Subcorpus

Reading comprehension (RC) can be regarded a special case of open-domain QA, where a corpus contains only a single gold passage (i.e., $C_q = \{c\}$) for each question. Here, the subcorpus is question-dependent. We first gather all gold passages from the development set as a small corpus C_0, a minimal set that contains answers to all development set questions. We consider a corpus R_r whose size is r times the size of the full corpus by simply appending C_0 with random passages by sampling from the full corpus C, i.e., $C_0 \subset R_r \subset C$ and $|R_r| = r|C|$. We first gather all gold passages from the development set as a small corpus C_0, a minimal set that contains answers to all development set questions. We consider a corpus C_r whose size is r times the size of the full corpus by simply appending C_0 with random passages by sampling from the full corpus C, i.e., $C_0 \subset C_r \subset C$ and $|C_r| = r|C|$. We specifically use $r = 1/100, 1/10$ in our experiments. As the corpus size increases, finding the correct answer from a larger number of possible candidates becomes more difficult, so the retrieval accuracy generally decreases (Reimers & Gurevych, 2021).

$$-\log \frac{e^{s(q,p^+;c)}}{e^{s(q,p^+;c)} + \sum_{(p^-;c) \in N_{\text{inp}}} e^{s(q,p^-;c)}} - \lambda \log \frac{e^{s(q,p^+;c)}}{e^{s(q,p^+;c)} + \sum_{(p^-;c') \in N_{\text{inb}} \cup N_{\text{prb}}} e^{s(q,p^-;c')}} \tag{1}$$

$$-\log \frac{e^{s(q,p^+;c)}}{e^{s(q,p^+;c)} + \sum_{(p^-;c') \in N_{\text{inp}} \cup N_{\text{inb}} \cup N_{\text{prb}} \cup N_{\text{hard}}} \lambda(p^-) e^{s(q,p^-;c')}} \tag{2}$$

DensePhrases (Lee et al., 2021a) simply choose the best checkpoint with the highest RC accuracy assuming that a model with better RC accuracy leads to a better retrieval accuracy, or use the last checkpoint at the end of the training. It is problematic since our preliminary experiments demonstrate that the RC accuracy and the retrieval accuracy on different sizes of corpus including the full corpus, do not necessarily correlate well with each other. Using a large subcorpus is better for accurate validation not to deviate much from the trends of retrieval accuracy of a full corpus. However, a smaller subcorpus would be better in terms of validation efficiency. This trade-off drives us to design a better way of constructing a validation corpus.

2.2. Hard Subcorpus

The retrieval accuracy given a subcorpus $C^\star$ should have a high correlation with the retrieval accuracy over the full corpus and the size of corpus $|C^\star|$ should be small enough (or as small as possible) for efficient validation. For a reasonably accurate dense retriever, it is relatively easy to discriminate a gold phrase from other phrases in random passages. Therefore, it is better to collect a subcorpus with *hard* passages to test dense retrievers on a similar condition to a full corpus which includes many difficult phrases to discriminate if the corpus can have a limited number of negative passages.

We construct a hard corpus H_k with a compact size using a pre-trained retriever $\bar{\mathcal{M}}$ to extract all context passages of top-k retrieved phrases for all query q in the development set Q_{dev}, and C_0 is merged to always include an answer, i.e., $H_k = C_0 \cup \bigcup_{(q,a) \in Q_{\text{dev}}} \bar{\mathcal{M}}_k(q|C)$ where $\mathcal{M}_k(q|C)$ denotes the top-k passage retrieval results for a query q from the model $\mathcal{M}$. If $\bar{\mathcal{M}}$ is reasonably accurate, negative examples retrieved by $\bar{\mathcal{M}}$ will make our new model $\mathcal{M}$ difficult to find a correct answer. We expect the retrieval accuracy from H_k quickly drops as k increases and reaches close to the retrieval accuracy on the full corpus C with a manageable k so that we can use retrieval accuracy on a hard subcorpus for efficient validation.

3. Optimized Training of DensePhrases

3.1. Background: Training of DensePhrases

The question encoder and the phrase encoder are jointly trained using reading comprehension datasets. A phrase p is represented as a concatenation of start and end token vectors from the contextualized representations of a context passage c using a phrase encoder. A question q is represented as a concatenation of vectors using two different encoders for the start and the end.

The main training objective is a sum of the two separate contrastive loss terms weighted by the λ coefficient as formally defined in Equation 1.[2] One is for contrasting a phrase token of positive start/end position (p^+) to that of other positions in the context passage ($N_{\text{inp}} = \{(p;c) \neq (p^+;c) | p \in c\}$). Another is for contrasting the same token to other positive tokens in a current ($N_{\text{inb}} = \{(p';c') \neq (p^+;c) | (q',p';c') \in \mathcal{B}\}$) or previous T mini-batches ($N_{\text{prb}} = \{(p';c') | (q',p';c') \in \mathcal{B}_{\text{pre}}\}$). The numbers of negatives are $|N_{\text{inp}}| = L - 1$, $|N_{\text{inb}}| = B - 1$, and $|N_{\text{prb}}| = B \times T$ where L is the sequence length of context passages and B (= $|\mathcal{B}|$) is the batch size. Additional training techniques for DensePhrases are described in Appendix B.

3.2. Unified Loss

The original training objective of DensePhrases (Equation 1) has separate terms for finding a relevant passage (in/pre-batch negatives) and finding the exact phrase position in the passage (in-passage negatives). However, we should find an answer phrase among all possible candidates at once during the test time. Therefore, we modify the loss term as a unified version (Equation 2) by putting all negatives together into the contrastive targets.

We also introduce the λ coefficient to the unified loss to give weights to negatives. It opens a new question of how we should set the value of λ. The role of λ can be interpreted in two ways. First, multiplying λ to an exponential of a score is equivalent to adding a positive value to the score ($\lambda e^s = e^{s+\log\lambda}$), and then the loss term becomes the soft version of margin-based loss. Second, using λ can mimic the inference time where the number of negative tokens is much larger by duplicating a negative λ times ($\lambda e^s = e^s + e^s + ... + e^s$) to close the gap between training and test. Based on the second interpretation, we set different value of λ depending on where negative phrase p^- is from: $\lambda(p^-) = \lambda_{\text{inp}}\delta(p^- \in N_{\text{inp}}) + \lambda_{\text{inb}}\delta(p^- \in N_{\text{inb}} \cup N_{\text{prb}}) + \lambda_{\text{hard}}\delta(p^- \in N_{\text{hard}})$.

We extend to use all tokens in context passages with a similar

[2]We denote the similarity score between a question q and a phrase p as $s(q,p;c)$. While the score and the loss term of start and end tokens are separately calculated in practice, we abbreviate it in the equation for simplicity.

intuition that contrasting with as many tokens as possible could be helpful instead of using only start/end position tokens to in/pre-batch negatives. It changes in/pre-batch negatives to $N_{\text{inb}} = \{(p; c') \neq (p^{+}; c) | p \in c', (q', p'; c') \in \mathcal{B}\}$ and $N_{\text{prb}} = \{(p; c') | p \in c', (q', p'; c') \in \mathcal{B}_{\text{pre}}\}$) and their sizes $|N_{\text{inb}}| = B \times L - 1$ and $|N_{\text{prb}}| = B \times T \times L$. This change also increases the number of negatives hundreds of times and turns out empirically advantageous.

3.3. Hard Negatives for Phrase Retrieval

We exploit hard negatives to benefit phrase retrieval, a widely used technique for passage retrieval[3] but never fully examined for phrase retrieval. We use a encoder model and phrase index from the first round to extract model-based hard negatives from top-k phrase retrieval results for questions in the training set. We exclude examples when a context passage of a retrieved phrase contains an answer.[4] A context passage corresponding to a retrieved phrase can be restored using information stored along with the index. It helps to focus more on topically different documents and shares the intuition from the analysis in Lee et al. (2021b) that DensePhrases less rely on the topical relevance than DPR. Using high-quality hard negatives by removing false negatives is important to train a better model. We left filtering based on a cross encoder (Qu et al., 2021; Ren et al., 2021) to future work due to the convenience of automatic filtering.

After the hard negative mining, we fine-tune a dual encoder with the hard negatives. We sample h hard negatives for each training step and append them to negative targets for the loss calculation. We expect that hard negatives give a better training signal than random in/pre-batch negatives (Xiong et al., 2020) because those are examples difficult to discriminate for the previous model. Moreover, hard negatives extracted from the larger corpus could expose a model to other diverse documents than the original training dataset. This is similar to query-side fine-tuning but differs in that both the question encoder and phrase encoder are updated.

Similar to §3.2, we include all tokens in the context passage of all hard negatives in a mini-batch for N_{hard}. Using all available tokens is generally better because they potentially belong to the final negative phrase candidates in inference. Training with larger numbers of negatives is beneficial to reduce the gap between training and inference. Including all of them does not induce significant additional memory overhead since we should encode the same number (h) of passages regardless of different options. Therefore, we use $N_{\text{hard}} = \bigcup_{(\hat{p}; \hat{c}) \in H(q, p^{+}; c), (q, p^{+}; c) \in \mathcal{B}} \{(p; \hat{c}) | p \in \hat{c}\}$ as all tokens from all hard negatives in a mini-batch where H is a set of the sampled h hard negatives and $|N_{\text{hard}}| = B \times L \times h$.

[3] Karpukhin et al. (2020) use one hard negative obtained from BM25 per example in addition to in-batch negatives for training a dual encoder. Xiong et al. (2020) globally select hard negatives from the entire corpus with asynchronously index updates for faster convergence. RocketQA (Qu et al., 2021) denoises hard negatives using cross encoder.

[4] Compared to more strict condition based on exact match, it reduces about 20% of negative pairs, hopefully reducing false negatives and achieving higher accuracy gain.

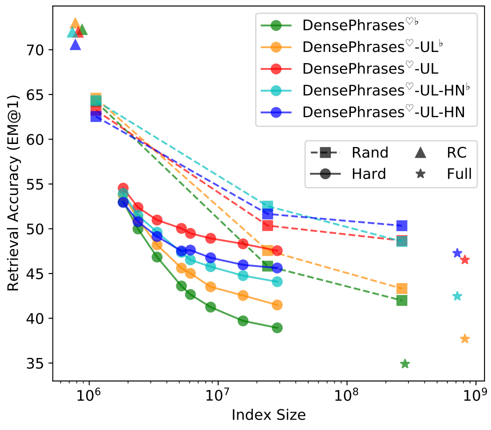


Figure 2: Validation results on open-domain QA. We plot retrieval accuracy (EM@1) on indexes with different sizes (log-scale) from random and hard subcorpora. Random subcorpora () include C_0, $R_{1/100}$, and $R_{1/10}$. Hard subcorpora (●) include H_k for $k \in \{1, 2, 4, 8, 10, 16, 32, 64\}$. We also plot reading comprehension (RC) accuracy and retrieval accuracy on the full index with filtering. We compare five representative models with and without our proposed training methods and query-side fine-tuning. All models are trained and evaluated on Natural Questions. UL, HN, and $^{\flat}$ indicate a model trained with the unified loss, a model trained with hard negatives, and a model before query-side fine-tuning.

3.4. Token Filter

We modify to train a token filter which is used to reduce the index size separately rather than joint training with the dual encoder. Appendix A contains a more detailed explanation and interesting analysis of the filter threshold.

4. Experiments

To show the effectiveness of our proposed method, we evaluate DensePhrases models on open-domain QA benchmarks following the experimental setup of Lee et al. (2021a;b).[5]

Datasets We measure phrase retrieval accuracy and passage retrieval accuracy on two open-domain QA datasets: Natural Questions (Kwiatkowski et al., 2019) and TriviaQA (Joshi et al., 2017). We train our phrase retrieval models with

[5] https://github.com/princeton-nlp/DensePhrases

Table 1: Open-domain QA phrase retrieval and passage retrieval results on the test set. We report the exact match score (EM@1) for phrase retrieval. We report top-k passage retrieval accuracy (Top-k, for $k \in \{1, 5, 20\}$), mean reciprocal rank at 20 (MRR@20), and precision at 20 (P@20) for passsage retrieval. $^\diamondsuit$: trained on each dataset independently. $^\spadesuit$: trained on multiple datasets. $^\heartsuit$: trained on Natural Questions datasets.

Model	**Phrase Retrieval**		**Passage Retrieval**									
	NaturalQ	TriviaQA	NaturalQ					TriviaQA				
	EM@1	EM@1	Top-1	Top-5	Top-20	MRR@20	P@20	Top-1	Top-5	Top-20	MRR@20	P@20
DPR$^\diamondsuit$ (+ BERT reader)	41.5	56.8	46.0	68.1	79.8	55.7	16.5	54.4	-	79.4	-	-
DPR$^\spadesuit$ (+ BERT reader)	41.5	56.8	44.2	66.8	79.2	54.2	17.7	54.6	70.8	79.5	61.7	30.3
RePAQ$^\diamondsuit$ (retrieval-only)	41.2	38.8	-	-	-	-	-	-	-	-	-	-
RePAQ$^\spadesuit$ (retrieval-only)	41.7	41.3	-	-	-	-	-	-	-	-	-	-
DensePhrases$^\heartsuit$	40.9	50.7	50.1	69.5	79.8	58.7	20.5	-	-	-	-	-
DensePhrases$^\spadesuit$	41.3	53.5	51.1	69.9	78.7	59.3	22.7	62.7	75.0	80.9	68.2	38.4
DensePhrases$^\heartsuit$-UL	43.5	51.3	57.1	75.7	83.7	65.2	22.0	62.0	74.6	80.6	67.6	33.3
DensePhrases$^\heartsuit$-UL-HN	**44.0**	47.0	**58.6**	75.7	83.4	**66.1**	21.9	60.3	73.3	79.6	66.1	32.3
DensePhrases$^\spadesuit$-UL	42.4	**55.5**	56.7	**75.9**	**83.8**	65.2	**23.7**	**65.0**	**76.6**	**82.7**	**70.2**	**39.0**

two different datasets followed by query-side fine-tuning on a target dataset: DensePhrases$^\heartsuit$ with Natural Questions reading comprehension dataset and DensePhrases$^\spadesuit$ with a combination of multiple reading comprehension datasets.[6] Training details are described in Appendix C. We build the phrase index using the 2018-12-20 Wikipedia snapshot (C) and its different smaller subsets $C^\star$.

Model Validation In our preliminary experiments, we observe that the best checkpoint among training epochs differs depending on the corpus size (especially for small scale). Figure 2 shows validation retrieval accuracy of the DensePhrases models with different training methods on various sizes of random and hard subcorpora. The retrieval accuracy on the hard subcorpus rapidly drops and reaches close to the retrieval accuracy on the full corpus as k increases with moderately increasing the index size. On the other hand, retrieval accuracy on a random subcorpus is higher than on a hard subcorpus with similar index size. For instance, retrieval accuracies on H_8 (5.1M) are lower than those on $R_{1/100}$ (24.2M) with 4 times smaller index, and retrieval accuracies on H_{16} (8.7M) are lower than those on $R_{1/10}$ (266.4M) with 30 times smaller index. It indicates that a hard subcorpus can effectively imitate inference with a full corpus where correct retrieval is the most difficult.

The relative order of accuracy between models on hard subcorpus converges quickly at around H_{10} (6.1M). Contrarily, the order changes from $R_{1/100}$ to $R_{1/10}$, meaning that it is difficult to be certain about the superiority of a model over another one based on a random subcorpus with an even larger index. Therefore, we can use retrieval accuracy on a hard subcorpus as an efficient validation metric.

[6] Natural Questions, WebQuestions (Berant et al., 2013), CuratedTREC (Baudiš & Šedivỳ, 2015), TriviaQA, and SQuAD (Rajpurkar et al., 2016)

Validation results on any subcorpus clearly demonstrate that unified loss is helpful. Query-side fine-tuning decreases RC accuracy and retrieval accuracy in C_0 (1.1M) while improving retrieval accuracy in larger indexes. It proves that a hasty conclusion with a small corpus size may avoid modeling development.

Open-domain QA results Table 1 summarizes experimental results on open-domain QA phrase retrieval and passage retrieval. Both unified loss and hard negatives (Appendix D) are shown to be effective. With the better training methods, our best model improves exact match accuracy on phrase retrieval from the original DensePhrases model by 2.7% in Natural Questions and 2.0% in TriviaQA, achieving the state-of-the-art retrieval-only performance. Our methods also improve passage retrieval accuracy significantly. Our best model improves top-20 accuracy on passage retrieval from the original DensePhrases model by 4.0% in Natural Questions and 1.8% in TriviaQA. We may use DensePhrases as a building block of other tasks and expect to achieve good phrase retrieval performance with expressive reader models like FiD (Izacard & Grave, 2021).

5. Conclusion

In this study, we aim to bridge the gap between training and inference of phrase retrieval. We first develop an efficient validation metric that measures retrieval accuracy on the index from a small corpus with hard passages using a pre-trained retriever. Based on this validation, we show that the improvements in training of dense phrase retrieval with unified loss and hard negatives are effective. As a result, we achieve state-of-the-art phrase retrieval and passage retrieval accuracy in open-domain question answering among retrieval-only approaches. We describe related work in Appendix E and potential implications in Appendix F.

References

Bartolo, M., Thrush, T., Jia, R., Riedel, S., Stenetorp, P., and Kiela, D. Improving Question Answering Model Robustness with Synthetic Adversarial Data Generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 8830–8848, 2021.

Baudiš, P. and Šedivỳ, J. Modeling of the Question Answering Task in the YodaQA System. In *Proceedings of the 6th International Conference on Experimental IR Meets Multilinguality, Multimodality, and Interaction*, pp. 222–228. Springer, 2015.

Berant, J., Chou, A., Frostig, R., and Liang, P. Semantic Parsing on Freebase from Question-Answer Pairs. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pp. 1533–1544, 2013.

Chen, D., Fisch, A., Weston, J., and Bordes, A. Reading Wikipedia to Answer Open-Domain Questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1870–1879, 2017.

Chopra, S., Hadsell, R., and LeCun, Y. Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pp. 539–546. IEEE, 2005.

Ge, T., He, K., Ke, Q., and Sun, J. Optimized Product Quantization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(4):744–755, 2013.

Guo, R., Sun, P., Lindgren, E., Geng, Q., Simcha, D., Chern, F., and Kumar, S. Accelerating Large-Scale Inference with Anisotropic Vector Quantization. In *International Conference on Machine Learning*, pp. 3887–3896. PMLR, 2020.

Guu, K., Lee, K., Tung, Z., Pasupat, P., and Chang, M.-W. REALM: Retrieval-Augmented Language Model Pre-Training. In *International Conference on Machine Learning*, 2020.

Hinton, G., Vinyals, O., Dean, J., et al. Distilling the Knowledge in a Neural Network. *arXiv preprint arXiv:1503.02531*, 2015.

Hofstätter, S., Lin, S.-C., Yang, J.-H., Lin, J., and Hanbury, A. Efficiently Teaching an Effective Dense Retriever with Balanced Topic Aware Sampling. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 113–122, 2021.

Izacard, G. and Grave, É. 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*, pp. 874–880, 2021.

Johnson, J., Douze, M., and Jégou, H. Billion-Scale Similarity Search with GPUs. *IEEE Transactions on Big Data*, 7 (3):535–547, 2019.

Joshi, M., Choi, E., Weld, D. S., and Zettlemoyer, L. 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)*, pp. 1601–1611, 2017.

Karpukhin, V., Oguz, B., Min, S., Lewis, P., Wu, L., Edunov, S., Chen, D., and Yih, W.-t. Dense Passage Retrieval for Open-Domain Question Answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 6769–6781, 2020.

Kaushik, D., Kiela, D., Lipton, Z. C., and Yih, W.-t. On the Efficacy of Adversarial Data Collection for Question Answering: Results from a Large-Scale Randomized Study. 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)*, pp. 6618–6633, 2021.

Khattab, O. and Zaharia, M. ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pp. 39–48, 2020.

Kiela, D., Bartolo, M., Nie, Y., Kaushik, D., Geiger, A., Wu, Z., Vidgen, B., Prasad, G., Singh, A., Ringshia, P., et al. Dynabench: Rethinking Benchmarking in NLP. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 4110–4124, 2021.

Kwiatkowski, T., Palomaki, J., Redfield, O., Collins, M., Parikh, A., Alberti, C., Epstein, D., Polosukhin, I., Devlin, J., Lee, K., et al. Natural Questions: A Benchmark for Question Answering Research. *Transactions of the Association for Computational Linguistics*, 7:452–466, 2019.

Lee, J., Seo, M., Hajishirzi, H., and Kang, J. Contextualized Sparse Representations for Real-Time Open-Domain Question Answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 912–919, 2020.

Lee, J., Sung, M., Kang, J., and Chen, D. Learning Dense Representations of Phrases at Scale. 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)*, pp. 6634–6647, 2021a.

Lee, J., Wettig, A., and Chen, D. Phrase Retrieval Learns Passage Retrieval, Too. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 3661–3672, 2021b.

Lee, K., Chang, M.-W., and Toutanova, K. Latent Retrieval for Weakly Supervised Open Domain Question Answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 6086–6096, 2019.

Lewis, P., Wu, Y., Liu, L., Minervini, P., Küttler, H., Piktus, A., Stenetorp, P., and Riedel, S. PAQ: 65 Million Probably-Asked Questions and What You Can Do With Them. *Transactions of the Association for Computational Linguistics*, 9:1098–1115, 2021.

Lin, J. A Proposed Conceptual Framework for a Representational Approach to Information Retrieval. In *ACM SIGIR Forum*, volume 55-2, pp. 1–29. ACM New York, NY, USA, 2022.

Melis, G., Dyer, C., and Blunsom, P. On the State of the Art of Evaluation in Neural Language Models. In *International Conference on Learning Representations*, 2018.

Min, S., Boyd-Graber, J., Alberti, C., Chen, D., Choi, E., Collins, M., Guu, K., Hajishirzi, H., Lee, K., Palomaki, J., et al. NeurIPS 2020 EfficientQA competition: Systems, analyses and lessons learned. In *NeurIPS 2020 Competition and Demonstration Track*, pp. 86–111. PMLR, 2021.

Mitra, B. and Craswell, N. *An Introduction to Neural Information Retrieval*. Now Foundations and Trends, 2018.

Nguyen, T., Rosenberg, M., Song, X., Gao, J., Tiwary, S., Majumder, R., and Deng, L. MS MARCO: A human generated machine reading comprehension dataset. In *CoCo@ NIPS*, 2016.

Nie, Y., Williams, A., Dinan, E., Bansal, M., Weston, J., and Kiela, D. Adversarial NLI: A New Benchmark for Natural Language Understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 4885–4901, 2020.

Petroni, F., Piktus, A., Fan, A., Lewis, P., Yazdani, M., De Cao, N., Thorne, J., Jernite, Y., Karpukhin, V., Maillard, J., et al. KILT: a Benchmark for Knowledge Intensive Language Tasks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 2523–2544, 2021.

Qu, Y., Ding, Y., Liu, J., Liu, K., Ren, R., Zhao, W. X., Dong, D., Wu, H., and Wang, H. 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*, pp. 5835–5847, 2021.

Rajpurkar, P., Zhang, J., Lopyrev, K., and Liang, P. SQuAD: 100,000+ Questions for Machine Comprehension of Text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 2383–2392, 2016.

Reimers, N. and Gurevych, I. The Curse of Dense Low-Dimensional Information Retrieval for Large Index Sizes. 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)*, pp. 605–611, 2021.

Ren, R., Qu, Y., Liu, J., Zhao, W. X., She, Q., Wu, H., Wang, H., and Wen, J.-R. RocketQAv2: A Joint Training Method for Dense Passage Retrieval and Passage Re-ranking. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 2825–2835, 2021.

Seo, M., Lee, J., Kwiatkowski, T., Parikh, A., Farhadi, A., and Hajishirzi, H. Real-Time Open-Domain Question Answering with Dense-Sparse Phrase Index. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4430–4441, 2019.

Van den Oord, A., Li, Y., and Vinyals, O. Representation Learning with Contrastive Predictive Coding. *arXiv e-prints*, pp. arXiv–1807, 2018.

Xiong, L., Xiong, C., Li, Y., Tang, K.-F., Liu, J., Bennett, P. N., Ahmed, J., and Overwijk, A. Approximate Nearest Neighbor Negative Contrastive Learning for Dense Text Retrieval. In *International Conference on Learning Representations*, 2020.

A. Token Filtering

Representation filtering is often applied in practice to reduce the index size for efficient search (Min et al., 2021). For phrase retrieval, tokens that are not likely to be a start/end position of an answer are filtered out using a trained filter classifier based on a logit score for each token. Only tokens with a score larger than a specific threshold are kept. After the filtering, the index is compressed using optimized product quantization (Ge et al., 2013).

A.1. Token Filter Threshold

A filter threshold for the token filter determines a trade-off between the index size (efficiency) and retrieval accuracy (Figure 3). Interestingly, we find that token filtering can even improve retrieval accuracy. As we increase a threshold from a very small value (not filtering), the accuracy fluctuates but generally increases until a specific threshold because the filter successfully reduces the number of candidates, making prediction easier. After that threshold, the accuracy drops quickly because the filter starts to leave out many correct tokens.

However, finding the peak retrieval accuracy requires a manual search of different thresholds after indexing and evaluating. Since using it as a validation metric is expensive, we first select the best checkpoint based on retrieval accuracy without performing any token filtering. Especially when the token filter is in the middle of training, the peak threshold will vary, and using a specific fixed threshold would not be fair. Also, the best threshold changes depending on the corpus size, so choosing a threshold for the full corpus based on a smaller corpus is not straightforward.

A.2. Token Filter Training and Valiation

Lee et al. (2021a) jointly train a token filter classifier with a dual encoder. It is convenient in that an additional training process is not required, while we should tune on the weight for a loss before adding to the overall training loss. Training pushes phrase vectors into two moving corns toward the start and end vectors since a logit is a dot product score between a phrase vector and a start/end vector. It has two potential disadvantages: (1) phrase representations are concentrated on the part of the entire feature space, so the expressiveness of the model is not fully exploited, and (2) optimization is more difficult because of the moving targets.

To address the issues, we change to train a token filter after training a phrase encoder. We could expect that the two-stage training process encourages phrase representations to be distributed over the space. Moreover, we can validate the token filter separately due to the separate training process and pick the best one. We can not decide the threshold during the filter training, so we use the AUC-PR metric for filter validation by measuring precision and recall by sweeping all threshold values.

B. Training Techniques for DensePhrases

To learn better representations, the dual encoder is first pretrained with question-answer pairs generated by a question generation model as a data augmentation mainly for better reading comprehension capability and then fine-tuned with original question-answer pairs. Also, knowledge distillation (Hinton et al., 2015) from a stronger reading comprehension model based on a cross encoder to the dual encoder is performed.

Documents in the reading comprehension dataset used for the training take only a tiny portion of the entire Wikipedia, and only a small number of negatives for each question-phrase pair are contrasted compared to billion-scale possible phrase candidates in the test time. The query encoder can be further fine-tuned to reduce this discrepancy between training and inference while fixing the phrase encoder and the index by maximizing the likelihood of the gold answer among retrieved phrases for each question. This process is called *query-side fine-tuning*.

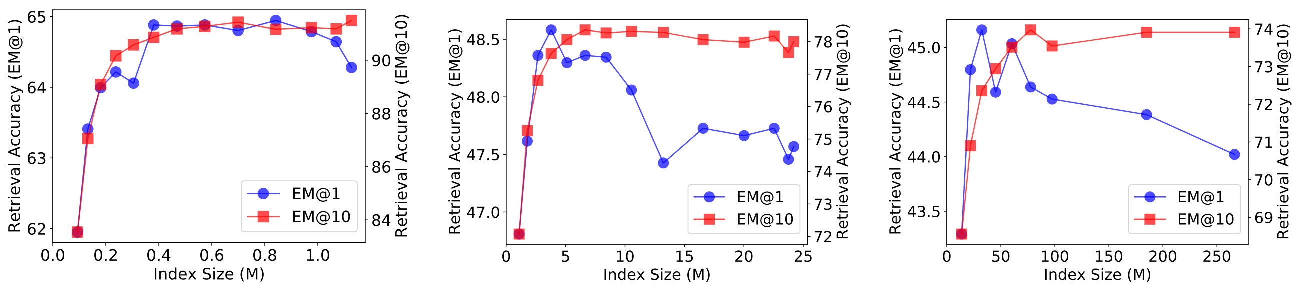
C. Training details

We use almost the same training hyperparameters of the original DensePhrases except for the batch size $B = 48$. We use the number of training epochs to 2 with the generated question-answer pairs and increase the number of training epochs to 10 with the standard reading comprehension dataset for more careful validation. We set $\lambda(p^-)$ to 256 for $p^- \in N_{\text{inb}} \cup N_{\text{prb}}$, otherwise to 1. We set $k = 10$ and $h = 1$ for the hard negative mining and sampling. Our token filter achieves a much higher AUC-PR value than the filter of the original DensePhrases model (e.g., 0.348 vs. 0.307). We use a filter threshold of -3 for the index with the full corpus.

D. Discussion on Hard Negatives

With hard subcorpora, a model trained with hard negatives shows higher validation accuracy than a model without hard negative training before query-side fine-tuning, but the order changes after query-side fine-tuning (Figure 2). We blame it on the similarity between the hard corpus construction and the hard negative mining process. Therefore, we may have to rely on another metric (including retrieval accuracy on random subcorpus) only for validating the effectiveness of hard negatives.

Interestingly, hard negatives improve in-domain accuracy but decrease out-of-domain accuracy. Since gathered hard negative passages are close to the original training data, it



(a) C_0 (b) $C_{1/100}$ (c) $C_{1/10}$

Figure 3: The trade-off between the index size and validation retrieval accuracy by changing filter thresholds on random pseudo corpora with different sizes, (a) C_0, (b) $C_{1/100}$, and (c) $C_{1/10}$. A threshold that gives better accuracy with a smaller index size exists. EM@1 (blue) accuracy is more unstable than EM@10 (red). Interestingly, the index size of peak EM@1 is smaller than that of peak EM@10.

can improve the performance of questions from the same domain but could overfit and may lose the generalization ability. This observation solicits better hard negative mining methods.

E. Related Work

Dense retrieval Retrieving relevant documents for a query (Mitra & Craswell, 2018) is crucial in many NLP applications like open-domain question answering and knowledge-intensive tasks (Petroni et al., 2021). Dense retrievers typically build a search index for all documents by pre-computing the dense representations of documents using an encoder. Off-the-shelf libraries for a maximum inner product search (MIPS) (Johnson et al., 2019; Guo et al., 2020) enable model training and indexing to be developed independently (Lin, 2022). However, both training dense retrievers and building indexes should take into account the final retrieval accuracy. In this respect, we aim to close the gap between training and inference of dense retrievers.

Phrase retrieval Phrase retrieval (Seo et al., 2019) directly finds an answer with MIPS from an index of contextualized phrase vectors. This removes the need to run an expensive reader for open-domain QA. As a result, phrase retrieval allows real-time search tens of times faster than retriever-reader approaches as an alternative for open-domain QA. DensePhrases (Lee et al., 2021a) removes the requirement of sparse features and significantly improves the accuracy from previous phrase retrieval methods (Seo et al., 2019; Lee et al., 2020). Lee et al. (2021b) show how retrieving phrases could be translated into retrieving larger units of texts like a sentence, passage, or document, making phrase retrieval a general framework for retrieval. Despite these advantages, phrase retrieval requires building a large index from billions of representations. In this work, we focus on improving phrase retrieval with more efficient validation.

Validation of dense retrieval Careful validation is essential for developing machine learning models to find a better configuration (Melis et al., 2018) or avoid falling to a wrong conclusion. However, many works on dense retrieval do not clearly state the validation strategy, and most of them presumably perform validation on the entire corpus. It is doable but quite expensive[7] to perform frequent validation and comprehensive tuning. Hence, it motivates us to devise efficient validation for dense retrieval. Like ours, Hofstätter et al. (2021) construct a small validation set by sampling queries and using a baseline model for approximate dense passage retrieval but limited to early stopping.

Hard examples Adversarial data collection by an iterative model (or human) in the loop process aims to evaluate or reinforce models' weaknesses, including the robustness to adversarial attacks (Kaushik et al., 2021; Bartolo et al., 2021; Nie et al., 2020; Kiela et al., 2021). In this work, we construct a compact corpus from a pre-trained dense retriever for efficient validation. Also, we extract hard negatives from retrieval results of the previous model for better dense representations.

F. Broader Impact

Our work demonstrates that thorough validation is crucial for the accurate and efficient development of phrase retrieval with large corpus. Also, it could be especially beneficial in real applications where the corpus size is much larger than

[7]For example, dense passage retrieval (DPR) (Karpukhin et al., 2020) takes 8.8 hours on 8 GPUs to compute 21-million passage embeddings and 8.5 hours to build a FAISS index. Also, ColBERT (Khattab & Zaharia, 2020) takes 3 hours to index 9M passages in the MS MARCO dataset (Nguyen et al., 2016) using 4 GPUs.

benchmark datasets. Moreover, we prove that modeling and training methods should be designed closely to retrieval in inference time. Despite its remarkable efficiency and flexibility, phrase retrieval has been relatively less studied than passage retrieval. We believe that our work can encourage more study on phrase retrieval with an efficient development cycle. Furthermore, we hope that our findings could be extended to dense retrieval in general to help a wide variety of applications.

You can have as much text here as you want. The main body must be at most 4 pages long. For the final version, one more page can be added. If you want, you can use an appendix like this one, even using the one-column format.