Dense Hierarchical Retrieval for Open-Domain Question Answering

Ye Liu[1]*, **Kazuma Hashimoto**[2], **Yingbo Zhou**[2], **Semih Yavuz**[2], **Caiming Xiong**[2], **Philip S. Yu**[1]
[1] University of Illinois at Chicago, Chicago, IL, USA
[2] Salesforce Research, Palo Alto, CA, USA
{yliu279, psyu}@uic.edu,
{yingbo.zhou, k.hashimoto, syavuz, cxiong}@salesforce.com

Abstract

Dense neural text retrieval has achieved promising results on open-domain Question Answering (QA), where latent representations of questions and passages are exploited for maximum inner product search in the retrieval process. However, current dense retrievers require splitting documents into short passages that usually contain local, partial and sometimes biased context, and highly depend on the splitting process. As a consequence, it may yield inaccurate and misleading hidden representations, thus deteriorating the final retrieval result. In this work, we propose Dense Hierarchical Retrieval (DHR), a hierarchical framework which can generate accurate dense representations of passages by utilizing both macroscopic semantics in the document and microscopic semantics specific to each passage. Specifically, a document-level retriever first identifies relevant documents, among which relevant passages are then retrieved by a passage-level retriever. The ranking of the retrieved passages will be further calibrated by examining the document-level relevance. In addition, hierarchical title structure and two negative sampling strategies (i.e., *In-Doc* and *In-Sec* negatives) are investigated. We apply DHR to large-scale open-domain QA datasets. DHR significantly outperforms the original dense passage retriever, and helps an end-to-end QA system outperform the strong baselines on multiple open-domain QA benchmarks.

1 Introduction

The goal of open-domain Question Answering (QA) is to answer a question without pre-specified source domain (Kwiatkowski et al., 2019). One of the most prevalent architectures in open-domain QA is the retriever-reader approach (Chen et al., 2017; Lee et al., 2019). Given a question, the task of the retrieval stage is to identify a set of relevant contexts within a diversified large corpus (e.g., Wikipedia). The reader component then consumes the retrieved evidence as input and predicts an answer. In this paper, we focus on improving the efficiency and the effectiveness of the retrieval component, which in turn leads to improved overall answer generation for open-domain QA.

Question: Who wrote the first declaration of human rights?
Answer: Cyrus
Gold Passage, History of human rights:
After his conquest of Babylon in 539 BC, the king issued the **Cyrus** cylinder, discovered in 1879 and seen by some today as the first human rights document....
DPR Retrieved Passage, John Peters Humphrey:
John Peters Humphrey, OC (April 30, ... He is most famous as the author of the first draft of the Universal Declaration of Human Rights....
DPR Retrieved Passage, Drafting of the Universal Declaration of Human Rights: The Universal Declaration of Human Rights ... by Drafting Committee ... Members of the Commission who contributed significantly to the creation of the Declaration included Canadian John Peters Humphrey of the United Nations Secretariat, Eleanor Roosevelt ...

Figure 1: An example of distracting passages in Natural Question (Kwiatkowski et al., 2019). The first DPR retrieved passage shares similar semantics with the gold passage. The document title of the second DPR retrieved passage matches most question tokens. Both of the retrieved passages tend to result in a wrong answer.

Pretrained transformer models, such as BERT (Devlin et al., 2019), are widely used in recent studies on the retriever-reader framework (Asai et al., 2019; Lewis et al., 2020; Guu et al., 2020). To serve as input to the retriever, documents are split into short passages, and in the Dense Passage Retrieval, DPR (Karpukhin et al., 2020), a dual encoder framework is applied to encode questions and the split passages separately. State-of-the-art dense retrievers outperform sparse term-based retrievers, like BM25 (Robertson and Zaragoza, 2009), but they suffer from several weaknesses. First, due to the lack of effective pruning strategy, extracting relevant passages from a large corpus undergoes an efficiency issue especially in the inference time. Second, given a

* Work was done when the first author was a research intern at Salesforce Research

Findings of the Association for Computational Linguistics: EMNLP 2021, pages 188–200
November 7–11, 2021.

question, many passages may comprehend similar topics with subtle semantic difference. This fact requires the retriever and the reader to encode passages to their accurate semantic representations, which is an overwhelmed task. Moreover, passages contain only local and specific information, thus easily leading to distracting representations. As illustrated in Figure 1, distracting passages with similar semantics may lead to a wrong answer.

To alleviate these issues, we present a *Dense Hierarchical Retriever* (DHR) framework, which consists of a dense document-level retriever and a dense passage-level retriever. Document-level retriever aims at capturing coarse-grained semantics of documents in the sense that the embeddings of questions and their relevant documents are positively correlated. The goal of document-level retriever is to prune answer-irrelevant documents and returns relevant ones, which will serve as a refined corpus to feed into passage-level retriever. Given relevant documents consisting of passages of similar topics, the passage-level retriever intends to identify the essential evidences, which may contribute to a correct answer.

In order to empower DHR, we formalize the hierarchical information of documents as a tree structure with two types of nodes, title nodes and content nodes. Then, a document is easily represented by its document summary, and a passage is represented by a hierarchical title list concatenated with its content. The benefit of using the hierarchical approach and exploiting the hierarchical information is three-fold: 1) Coarse-grained information explicitly or implicitly covered in the document will guide the passage-level retriever to deviate from a fallacious embedding function; 2) Passage-level retriever, a fine-grained component, will provide essential capability of identifying the necessary relevant evidences among similar passages; 3) Document-level retriever prunes substantial amount of irrelevant and peripheral documents, and triggers a much faster inference. To further enhance the ability of the passage-level retriever in detecting gold passages among similar passages, we propose two negative sampling strategies (i.e., *In-Doc* and *In-Sec* negative sampling).

Our main contributions are summarized as: 1) We propose a hierarchical dense retrieval on open-domain QA and achieve a fast inference speed with high retrieval precision; 2) The hierarchical information is used in a more structural way, which leads to a meaningful and global passage representation consistent with its document; 3) We conduct comprehensive experiments with state-of-the-art approaches on multiple open-domain QA datasets. Our empirical results demonstrate that we achieve comparable or better results in the open-retrieval setting. Extensive ablation studies on various components and strategies are conducted.

2 Notations and Preliminaries

2.1 Text Retrieval for Open-Domain QA

In open-domain QA, we are given a large corpus (e.g., Wikipedia) $\mathcal{C} = \{d_1, d_2, \ldots, d_N\}$, where each document d_i is formed by a sequence of passages, $d_i = \{p_1^{(i)}, p_2^{(i)}, \ldots, p_l^{(i)}\}$. The task of end-to-end open-domain QA can be formulated with a retriever-reader approach (Chen et al., 2017); we first find a passage (or a set of passages) relevant to a given question, and then use a reading comprehension model to actually derive its answer. It is common that we retrieve top-k passages to be examined by the reading step. The retrieval step is crucial, affecting the reading comprehension step.

2.2 Dual Encoder Retrieval Model

In the retrieval process, a commonly used approach referred as a dual encoder model (Bromley et al., 1993) consists of a question encoder E_Q and a context encoder E_P, which encodes the question and the passage to l dimensional vectors, respectively. Unlike sparse term-based retrievers that rely on term frequency and inverse document frequency, dense neural retrievers formulate a scoring function between question q and passage p by the similarity of their embeddings, formalized as

$$f_\theta(q, p) = \left\langle E_Q^\theta(q), E_P^\theta(p) \right\rangle,$$

where $E_Q^\theta(q) \in \mathbb{R}^l$ and $E_P^\theta(p) \in \mathbb{R}^l$ are the embeddings, and $\langle \cdot, \cdot \rangle$ represents a similarity function such as doc product and cosine similarity. Typically, E_Q^θ and E_P^θ are two large pre-trained models, e.g., BERT (Devlin et al., 2019). We use different subscripts and same superscript θ to emphasize that these are two language models and fine tuned jointly. DPR (Karpukhin et al., 2020) is one of the representative models in this model family.

Contrastive Learning. Given a training set $\mathcal{S} = \{(q_1, y_1), \cdots, (q_m, y_m)\}$, we can create a training set $\mathcal{T} = \{(q_1, p_1^+, \mathcal{P}_1^-), \cdots, (q_m, p_m^+, \mathcal{P}_m^-)\}$ for the retrieval, where $q_i, y_i, p_i^+, \mathcal{P}_i^-$ are a question,

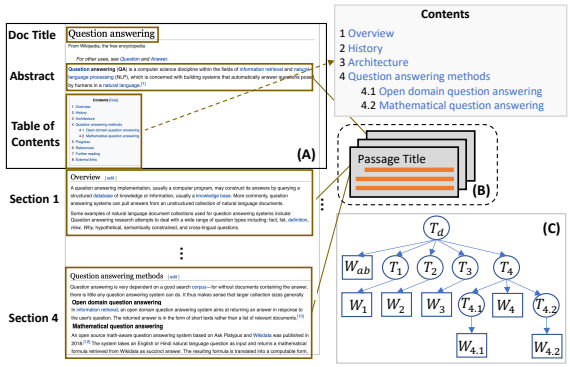

Figure 2: An illustration of a typical Wikipedia page. (A) Document representation. (B) Passage representations. (C) Hierarchical title structure, i.e., title tree.

its answer, its positive passage and a set of negative passages, respectively. All the selections of positive passages or documents in this paper are described in Appendix 4.3. For the set of negative passages $\mathcal{P}_i^-$, it is constructed in two ways: 1) BM25 negatives: top BM25-based passages not containing the answer; 2) In-batch negatives: passages paired with other questions appearing in the same mini batch.

Training. The objective of training is to learn an embedding function such that relevant pairs of questions and passages will have higher similarity than the irrelevant ones. For each training instance $(q_i, p_i^+, \mathcal{P}_i^-) \in \mathcal{T}$, we contrastively optimize the negative log-likelihood of each positive passages against their negative passages,

$$\text{loss}_\theta(q_i, p_i^+, \mathcal{P}_i^-) = -\log \frac{e^{f_\theta(q_i, p_i^+)}}{\sum_{p \in \{p_i^+\} \cup \mathcal{P}_i^-} e^{f_\theta(q_i, p)}}.$$

Inference. During inference, we encode the given question q and conduct the maximum inner product search between $E_Q^\theta(q)$ and $E_P^\theta(p)$ for every passage p. Then a ranked list of k most relevant passages are served as input of the reader.

3 Dense Hierarchical Retrieval (DHR)

This section presents our Dense Hierarchical Retrieval (DHR) model, which consists of a Dense Document-level Retrieval (DHR-D) and a Dense Passage-level Retrieval (DHR-P). Figure 3 shows an overview of our proposed method.

3.1 Structural Document

A structured web article like a Wikipedia page in Figure 2 contains a document title, abstract, table of contents and different levels of sections consisting of titles and paragraphs. To better leverage the hierarchical information of the document, we formalize the structural document as a tree structure called title tree with the hierarchical title structure being the backbone. The title tree uses the document title T_d as the root, the section titles of different levels as intermediate nodes, and the textual content under the same title as a leaf. Note that there are two types of nodes namely title node and content node. Each title or content will appear in the tree exactly once.

3.2 Dense Document-level Retrieval (DHR-D)

Dense Document-level Retrieval (DHR-D) aims at capturing the semantics of the documents in the sense that the embeddings of the questions and their relevant documents are positively correlated. DHR-D employs a BERT-based dual encoder model consisting of a question encoder E_Q^ϕ and a document encoder E_D^ϕ, where ϕ emphasizes that two encoders are trained jointly. The relevance score of a document to a question is computed by dot product of their dense representation vectors:

$$f_\phi(q, d) = \langle \mathrm{E}_Q^\phi(q),\ \mathrm{E}_D^\phi(d) \rangle, \qquad (1)$$

where $\mathrm{E}_Q^\phi(q) \in \mathbb{R}^l$, $\mathrm{E}_D^\phi(d) \in \mathbb{R}^l$ and $\langle \cdot \rangle$ represents the dot product.

Document Representation. In order to enable the document encoder to capture holistic view of the documents covering their essential topics (Chang et al., 2020), we use their summary as input to E_D^ϕ. We define *document summary* as a concatenation of title T_d, abstract W_{ab}, and the linearized table of contents T_{table}. We linearize the table of contents by following a pre-order traversal on only title nodes of the title tree excluding the root node T_d. Separating each title by the special token `[SEP]` (or comma), we finalize the representation of table of contents as $T_{table} = T_1$ `[SEP]` $T_{1.1}$ `[SEP]` $\cdots$ `[SEP]` $T_{i.\ldots.l}$. Then the final representation of the document summary to be consumed by E_D^ϕ is defined as $d =$ `[CLS]` T_d `[SEP]` W_{ab} `[SEP]` T_{table} `[SEP]`.

Negative Sampling. Recall that given a training sample $(q_i, y_i) \in \mathcal{S}$ for open-domain QA, we can create a contrastive training instance with $(q_i, d_i^+, \mathcal{D}_i^-)$ for the retrieval, where $q_i, y_i, d_i^+, \mathcal{D}_i^-$ correspond to question, answer, positive document and a set of negative documents, respectively. With respect to the negative documents, besides leveraging in-batch negatives similar to DPR (Karpukhin

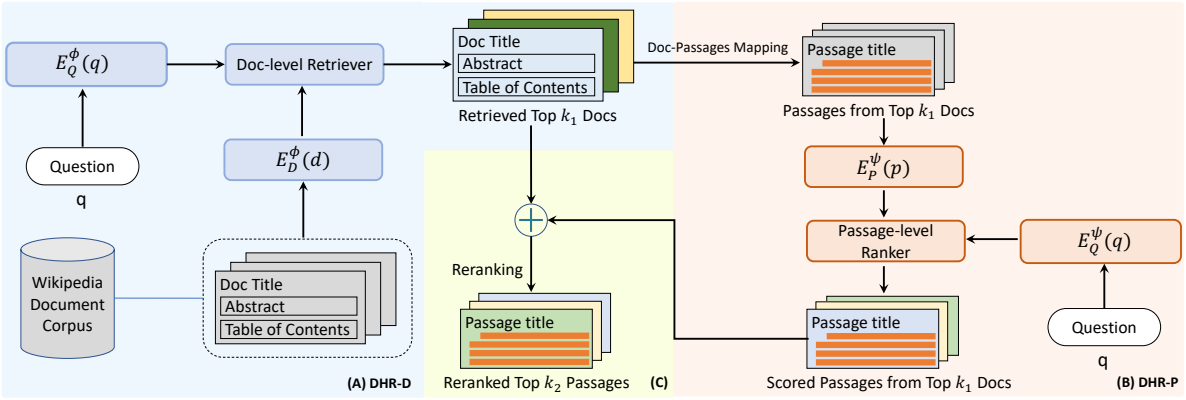

Figure 3: An overview of DHR. During inference, document-level retriever first retrieves top-k_1 documents (as shown in (A)). Then, passage-level retriever scores the passages in top-k_1 documents (as shown in (B)). At last, DHR reranks passages based on two levels of relevance scores and return top-k_2 passages (as shown in (C)).

et al., 2020), we introduce two negative document sampling strategies: 1) *Abstract negatives*: we select top-ranked documents by BM25 whose abstract contains most question tokens but the whole document content doesn't contain the gold answer; 2) *All-text negatives*: we select top-ranked documents by BM25 whose whole document content contains most question tokens but doesn't contain the gold answer.

We train DHR-D following the strategy in Section 2.2. We also optimize the negative log-likelihood. The only difference here is that the examples are documents instead of passages. During inference, DHR-D extracts a set of relevant documents for each question which then serve as a more focused evidence pool to be processed by Dense Passage-level Retrieval (DHR-P). The overall inference process will be elaborated in Section 3.5.

3.3 Dense Passage-level Retrieval (DHR-P)

Given relevant documents from DHR-D, the goal of our Dense Passage-level Retrieval (DHR-P) is to detect the most crucial evidence that may contribute to answer the question. Another dual encoder model is used in DHR-P with one BERT representing the question encoder E_Q^{ψ} and the other BERT representing the passage encoder E_P^{ψ}. Notice that we use ψ here to distinguish between the pair of encoders used for DHR-P and DHR-D. Similarly, the relevance score between passage p and question q is calculated by the dot product of their semantic embeddings:

$$f_{\psi}(q,p) = \langle \mathrm{E}_Q^{\psi}(q),\ \mathrm{E}_P^{\psi}(p) \rangle. \qquad (2)$$

Passage Representation. Here we describe two major differences of our passage representation from the previous work. First, instead of naively splitting document into passages, we only allow the passages within the same section to split. In other words, splitting can only happen within each leaf in the title tree. In this way, each passage will be semantically more consistent. Second, to magnify the differences between passages of similar topics, we augment each passage with a passage title. We define a *passage title* as the concatenation of titles on the path from the root node to its content leaf using special token `[SEP]` (or comma) as the separator inserted in between. Then the passage p can be represented as `[CLS]` T_{d_p} `[SEP]` T_{i_1} `[SEP]` T_{i_2} `[SEP]` $\cdots$ `[SEP]` T_{i_n} `[SEP]` W_p `[SEP]`, where d_p represents the document it belongs to and W_p indicates the passage content.

Negative Sampling. In order to improve the capability of detecting the essential evidence among similar passages, we propose two hard negative sampling strategies for DHR-P. Besides using BM25 negatives and in-batch negatives, we propose *In-Doc negative* and *In-Sec negative*. While *In-Sec* negatives are the passages which are in the same section with the positive passage but don't contain the answer. *In-Doc* negatives are passages which are in the same document with the positive passage but don't contain the answer.

3.4 Iterative Training

Inspired by improvement of using semantically related negative examples generated by previous checkpoint in ANCE (Xiong et al., 2020a), we adopt an iterative training scheme for both DHR-D and DHR-P. More precisely, we use the retriever resulting from the initial phase of training to generate hard negative examples, which may be semanti-

cally quite related to the question but don't contain the answer. From the perspective of adversarial training (Madry et al., 2017), these negative examples can also be regarded as adversarial examples. Hence, training on these examples will increase the robustness of the models. Therefore, in the second iteration we further train DHR-D and DHR-P with generated negative examples together with the negative examples used in the first iteration.

3.5 Inference

Before inference, we apply the document encoder E_D^{ϕ} to all the documents and index them using efficient FAISS (Johnson et al., 2019) offline. Given a question q at inference time, we compute its embeddings $\mathrm{E}_Q^{\phi}(q)$ and $\mathrm{E}_Q^{\psi}(q)$ for DHR-D and DHR-P respectively. Then, we first retrieve top-k_1 relevant documents using the index built by DHR-D. The passages of the top-k_1 retrieved documents then serve as a refined corpus, upon which E_P^{ψ} is applied to select top-k_2 passages.

Although DHR-D has already helped pruning of irrelevant documents before passage retrieval, we still leverage document-level similarity score it offering in addition to passage similarity score for the final ranking. Thus, we define the final passage score function as a combination of relevance scores provided by DHR-D and DHR-P:

$$f(q, p_i) = f_{\psi}(q, p_i) + \lambda \cdot f_{\phi}(q, d_{p_i}) \quad (3)$$

where d_{p_i} is the document p_i belongs to and λ is the coefficient controlling the balance. We noticed that the relevance scores of DHR-D and DHR-P are in the close scale for all our experiments. $\lambda \in [0.5, 1]$ is a quite robust choice for desired performance.

4 Experimental Setup

In this section, we describe the dataset and basic setup for experiments.

4.1 Wikipedia Data Pre-processing

Following Karpukhin et al. (2020), we use the English Wikipedia dump from Dec. 20, 2018 as the source documents for answering questions. We first apply the WikiExtractor to extract the clean, textual documents with hierarchical title list from the Wikipedia dump, which removes semi-structured data, such as tables, infoboxes, lists, as well as disambiguation pages. In DPR (Karpukhin et al., 2020), all the texts under the same document are first concatenated as a single block, which is then split into multiple blocks of fixed-length passages of 100 words, discarding the blocks of shorter than 100 words. We follow a different, more principled strategy to avoid ending up with abruptly broken and unnatural passages. To this end, we concatenate the text under the same section and split each section into multiple, disjoint text blocks, whose maximum length is not over 100 words. Following this strategy, we obtain 25,992,490 passages from 5,380,681 documents.

Dataset	Train		Dev		Test
NQ	79,168	59,906	8,757	6,610	3610
TriviaQA	78,785	60,314	8,837	6,753	11,313
WebQuestions	34,17	2,432	361	275	2,032
CuratedTrec	1,353	1,114	133	114	694

Table 1: Number of questions in each QA dataset. The two columns of **Train** and **Dev** denote the original examples in the dataset and the actual questions used. See Section 4.3 for more details.

4.2 Question Answering Datasets

We use four QA datasets that have been most commonly used benchmarks for open-domain QA evaluation (Lee et al., 2019; Karpukhin et al., 2020):

Natural Questions (NQ) (Kwiatkowski et al., 2019) consists of questions mined from real Google search queries, for which the answers are spans in Wikipedia documents identified by annotators.

TriviaQA (Joshi et al., 2017) contains a set of trivia questions with answers that were originally scraped from the Web.

WebQuestions (Berant et al., 2013) consists of questions selected using Google Suggest API, where the answers are entities in Freebase.

CuratedTREC (TREC) (Baudiš and Šedivỳ, 2015) is a collection of questions from TREC QA tracks as well as various Web sources, intended for open-domain QA from unstructured text.

4.3 Selection of positive documents and passages.

To determine the positive passage (or document), we assign it as the passage (or document) containing the gold context of the answer when it is given by human annotation; otherwise we feed the question to a BM25 system to retrieve the top-1 passage (or document) containing the answer as the positive passage (or document).

For Natural Questions, since the relevant context and document title are provided, we directly use

Retriever	NQ			TriviaQA			WebQ			TREC		
	Top-1	Top-20	Top-100	Top-1	Top-20	Top-100	Top-1	Top-20	Top-100	Top-1	Top-20	Top-100
BM25	-	59.1	73.7	-	66.9	76.7	-	55.0	71.1		70.9	84.1
BM25*	18.48	60.19	75.98	40.02	72.78	81.03	17.03	56.45	73.77	27.89	74.97	87.92
1-iter												
DPR	45.87	79.97	85.87	-	79.4	85.0	-	73.2	81.4	-	79.8	89.1
DPR*	40.08	81.05	88.31	52.94	80.43	85.40	35.78	73.87	81.64	36.03	80.70	90.49
(Lu et al., 2020)	52.0	82.8	88.4	-	-	-	-	-	-	-	-	-
DHR	**55.37**	**85.07**	**89.92**	**54.40**	**80.81**	**85.69**	**36.86**	**73.98**	**82.69**	**48.27**	**84.01**	**91.26**
2-iter												
DPR	52.47	81.33	87.29	-	-	-	-	-	-	-	-	-
DPR*	52.67	84.67	89.95	53.89	79.68	85.63	38.44	75.19	82.87	41.35	79.68	91.21
DHR	**57.04**	**85.60**	**90.64**	**55.08**	**80.76**	**85.97**	**41.73**	**75.29**	**83.05**	**48.42**	**84.17**	**91.34**

Table 2: Top-1, Top-20 and Top-100 passage-level retrieval accuracy on test sets, measured as the percentage of top 1/20/100 retrieved passages that contain the answer. * represents the reproduction on our processed Wikipedia data. We bold the best performance and underline the second best performance.

the provided document title to find our processed corresponding document and use the relevant context map to our processed passage in the candidate pool. The questions are discarded when the matching is failed due to different Wikipedia versions or pre-processing. Because only pairs of questions and answers are provided in TREC and TriviaQA, we use the highest-ranked passage from BM25 that contains the answer as the positive passage and its belonging document as the gold document. If none of the top 100 retrieved passages has the answer, the question will be discarded. For WebQ, since it contains the gold title in the Freebase, we try both ways (matching and BM25 ranking) and find that using the highest-ranked passage from BM25 as the positive passage and its belonging document as the positive document can produce better performance than using the gold title from Freebase. We think it is due to the discrepancy between Freebase and Wikipedia corpus. Table 1 shows the number of questions in training/dev/test sets for all the datasets and the actual questions used in training and dev sets.

5 Experiments

In this section, we evaluate the performance of our Dense Hierarchical Retriever (DHR[1]), along with analysis on how each component affects the results and the retrieval time efficiency. For the retrieval implementation detail, please refer to Appendix A.

5.1 Main Results

In Table 2, we report the retrieval performance of different systems on four QA datasets in terms of Top-1, 20 and 100 passage-level retrieval accuracy. For a fair comparison, we first re-implemented the DPR method on the Wikipedia data processed with our passage construction strategy defined in Section 4.1, which is denoted as DPR* in Table 2. The retrieval performance of DPR* outperforms the original DPR on all datasets, except the Top-1 retrieval performance on NQ, showing the clear advantage of using our more principled in-section splitting strategy, which can better preserve the contextual consistency in each passage. Secondly and perhaps most importantly, we would like to highlight the benefit of our proposed hierarchical dense retrieval method (DHR) over the baseline DPR*. As shown in Table 2, DHR consistently and significantly outperforms DPR* across the board over four datasets we conduct experiments on. Most notably, it can provide up to 12% and 4% absolute improvement in top-1 and top-20 retrieval accuracy over DPR* on NQ and TREC benchmarks. Also, we observe that DHR's improved retrieval performance translates well on to the iterative training setting. More precisely, using the wrong passages that are found semantically relevant by the first iteration model as negatives for training the second iteration greatly helps further improve the performance of DHR. Although the iterative training significantly boosts the performance of DPR*, our proposed DHR model still significantly outperforms DPR* across the four datasets, which is consistent with the conclusion from the first iteration setting. Finally, we note that DHR also improves upon a recent work (Lu et al., 2020), which achieves significant improvement over DPR using better negative samples.

[1] https://github.com/yeliu918/DHR

Retriever	Top-1	Top-5	Top-20	Top-100
BM25	28.95	54.21	71.97	83.88
DHR-D(Abs)	65.32	82.85	88.75	92.35
DHR-D(All)	64.04	81.81	87.71	92.11
DHR-D(Abs)+T	**68.28**	**83.80**	**89.28**	**92.83**
2-iter				
DHR-D(Abs)+T	**71.86**	**85.35**	**90.30**	**93.16**

Table 3: Top-1, Top-5, Top-20 and Top-100 document-level retrieval accuracy on NQ test sets. Abs denotes *Abstract Negative*. All means *All-text Negative*. T means that we add the table of contents into the document context.

Retriever	Top-1	Top-5	Top-20	Top-100
DPR*	40.08	66.79	81.05	88.31
DHR-P+Tc	43.74	68.67	81.42	88.75
DHR-P+Tt	43.67	68.39	81.05	88.81
DHR-P(Sec)+Tc	50.17	71.80	82.16	88.12
DHR-P(Doc)+Tc	**51.61**	**73.16**	**82.87**	**89.16**
2-iter				
DHR-P(Sec)+Tc	54.46	75.54	84.99	90.19
DHR-P(Doc)+Tc	**55.12**	**76.06**	**85.01**	**90.19**

Table 4: Top-1, Top-5, Top-20 and Top-100 passage-level retrieval accuracy on NQ test sets. Tc and Tt denote using comma and `[SEP]` to separate the passage title, respectively. Sec denotes using *In-Sec negative*. Doc means using *In-Doc negative*.

5.2 Ablation Study

To further understand how each component of DHR works, we conduct several additional experiments on both Document-level retrieval and Passage-level retrieval on the NQ dataset.

Ablation Study on DHR-D. From Table 3, our Doc-level retrieval accuracy greatly outperforms the result of BM25, which shows the efficiency of our dense document-level retriever. Comparing the retriever results of using *Abstract negative* with *All-text negative* in lines 2 and 3, using *Abstract negative* outperforms the performance of using *All-text negative*, which may due to the noisy context bringing from the whole document context harming the performance.

We test the influence of whether uses the table of contents as the context in the document-level retriever. As shown in the bottom block of Table 3, the table of contents can improve the performance of document-level retrieval considerably, which demonstrates our assumption that the table of contents can be viewed as the highlight or summarization of the document contents.

Ablation Study on DHR-P. To fairly compare with DPR, all the results of DHR-P in this section are from retrieving the whole passage corpus without the help of DHR-D to retrieve the relevant documents. We introduced two different ways to linearize the passage title tree in Table 4. The comparison results of Tc and Tt show that using a comma as a separator is better than using a special token `[SEP]`, and containing the passage title with the passage context is better than without it. We think it shows that the hierarchical passage title can help the passage context capturing more global information from the document and help the retriever achieve better performance.

As shown in the bottom block of Table 4, the *In-Doc negative* and *In-Sec negative* improve the passage-level retrieval accuracy, which verifies the idea that improving the passage-level retrieval to distinguish the positive passage from the other passages in the same document is a simple and effective way. The reason why *In-Doc negative* outperforms *In-Sec negative* is that the number of the passage in the same section is less than the number of the same document passage and the passages in the same document also share the close semantic similarity.

Ablation Study on Reranking. In Table 5, we use the DHR-P model with In-Doc negative and title, which achieves the best performance in Table 4 on the whole passage corpus to compare with the two-step hierarchical retrieval models. DHR w/o rerank denotes the passage-level retrieval result from the passage corpus of Top-k_1 relevant documents without reranking. DHR w/o rerank outperforms DHR-P, demonstrating that the Doc-level retrieval can eliminate the distracting documents which could harm the Passage-level retrieval.

We propose different ways to combine the Doc-level and Passage-level retriever scores to rerank the passages. DHR w rerank is the serial strategy proposed in the paper that we first use the Doc-level retriever to get the Top-k_1 relevant documents and use the Passage-level ranker to score the passage from the retrieved documents and rerank them based on the combination of Doc-level and Passage-level similarity scores. DHR para rerank is a parallel way to rank the passages. Firstly, Doc-level retriever scores all documents and Passage-level retriever scores all passages in the corpus. Then the model aggregates those two scores together for each question. The result in Table 5 shows the effectiveness of our approach and demonstrates

	Top-1	Top-5	Top-20	Top-100
DHR-P(Doc)+Tc	51.61	73.16	82.87	89.16
DHR w/o rerank	52.80	73.82	83.80	89.81
DHR w rerank	**55.68**	**75.51**	**84.96**	**89.85**
DHR para rerank	55.29	75.10	84.24	89.15
2-iter				
DHR-P(Doc)+Tc	55.12	76.06	84.99	90.19
DHR w/o rerank	55.90	76.32	85.18	90.42
DHR w rerank	**56.62**	**76.54**	**85.35**	**90.53**

Table 5: Top-1, Top-5, Top-20 and Top-100 passage-level retrieval accuracy on NQ test sets. DHR para rerank represents the parallel generating the document-level and passage-level similarity scores and add them based on Eq. 3.

	Top-1	Top-5	Top-20	Top-100
1 iter				
DHR(λ=1)	55.68	75.51	84.96	89.85
DHR(λ=0.57)	**55.37**	**75.43**	**85.07**	**89.92**
2-iter				
DHR(λ=1)	56.62	76.54	85.35	90.53
DHR(λ=0.50)	**57.04**	**77.06**	**85.60**	**90.64**

Table 6: Top-1, Top-5, Top-20 and Top-100 passage-level retrieval accuracy on NQ test sets.

that using the Doc-level retriever first to limit the documents to a small relevant set will not harm the overall retrieval performance but help filter out some answer-irrelevant documents. Moreover, both the rerank methods outperform DHR w/o rerank, which shows the necessity of the reranking.

5.3 Hyperparameter Sensitivity Analysis

We analyze the parameter λ, which is used in Eq. 3 as the coefficient of combining doc-level score and passage-level score. We tuned the λ values on different datasets by optimizing Top-20 retrieval accuracy on the development set. We obtained the optimal weight by performing a grid search in the range [0, 2]. We started with step size 0.1 and found the optimal λ_1. Then, we used step size 0.01 in the range $[\lambda_1 - 0.05, \lambda_1 + 0.05]$ to find the optimal λ. From the results in Table 6, we can see that directly adding two scores together (λ=1) can lead to the good performance compared with the best performance model (λ=0.57 first iter, λ=0.5 second iter), which shows the robustness of the model without too many parameters tuning.

For the top-k_1 retrieved documents that are given to passage-level retrieval, it is different with the datasets. We get the best performance when k_1 equals 100 in NQ, 500 in TriviaQA, 500 in WebQ, and 300 in the TREC dataset.

	NQ	TriviaQA	WebQ	TREC
DPR	75.5ms	86.5ms	78.5ms	91.5ms
DHR-D	16.3ms	19.4ms	17.3ms	19.6ms
DHR-P	2.5ms	9.9ms	7.2ms	4.5ms
Speedup	**4.02x**	**2.94x**	**3.20x**	**3.80x**

Table 7: The comparison of retrieval time efficiency between DPR and the proposed DHR.

5.4 Retrieval Time Efficiency

During inference, DPR needs to search the gold passage from the 21-million passages. In contrast, DHR only targets 5.38-million documents and the passages from retrieved top-k_1 documents. As discussed in the previous Section 5.3, k_1 is usually a small number like a few hundred. Therefore, the total amount of searching space decreases from 21-million to around 6-million.

Since document embeddings and passage embeddings are encoded once after the model is trained, so we only discuss the index search time here. We run the best model of the first iteration on the test set twice and calculate the average index search time. We separately present the time cost on the document-level retrieval from the whole document corpus (shown in line DHR-D) and the time cost on the passage-level retrieval from the passage corpus of the retrieved documents (shown in line DHR-P) in Table 7. The total time cost of our method is the addition of the DHR-D and DHR-P phrase time cost. And compared with the time cost in DPR, our proposed approach is nearly 3 to 4 times faster. This is a notable advantage of our method in practice.

5.5 End-to-end QA System

To test the end-to-end QA performance, we follow the DPR use extractive reader constructed by BERT. Given the top k retrieved passages (maximum 100 in our experiments), we combine the passage title, passage token with a special token `[SEP]` and send it to the reader. The reader assigns a passage selection score to each passage. In addition, it extracts an answer span from each passage by determining the start and end indexes and assigns a span score. The best span from the passage with the highest passage selection score is chosen as the final answer. And we declare the implementation detail in Appendix C.1.

Table 8 shows our final end-to-end QA results compared with ORQA (Lee et al., 2019) and

Model	NQ	TriviaQA	WebQ	TREC
BM25+BERT	26.5	47.1	17.7	21.3
ORQA	33.3	45.0	36.4	**30.1**
DPR	41.5	56.8	34.6	25.9
DPR*	42.4	56.9	35.5	26.0
DHR	**43.6**	**57.0**	**36.6**	27.3

Table 8: End-to-end QA (Exact Match) accuracy. The first block of results are copied from their cited papers.

DPR (Karpukhin et al., 2020), measured by exact match with the reference answer. Overall, DHR leads to improvement of the QA scores on all four datasets. For reference, we also experiment our retriever with a generative reader in Appendix C.2.

6 Related Work

Hierarchical Retrieval. Hierarchical sparse retriever got attention in early 2000s. Levinson and Ellis (1992) proposed a multi-level hierarchical retrieval method in database search of conceptual graphs. In Web search, Cui et al. (2003) developed a structured document retriever which exploits both content and hierarchical structure of documents, and returns document elements with appropriate granularity. Bonab et al. (2019) incorporated hierarchical domain information into information retrieval models such that the domain specification resolves the ambiguity of questions. Recently, Nie et al. (2019); Asai et al. (2019) proposed a hierarchical retrieval approach with both paragraph and sentence level retrievers to extract supporting facts for the large-scale machine reading task.

Dense Retrieval with Pre-trained Encoders. With the strong embedding-based ability of the pre-trained model, Lee et al. (2019); Chang et al. (2020) showed the advantage of dual encoder framework with a set of pre-training tasks (Liu et al., 2020b) can achieve strong baselines in the large-scale question-document retrieval task. DPR (Karpukhin et al., 2020) developed a better training scheme using contrastive learning and shows that without the pre-training task, just using a small number of training pairs can achieve state-of-the-art. DPR has been used as an important module in very recent works. Xiong et al. (2020b) extended the DPR to the multi-hop setting (Liu et al., 2020a) and shows that DPR using passage text only to retrieve multi-hop passages can achieve good performance, without the help of the hyperlinks.

Recent research explored various ways to construct better negative training instances for dense retrieval. ANCE (Xiong et al., 2020a) used the retrieval model trained in the previous iteration to discover new negatives and construct a different set of examples in each training iteration. Lu et al. (2020) explored different types of negatives and uses them in both the pre-training and fine-tuning stages. The other direction of recent research works on improving the training strategy in dense retrieval. Rather than using the gold document as distant supervised training of retrieval, Izacard and Grave (2020) leveraged attention score of a reader model to obtain synthetic labels for the retriever. And Sachan et al. (2021) presented the end-to-end supervised training of the reader and retriever. Furthermore, Mao et al. (2020) generated various contexts of a question to enrich the semantics of the questions is beneficial to improve DPR retrieval accuracy. Xiong et al. (2020c) used a pretrained sequence-to-sequence model to generate question-passage pairs for pretraining and proposed a simple progressive pretraining algorithm to ensure the effective negative samples in each batch. A pretrained sequence-to-sequence model is exploited to create question-passage pairs in the zero-shot setting (Ma et al., 2021).

7 Conclusion

In this work, we propose Dense Hierarchical Retrieval (DHR) for open-domain QA and demonstrate that the hierarchical model provides evident benefits in terms of accuracy and efficiency. The hierarchical information is crucial to associate passages with documents such that the passage-level retriever tends to deviate from a misguided embedding function. Contrastive learning using proposed negatives further encourages a robust decision boundary between positives and hard negatives, leading to a meaningful fine-grained retriever. Extensive experiments and analysis on four Open-domain QA benchmarks demonstrate the effectiveness and efficiency of our proposed approach.

Acknowledgements

We would like to thank all the reviewers for their helpful comments. This work is supported by NSF under grants III-1763325, III-1909323, III-2106758, and SaTC-1930941.

References

Akari Asai, Kazuma Hashimoto, Hannaneh Hajishirzi, Richard Socher, and Caiming Xiong. 2019. Learning to retrieve reasoning paths over wikipedia graph for question answering. *International Conference on Learning Representations (ICLR)*.

Petr Baudiš and Jan Šedivỳ. 2015. Modeling of the question answering task in the yodaqa system. In *International Conference of the Cross-Language Evaluation Forum for European Languages*, pages 222–228. Springer.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on freebase from question-answer pairs. In *Proceedings of the 2013 conference on empirical methods in natural language processing(EMNLP)*, pages 1533–1544.

Hamed Bonab, Mohammad Aliannejadi, John Foley, and James Allan. 2019. Incorporating hierarchical domain information to disambiguate very short queries. In *Proceedings of the 2019 ACM SIGIR International Conference on Theory of Information Retrieval*, pages 51–54.

Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. 1993. Signature verification using a" siamese" time delay neural network. *Advances in neural information processing systems (NeurIPS)*, 6:737–744.

Wei-Cheng Chang, Felix X Yu, Yin-Wen Chang, Yiming Yang, and Sanjiv Kumar. 2020. Pre-training tasks for embedding-based large-scale retrieval. In *International Conference on Learning Representations (ICLR)*.

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

Hang Cui, Ji-Rong Wen, and Tat-Seng Chua. 2003. Hierarchical indexing and flexible element retrieval for structured document. In *European Conference on Information Retrieval*, pages 73–87. Springer.

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

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Realm: Retrieval-augmented language model pre-training. *International Conference on Machine Learning(ICML)*.

Gautier Izacard and Edouard Grave. 2020. Distilling knowledge from reader to retriever for question answering. In *International Conference on Learning Representations (ICLR)*.

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(EACL)*.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*.

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

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

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 (TACL)*, 7:453–466.

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

Robert Levinson and Gerard Ellis. 1992. Multilevel hierarchical retrieval. *Knowledge-Based Systems*, 5(3):233–244.

Patrick Lewis, Ethan Perez, Aleksandara Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems (NeurIPS)*.

Ye Liu, Shaika Chowdhury, Chenwei Zhang, Cornelia Caragea, and Philip S Yu. 2020a. Interpretable multi-step reasoning with knowledge extraction on complex healthcare question answering. *arXiv preprint arXiv:2008.02434*.

Ye Liu, Yao Wan, Lifang He, Hao Peng, and Philip S Yu. 2020b. Kg-bart: Knowledge graph-augmented bart for generative commonsense reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Jing Lu, Gustavo Hernandez Abrego, Ji Ma, Jianmo Ni, and Yinfei Yang. 2020. Neural passage retrieval with improved negative contrast. *arXiv preprint arXiv:2010.12523*.

Ji Ma, Ivan Korotkov, Yinfei Yang, Keith Hall, and Ryan McDonald. 2021. Zero-shot neural passage retrieval via domain-targeted synthetic question generation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics(EACL)*, pages 1075–1088.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2017. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*.

Yuning Mao, Pengcheng He, Xiaodong Liu, Yelong Shen, Jianfeng Gao, Jiawei Han, and Weizhu Chen. 2020. Generation-augmented retrieval for open-domain question answering. *arXiv preprint arXiv:2009.08553*.

Yixin Nie, Songhe Wang, and Mohit Bansal. 2019. Revealing the importance of semantic retrieval for machine reading at scale. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

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

Devendra Singh Sachan, Mostofa Patwary, Mohammad Shoeybi, Neel Kant, Wei Ping, William L Hamilton, and Bryan Catanzaro. 2021. End-to-end training of neural retrievers for open-domain question answering. *arXiv preprint arXiv:2101.00408*.

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul Bennett, Junaid Ahmed, and Arnold Overwijk. 2020a. Approximate nearest neighbor negative contrastive learning for dense text retrieval. In *International Conference on Learning Representations (ICLR)*.

Wenhan Xiong, Xiang Lorraine Li, Srini Iyer, Jingfei Du, Patrick Lewis, William Yang Wang, Yashar Mehdad, Wen-tau Yih, Sebastian Riedel, Douwe Kiela, et al. 2020b. Answering complex open-domain questions with multi-hop dense retrieval. In *International Conference on Learning Representations (ICLR)*.

Wenhan Xiong, Hong Wang, and William Yang Wang. 2020c. Progressively pretrained dense corpus index for open-domain question answering. In *Conference of the European Chapter of the Association for Computational Linguistics (EACL)*.

A DHR Implementation Detail

Our document-level and passage-level retrievers use the base version of BERT as the pre-trained encoder. In the document-level retriever, we use the fixed token length 512 for document input and in the passage-level retriever, we use the fixed token length 280 for passage input. And the length of the question for both retrievers is 80. Our model is trained using the in-batch negative setting with a batch size of 128. We trained the document-level retriever and passage-level retriever for up to 40 epochs for large datasets (NQ, TriviaQA) and 100 epochs for small datasets (WebQ, TREC), with a learning rate of 10^{-5} using Adam, linear scheduling with warm-up and dropout rate 0.1. All the experiments are implemented on 8 A100 GPUs.

B Case study of DHR

We compare the Top-1 retrieved passages and their corresponding documents from DHR and DPR. Table 9 shows an example that DHR retrieves the gold passages but DPR fails. The question asks for the person proposing the first DNA accurate model. The passage retrieved by DPR is under the section of History in the document DNA, which is relevant to the question but it doesn't contain the answer. The reason is mainly that the question asks for a DNA model rather than DNA. In contrast, the retrieved passage by DHR is under the topic of the DNA model and it contains the answer. It's hard for the dense retriever to retrieve the correct passage directly since the passage under DNA, history is so related to the question. But own to the help of the Document-level retrieval in our hierarchical retriever framework, it's easy to discover that the document DNA sequencing is much more related than the document DNA to the question.

C End-to-End QA

C.1 Extractive Reader Implementation

For the implementation of the extractive reader, we sample 1 positive and 24 negative passages from the top 100 retrieved passages for each question. The training objective is to maximize the marginal log-likelihood of all the correct answer spans in the positive passage, combined with the log-likelihood of the positive passage being selected. We use the batch size of 16 for large datasets (NQ and TriviaQA) with a maximum of 40 epochs for large and 4 for small (WebQ and TREC) datasets with a maximum of 100 epochs. And we evaluate the development set at every 1000 steps.

C.2 QA results with the Generative Reader

We implement our retrieval results on NQ test set with the Fusion-in-Decoder model (FiD) (Izacard and Grave, 2021), a generative reader using pre-trained sequence-to-sequence model T5 (Raffel et al., 2019). The model takes the question, retrieved passages as input, and generates the answer. More precisely, each retrieved passage and its passage title are concatenated with the question and processed independently from other passages by the encoder. And the decoder calculates the attention over the concatenation of the joint representations of all the retrieved passages.

We use top-50 retrieved passages for both training and inference, while T5-base is used as the underlying architecture. We train the model for 10 epochs with a batch size of 64 and a learning rate of $1e-4$. We evaluate the model on the development set at every 500 steps, and select the checkpoint obtaining the highest EM score as the final model, and report its results on the NQ test.

From the Table 10, we can see that our proposed model DHR outperforms the DPR results in both first and second iteration, even with the less retrieved passages (FiD implementation uses top-100 retrieved passages), which shows the better retrieval results lead to the better generative answering results. And the generative greatly outperforms the extractive approach in Section 5.5.

Question	**Who proposed the first accurate model of DNA?** **Answer: James Watson**	
	Retrieved Passage	**Document of that Passage**
DPR	**Title:** DNA **DNA** was **first** isolated by the Swiss physician Friedrich Miescher who, in 1869, discovered a microscopic substance in the pus of discarded surgical bandages. As it resided in the nuclei of cells, he called it "nuclein". In 1878, Albrecht Kossel isolated the non-protein component of "nuclein", nucleic acid, and later isolated its five primary nucleobases. In 1909, Phoebus Levene identified the base, sugar, and phosphate nucleotide unit of the RNA (then named "yeast nucleic acid"). In 1929, Levene identified deoxyribose sugar in "thymus nucleic acid" (**DNA**). **Title list:** History	**Title**: DNA Deoxyribonucleic acid (; **DNA**) is a molecule composed of two chains that coil around each other to form a double helix carrying the genetic instructions used in the growth, development, ... The nitrogenous bases of the two of the two separate polynucleotide strands are bound together, according to base pairing rules, with hydrogen bonds to make double-stranded **DNA**. **Title list:** Properties, Nucleobase classification, Non-canonical bases, Listing of non canonical bases found in DNA, Base pairing, Sense and antisense, Supercoiling,...
DHR	**Title:** DNA sequencing Deoxyribonucleic acid (**DNA**) was **first** discovered and isolated by Friedrich Miescher in 1869, but it remained understudied for many decades because proteins, rather than **DNA**, were thought to hold the genetic blueprint to life .. This was the **first** time that **DNA** was shown capable of transforming the properties of cells. In 1953, James Watson and Francis Crick put forward their double-helix **model** of **DNA**, based on crystallized X-ray structures being studied by Rosalind Franklin 2013. **Title list:** History, Discovery of DNA structure and function	**Title:** DNA sequencing **DNA** sequencing is the process of determining the order of nucleotides in **DNA**. ...The advent of rapid **DNA** sequencing methods has greatly accelerated biological and medical research and discovery. ... The **first DNA** sequences were obtained in the early 1970s by academic researchers using laborious methods based on two-dimensional chromatography. **Title list:** Applications, ... History, Discovery of DNA structure and function, RNA sequencing, Early DNA sequencing methods, Sequencing of full genomes,..., Basic methods ...

Table 9: An example of passages returned by DPR and DHR and their corresponding document abstract. The words in bold means it appears in the question and the correct answers are written in red.

	EM	F1
1-iter		
DPR	48.20	
DPR*	48.72	56.64
DHR w/o Title	**50.63**	**58.74**
DHR w Title	49.86	57.97
2-iter		
DPR*	48.55	56.34
DHR w/o Title	**50.33**	**58.28**
DHR w Title	50.27	58.20

Table 10: End-to-end QA evaluation results on NQ test set using Fusion-in-Decoder model (Izacard and Grave, 2021). * represents reproducing results on our processed Wikipedia data.