End-to-End Training of Neural Retrievers for Open-Domain Question Answering

Devendra Singh Sachan[1,2]*, **Mostofa Patwary**[3], **Mohammad Shoeybi**[3], **Neel Kant**[3], **Wei Ping**[3], **William L Hamilton**[1,2,4], **Bryan Catanzaro**[3]
[1]Mila - Quebec AI Institute; [2]McGill University; [3]NVIDIA; [4]Canada CIFAR AI Chair
sachande@mila.quebec, mpatwary@nvidia.com

Abstract

Recent work on training neural retrievers for open-domain question answering (OpenQA) has employed both supervised and unsupervised approaches. However, it remains unclear how unsupervised and supervised methods can be used most effectively for neural retrievers. In this work, we systematically study retriever pre-training. We first propose an approach of unsupervised pre-training with the Inverse Cloze Task and masked salient spans, followed by supervised finetuning using question-context pairs. This approach leads to absolute gains of 2+ points over the previous best result in the top-20 retrieval accuracy on Natural Questions and TriviaQA datasets. We next explore two approaches for end-to-end training of the reader and retriever components in OpenQA models, which differ in the manner the reader ingests the retrieved documents. Our experiments demonstrate the effectiveness of these approaches as we obtain state-of-the-art results. On the Natural Questions dataset, we obtain a top-20 retrieval accuracy of 84%, an improvement of 5 points over the recent DPR model. We also showcase good results on answer extraction, outperforming recent models such as REALM and RAG by 3+ points. Our code is available at: https://github.com/NVIDIA/Megatron-LM.

1 Introduction

The task of open-domain question answering (OpenQA) consists of finding *answers* to the information-seeking *questions* using a large knowledge source such as Wikipedia. This knowledge source is also referred to as *evidence* and it typically contains millions of documents. Most approaches for OpenQA consist of a two-stage pipeline (Chen et al., 2017; Chen, 2018). In the first stage, given a question, a *retriever* module identifies the most relevant documents, which is often a very small subset of the evidence known as *context documents*. Traditionally, approaches based on document ranking such as BM25 (Robertson and Zaragoza, 2009) have been used for the retriever. In the second stage, these relevant documents are given as input to the *reader* module, which understands them and extracts the answer for the question (Figure 1).

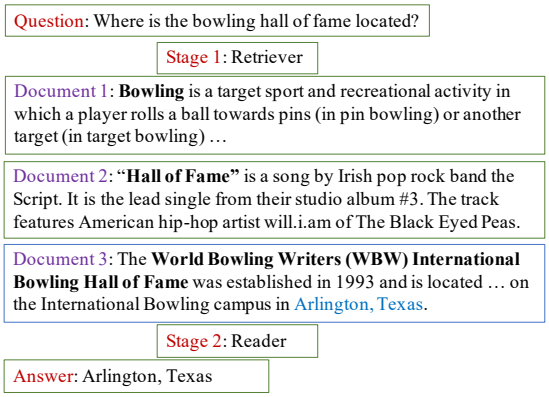

Figure 1: An example illustrating OpenQA pipeline.

The main drawback of the BM25 method is that it is not trainable and hence it can't be adapted to tasks involving open-retrieval. Recent work has addressed this limitation by building upon advances in self-supervised learning, such as BERT (Devlin et al., 2019). These approaches model both the retriever and reader using neural networks, allowing the retriever to be trained using task-specific datasets (Lee et al., 2019; Guu et al., 2020). Typically, the retriever model consists of a *dual-encoder* architecture (Bromley et al., 1994), where one encoder processes the question and the other encoder processes the context document. Prior work has investigated both unsupervised and supervised approaches to train the retriever. Unsupervised approaches include separately training the retriever with Inverse Cloze Task (ICT) (Lee et al., 2019) or training the retriever and reader jointly by pre-

*This work was done during an internship at NVIDIA. Corresponding authors: Devendra Sachan, Mostofa Patwary.

Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, pages 6648–6662
August 1–6, 2021.

dicting masked salient spans (REALM) (Guu et al., 2020), while supervised approaches such as Dense Passage Retrieval (DPR) (Karpukhin et al., 2020) train the retriever using human-annotated sets of question and context pairs.

However, there is no study that investigates the comparative advantages of using these two styles of training when the retrieval task is challenging, *i.e.*, when the evidence contains millions of documents. It is unclear if the unsupervised approaches can further help to improve the performance of *strong* supervised approaches, and, if so, under what conditions. A core focus of this work is systematically studying these aspects of retriever training.

We propose a unified approach to train the retriever: unsupervised pre-training followed by supervised finetuning. We also investigate key design choices—such as relevance score scaling and longer training—and showcase their effectiveness. Our results demonstrate that the proposed approach obtains substantial accuracy gains when evaluated on benchmark OpenQA datasets. Extensive experiments also highlight the relative importance of different pre-training strategies, revealing important trade-offs when varying the amount of supervised data available to train the retriever.

Furthermore, motivated by recent work (Guu et al., 2020; Lewis et al., 2020a), we also explore two approaches for end-to-end supervised training of the reader and retriever components. In the first approach, the reader considers each retrieved document separately while in the second approach, the reader takes as input all the retrieved documents together. We compare the effectiveness of these approaches on both retrieval accuracy and answer extraction. We show that the first approach leads to an improved retrieval performance, while the second approach results in an improved answer extraction. With end-to-end training, we outperform previous best models to obtain new state-of-the-art results on retrieval accuracy and answer extraction. We also perform experiments by scaling the model size to a large configuration for both retriever and reader and observe consistent improvements, compared with smaller models.

In summary, the contributions of this work are:

- We demonstrate that our proposed method of *unsupervised pre-training* of the retriever with ICT followed by *supervised finetuning* leads to absolute gains of more than 2 points in the top-20 retrieval accuracy over the previous best result on Natural Questions and TriviaQA datasets.
- We show that *masked salient spans*-based pre-training of the retriever is more effective when the supervised dataset sizes are small.
- Our *end-to-end* training approach obtains new state-of-the-art performance on retrieval accuracy. On Natural Questions, our top-20 accuracy is 84, which is a 5 points gain over DPR results.
- We achieve competitive results on *answer extraction* with gains of more than 3 points over recent models such as REALM (Guu et al., 2020) and RAG (Lewis et al., 2020c).
- We *scale up* end-to-end training to *large models* and show *consistent gains* in performance.

The rest of the paper is organized as follows. Sec. 2 and 3 explain the retriever model and end-to-end training, respectively. Sec. 4-6 describe the experimental details with the results. Sec. 7 reviews the related work followed by conclusion in Sec. 8.

2 Neural Retriever

In this section, we first describe the retriever architecture and then discuss different approaches to train it, including our proposed approach.

2.1 Background

Given a collection of documents in the evidence $\mathcal{Z} = \{z_1, \cdots, z_m\}$ and a question q, the task of the retriever is to select a relevant subset of documents for the question. To do this, the retriever performs a ranking of the evidence documents conditioned on the question and outputs the top-ranked documents.

The retriever model consists of two modules: a question encoder (f_Q) and a context encoder (f_Z). Such a model is often referred to as a *dual-encoder model* (Bromley et al., 1994). Here, we detail the training methodology of the dual-encoder model given a questions (q) and context documents (z_i) from $\mathcal{Z}$. First, we compute the *relevance score* between the question and context. We define the relevance score to be the dot-product between the question and context representations

$$s(q, z_i; \phi) = f_Q(q)^\top f_Z(z_i) \tag{1}$$

where $f_Q(q) \in \mathbb{R}^d$ and $f_Z(z) \in \mathbb{R}^d$ denote the question and context encoders, respectively, which are parameterized by $\phi = [\phi_Q, \phi_Z]$. We model the f_Q and f_Z using BERT-style transformer networks (Devlin et al., 2019; Vaswani et al., 2017). We consider the hidden states of the first token of

the sequence (i.e. [CLS] token) as the encoder's output. The probability of a context document z_i being relevant to the question q is calculated as

$$p(z_i \mid q, \mathcal{Z}; \phi) = \frac{\exp(s(q, z_i; \phi)/\tau)}{\sum_{j=1}^{|\mathcal{Z}|} \exp(s(q, z_j; \phi)/\tau)} \quad (2)$$

where τ is a scaling factor. While previous work had used the setting of $\tau = 1$, in this work, we set $\tau = \sqrt{d}$. Bigger scaling factor helps in better optimization when the model hidden size (d) is large. We refer to this as *relevance score scaling*. To train the retriever, we maximize the log-likelihood computed from Eq. 2.

In practice, as the evidence set consists of millions of documents, the normalization term would be expensive to compute. Hence, we approximate the denominator of the above equation by using the context documents in the batch as negative examples, a technique that has shown to perform well in practice (Chen et al., 2020).

2.2 Training

In this section, we discuss different approaches to train the retriever. In all the approaches, we initialize the parameters of both the question and context encoders using BERT weights as implemented in Megatron-LM (Shoeybi et al., 2019). We also experimented with random initialization but it vastly underperformed BERT initialization.

2.2.1 Supervised Training

In the supervised setting, *human-annotated* questions, answers, and sometimes context are provided. If the context is not included, then a common approach is to use distant supervision (Mintz et al., 2009) to obtain the context document. Specifically, we select the top-ranked document using BM25 (Robertson and Zaragoza, 2009) from the evidence that contains the answer as the context. We also select other top-ranked documents that do not contain the answer as additional *hard* negative examples. This approach to train neural retriever was popularized by (Karpukhin et al., 2020).

2.2.2 Unsupervised Training

Inverse Cloze Task (ICT): In this setup, we do not consider the human-annotated question-context pairs. Instead, the retriever is trained in an unsupervised manner. Specifically, a randomly sampled sentence from a paragraph is considered as the query while other sentences as the context. This approach was first proposed by (Lee et al., 2019).

Masked salient spans training: (Guu et al., 2020) showcased that the ICT initialized retriever can be further improved by training it with an objective where the reader predicts the masked salient spans such as named entities conditioned on the retrieved documents. In this work, we adopt the same approach. However, unlike (Guu et al., 2020) who use BERT for the reader, we use a generative language model based on T5 (Raffel et al., 2020).

2.3 Proposed Approach: Unsupervised Pre-training and Supervised Finetuning

To improve the retriever training, we propose the approach of unsupervised pre-training of the retriever followed by supervised finetuning. In this approach, we first pre-train the retriever weights with ICT training or masked salient spans training (Sec. 2.2.2). After pre-training, we finetune the retriever with supervised training (Sec. 2.2.1).

3 End-to-End Retriever and Reader Training

In this section, we explore two *supervised training* approaches to end-to-end train the reader and retriever components from the task-specific data. In the first approach, the reader considers each retrieved document separately (Sec. 3.1) while in the second approach, the reader takes as input all retrieved documents together (Sec. 3.2). These approaches are designed such that when predicting the answer conditioned on the question, the learning process improves both the reader and retriever.

Background and notation: In end-to-end training, the trainable components consists of the retriever (ϕ) and reader (θ) parameters. For retriever, we use the dual-encoder architecture and train it as discussed previously in Sec. 2.3. Our reader is a *generative model* designed according to the sequence-to-sequence modeling paradigm (Sutskever et al., 2014). Specifically, we use pre-trained T5 as the reader. The inputs to the training process are *questions* (q) and its *answers* (a), both in string form. Given a question, first the retriever obtains the *k relevant* context documents ($\mathcal{K}$) from the evidence ($\mathcal{Z}$) as

$$\mathcal{K} = \underset{z_i \in \mathcal{Z}}{\arg\text{sort}}\, s(q, z_i; \phi)[:k] \quad (3)$$

The reader then takes the question and one or more context documents (z_i) as input to predict the an-

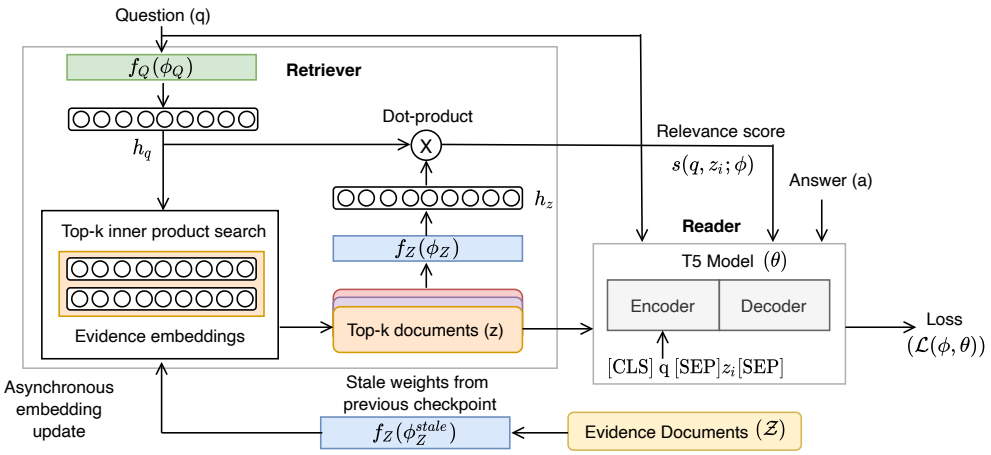

Figure 2: A schematic diagram illustrating end-to-end supervised training of the retriever and reader components.

swer, the likelihood of which is defined as

$$p(a \mid q, z_i; \theta) = \prod_{j=1}^{N} p\left(a_j \mid a_{1:j-1}, q, z_i; \theta\right), \quad (4)$$

where N is the number of answer tokens. Next, we describe the two proposed approaches. A block diagram illustrating the end-to-end training process is shown in Figure 2.

3.1 Approach 1: Individual Top-k

In this approach, similar to (Guu et al., 2020), the reader's likelihood is first computed conditioned on the question and each retrieved document. The marginal likelihood is defined as the weighted average of the *individual* likelihoods as

$$p(a \mid q; \theta, \phi) = \sum_{z_i \in \mathcal{K}} p(a \mid q, z_i; \theta) p(z_i \mid q, \mathcal{Z}; \phi), \quad (5)$$

where $p(z_i \mid q, \mathcal{Z}; \phi)$ is computed using Eq. 2. However, the normalization is done over $\mathcal{K}$ instead of $\mathcal{Z}$. The final loss is defined as the negative marginal log-likelihood

$$\mathcal{L}(q, a) = -\log p(a \mid q; \theta, \phi). \quad (6)$$

We note that the RAG model (Lewis et al., 2020c) also proposed a similar approach, but there are two main differences. The first is that while we update all the parameters of the retriever (both the query and context encoders), RAG just updates the query encoder. The second is that we use T5 model as the reader while RAG uses BART model (Lewis et al., 2020b). These enhancements help us obtain substantial gains over the RAG model, which we will discuss in Sec. 6.

3.2 Approach 2: Joint Top-k

In this approach, similar to (Lewis et al., 2020a), the likelihood is defined as the reader's likelihood conditioned on the question, *all* the retrieved documents, and the retrieval score

$$p(a \mid q; \theta, \phi) = p(a \mid q, z_{1:k}, p(z \mid q, \mathcal{Z}; \phi); \theta). \quad (7)$$

As the T5 reader consists of separate encoder and decoder modules, it provides the flexibility to customize the input or output of the encoder. We concatenate each retrieved document with the question and feed them as input to the encoder, which computes their hidden representations. Next, we stack the hidden representations of all the retrieved documents, which the decoder *jointly* attends to during the encoder-decoder attention, thus allowing a more powerful form of information aggregation from multiple retrieved documents. We also add retriever similarity score to bias the encoder-decoder attention as it helps facilitate end-to-end training and enables the reader to pay higher attention to the relevant documents. The interaction score during the encoder-decoder attention is computed as

$$\text{attn}(q, a, z_{1:k}) \propto Q(a)^\top K(z_{1:k}, q) + \lambda p(z \mid q; \phi), \quad (8)$$

where Q is the query vector computed from decoder's input, K is the key vector computed from encoder's output, and λ is a trainable parameter.

Final loss is defined according to Eq. 6. We further note that a similar approach for OpenQA was proposed in (Izacard and Grave, 2020) but it only optimizes the reader model and didn't perform end-to-end training of the retriever.

Dataset	Train	Filtered Train	Dev	Test
NQ	79,168	58,880	8,757	3,610
TriviaQA	78,785	60,413	8,837	11,313

Table 1: OpenQA dataset statistics. The training set is used for end-to-end training, while the filtered version is used for retriever training. The filtered set ignores those examples where the document retrieved from evidence does not align with the ground-truth document.

4 Experimental Setup

In this section, we describe the datasets and model settings. For reproducibility, we provide training details and list the hyperparameters in Appendix A.

4.1 OpenQA Datasets

We perform experiments using two widely used QA datasets whose details are provided below and their statistics are shown in Table 1.

Natural Questions (NQ): This corpus consists of real questions asked from the Google search engine along with their long and short answer annotations from the top-ranked Wikipedia pages (Kwiatkowski et al., 2019). Following prior work (Karpukhin et al., 2020), we use the same subset of the short answer questions in our experiments, as it is more suited for OpenQA.

TriviaQA: This corpus consists of a collection of trivia questions and their answers scraped from multiple sources in the Web (Joshi et al., 2017).

Evidence: Following (Karpukhin et al., 2020), we make use of their released preprocessed English Wikipedia dump from December 2018 as the source of evidence documents. Overall, there are $21,015,324$ documents, each 100 words long.

4.2 Model Details

We use two models of different sizes, *base* and *large*, for the experiments. The base configuration consists of 12 layers, 768-d hidden size, and 12 attention heads. The BERT-base contains 110M parameters while the T5-base contains 220M parameters. The large configuration consists of 24 layers, 1024-d hidden size, and 16 attention heads. The BERT-large contains 330M parameters while the T5-large contains 770M parameters.

5 Results: Retriever Training

In this section, we compare different approaches to train the retriever. Retrieval accuracy is evaluated using the top-k metric ($k \in \{1, 5, 20, 100\}$).

Setting	Top-1	Top-5	Top-20	Top-100
Base Configuration				
[CLS], 40 epochs	32.6	60.1	76.4	85.9
+ score scaling	34.1	60.9	77.6	85.9
+ 80 epochs	36.7	62.2	77.4	**86.0**
+ 1 hard negative	**48.6**	**74.5**	**79.0**	85.8
DPR (Official)	–	67.1	78.4	85.4

Table 2: Effect of different factors on the supervised training of retriever when evaluated on NQ test set.

5.1 Effect of Relevance Score Scaling, Longer Training, and Hard Negatives

We explore the best training settings for *supervised* training of the retriever. To do so, we perform a series of experiments on the NQ dataset starting with the training settings from the popular DPR model and then progressively improve it. DPR was initialized with BERT, trained for 40 epochs, with a scaling factor of 1, and utilized [CLS] token embeddings from the retriever. Our result with this setting is shown in Table 2. We then observe that incorporating relevance score scaling and longer training till 80 epochs helps to improve the top-5 and top-20 accuracy by 1.5-2 points. These results also signify that the original DPR model was significantly undertrained and not fully optimized.

In addition to score scaling, we further include 1 additional hard-negative example (similar to DPR) for each question-context pair and train the model for 80 epochs. Our results, in sync with the results of DPR, obtain substantial additional gains in performance. *These findings highlight that relevance score scaling, longer training, and including a hard negative example are essential to improve the supervised retriever's accuracy*. These supervised training results can be considered as a very strong baseline. Hence, we employ these settings in subsequent experiments.

5.2 Effect of Retriever Initialization

We first characterize the zero-shot retriever's performance when its weights are initialized with either BERT or ICT or masked salient spans pre-training (Table 3). As is understood that unsupervised language models do not perform well in information retrieval tasks (Lee et al., 2019), evidently, BERT also leads to a poor retrieval accuracy. We note that ICT initialization is quite effective in providing a non-trivial zero-shot accuracy which is further improved by masked salient spans training by more than 8 points. Both being unsupervised approaches

Model	NQ				TriviaQA			
	Top-1	Top-5	Top-20	Top-100	Top-1	Top-5	Top-20	Top-100
Base Configuration								
BERT (zero-shot)	0.9	3.9	9.4	20.3	0.6	2.8	7.2	17.8
ICT (zero-shot)	12.6	32.3	50.6	66.8	19.2	40.2	57.5	73.6
Masked salient spans (zero-shot)	20.0	41.7	59.8	74.9	31.7	53.3	68.2	79.4
BERT + Supervised	48.6	68.8	79.0	85.8	57.5	72.2	80.0	85.1
ICT + Supervised	48.4	**72.1**	81.8	**88.0**	58.4	73.9	81.7	86.3
Masked salient spans + Supervised	**50.3**	71.9	**82.1**	87.8	**60.6**	**74.8**	**81.8**	**86.6**
Large Configuration								
ICT (zero-shot)	13.0	31.8	49.3	66.1	20.1	41.6	58.5	74.1
BERT + Supervised	51.4	71.0	81.0	87.2	60.4	74.5	81.4	86.0
ICT + Supervised	**52.4**	**72.7**	**82.6**	**88.3**	**61.9**	**76.2**	**82.9**	**87.1**

Table 3: Effect of unsupervised pre-training on retrieval accuracy when evaluated on NQ and TriviaQA test sets.

demonstrate their utility in effectively bootstrapping the retriever almost from scratch.

We next empirically analyze our proposed approach of pre-training with ICT and masked salient spans followed by supervised finetuning. We observe that it provides absolute improvements of 2-3 points over the already strong supervised training results, with the gains being consistent across both the datasets. These results highlight that even after finetuning the retriever with thousands of labeled examples, it does not lead to catastrophic forgetting of the discriminative properties learned by the retriever during ICT and masked salient spans pre-training. Another merit is that being unsupervised, large text collections can be leveraged to pre-train the retriever, a considerable advantage over data-augmentation methods which rely on the availability of human-annotated question-context pairs. *Furthermore, when comparing ICT with masked salient spans initialization, we note that their accuracy gains are roughly similar.*

5.3 Effect of Amount of Training Data

We study the effect on accuracy when the retriever is pre-trained with BERT, ICT, or masked salient spans and the amount of supervised training data is varied. We train the retriever with 1%, 2%, 5%, 10-50%, of NQ's training data and plot the top-20 accuracy in Figure 3. Results reveal that in the low-resource regime, *masked salient spans pre-training is much more effective than ICT, consistently leading to large gains*. As the fraction of training data increases to beyond 40% towards a high-resource setup, *the gains from salient spans pre-training saturates to that of ICT*. We believe that these findings will have important implications for future research in OpenQA—with only a few hundred examples, performing expensive masked salient span training is beneficial while if the training data has thousands of examples, ICT is just as optimal as masked salient spans training.

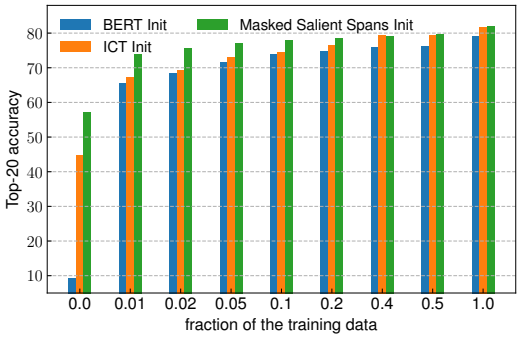

Figure 3: Effect of amount of training data on retrieval accuracy when evaluated on NQ test set.

5.4 Effect of End-to-End Training

For end-to-end training, retriever weights are initialized with the previous best setting of ICT pre-training and supervised finetuning. The number of retrieved evidence documents for the reader is considered as a hyperparameter and is selected via performance on the dev set. The focus here is to analyze the effect on retrieval accuracy when updating the retriever weights using question-answer pairs in an end-to-end setting (Sec. 3). From the results in Table 4, we observe that for *Individual Top-k*, when only the query encoder is updated, it tends to improve retrieval accuracy. In addition, when the context encoder is also updated, the retrieval accuracy improves to 75% at top-5, a big gain of 8 points over the previous best DPR retriever. Larger models further help to improve the performance leading to new state-of-the-art results.

On the other hand, in *Joint Top-k*, updating the

Model	Q	C	NQ Top-1	NQ Top-5	NQ Top-20	NQ Top-100	TriviaQA Top-1	TriviaQA Top-5	TriviaQA Top-20	TriviaQA Top-100
Base Configuration										
DPR (Karpukhin et al., 2020)			–	67.1	78.4	85.4	–	–	79.4	85.0
ICT + Supervised			48.4	72.1	81.8	88.0	58.4	73.9	81.7	86.3
Individual Top-k	✓	✗	54.5	73.7	83.2	88.6	61.4	75.6	82.1	86.7
Individual Top-k	✓	✓	**56.8**	**75.0**	**84.0**	**89.2**	**63.5**	**76.8**	**83.1**	**87.0**
Joint Top-k	✓	✗	51.1	72.1	81.8	87.8	59.1	74.1	81.3	86.3
Large Configuration										
ICT + Supervised			52.4	72.7	82.6	88.3	61.9	76.2	82.9	87.1
Individual Top-k	✓	✓	**57.5**	**76.2**	**84.8**	**89.8**	**66.4**	**78.7**	**84.1**	**87.8**
Joint Top-k	✓	✗	53.7	73.3	83.2	88.0	61.2	75.9	82.7	87.0

Table 4: Effect of end-to-end training using question-answer pairs on retrieval accuracy. **Q** and **C** signify if the query encoder and the context encoder are updated during training or not, respectively.

$\times\sqrt{d}$	Top-1	Top-5	Top-20	Top-100	Avg.
Base Configuration					
0.25	48.8	69.3	78.7	85.5	70.6
0.5	51.4	71.6	81.5	87.7	73.1
1	51.1	71.8	82.1	87.7	**73.2**
2	50.2	71.5	81.9	87.9	72.9
4	50.6	71.7	81.7	88.0	73.0

Table 5: Effect of score scaling factor (τ) on the retrieval accuracy when evaluated on the NQ test set. The first column denotes the multiple (m) that is multiplied by $\sqrt{d}$ to obtain τ, i.e., $\tau = m \times \sqrt{d}$ in Equation 2.

query encoder just improves the top-1 score but does not really lead to much accuracy gains for higher top-k's. We also do not update the context encoder for *Joint Top-k* as it did not result in improvements during our initial experiments.

These results showcase that when the retriever is already well-initialized, the objective function of *Individual Top-k* method is designed such that it significantly improves the retrieval accuracy while the *Joint Top-k* method does not result in improvements. As we will show next, that the usefulness of this method lies in answer extraction.

5.5 Intuition for Retriever Score Scaling

Retrieval score scaling is used when computing the probability distribution of the retrieved documents according to Equation 2, where the retrieval score is normalized by the scaling factor (τ). To study the effect of τ on the retrieval accuracy, we perform an ablation study with different values of τ on the NQ retrieval task, whose results can be seen in Table 5. More specifically, we choose different values of τ as a multiple of $\sqrt{d}$, where d is the hidden size of the model. Our results indicate that the choice of $\tau = \sqrt{d}$ works well in practice.

Here, we briefly explain the intuition regarding the usage of the scaling factor. In our preliminary experiments on retriever training and end-to-end training without the scaling factor, we observed that a few of the top-k document's similarity score with the query was very high that in turn led to it being assigned a high retrieval probability score. This high score was leading to a skewed probability distribution with most of the mass being centered over the top-1 or top-2 retrieved documents. A larger value of scaling factor results in a more even distribution of probability mass over the top-k documents, which in turn leads to better results in both retrieval accuracy and in the end-to-end training.

6 Results: Answer Extraction

We next present the results of end-to-end training on answer extraction. To train the model, retriever weights are initialized with ICT pre-training and supervised finetuning while the reader is initialized with pre-trained T5 weights. The number of retrieved evidence documents for the reader is tuned on the dev set. Results are reported using the conventional Exact Match (EM) metric.

6.1 Individual Top-k Approach

We compare our results as presented in Table 6 with the recent related approaches in OpenQA. For the base configuration on NQ, our model outperforms both REALM and DPR by more than 4 points. For the large configuration, we compare with the RAG model (Lewis et al., 2020c), where our approach outperforms it by 3.5+ points on NQ and by 2.8 points on TriviaQA. *Our improved results are because of a more accurate initial retriever, stronger reader, and updating both the query and context encoders during training*.

Model	NQ	TriviaQA
Base Configuration		
ORQA (Lee et al., 2019)	33.3	45.0
REALM (Guu et al., 2020)	40.4	–
DPR (Karpukhin et al., 2020)	41.5	56.8
Individual Top-k	**45.9**	56.3
Large Configuration		
RAG (Lewis et al., 2020c)	44.5	56.8
Individual Top-k	**48.1**	**59.6**

Table 6: Answer extraction results using *Individual Top-k* approach. The grouping under base and large configurations is based on the size of the reader model.

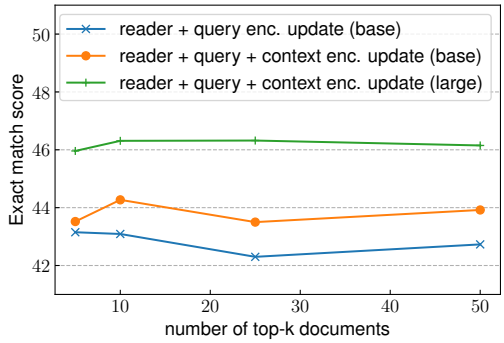

Figure 4: Effect of increasing top-k documents on answer generation for *Individual Top-k* approach.

Our analysis in Figure 4 reveals that updating the context encoder improves the results for both the base and large configurations. Quite surprisingly, we also observe that the performance of *Individual Top-k* approach is sensitive to the number of top-k documents and can also decrease with an increase in top-k documents. We leave an in-depth investigation of this as a future work.

6.2 Joint Top-k Approach

We compare our results with the recent Fusion-in-Decoder (FiD) approach (Izacard and Grave, 2020) that also performs joint encoder-decoder attention. It consists of DPR as the retriever and T5 as the reader, which are initialized with their open-source weights. However, unlike our approach, FiD just finetunes the reader weights. Our results in Table 7 show that for the base configuration, Joint Top-k outperforms the FiD model by 1 point on NQ, highlighting the significance of end-to-end training. For the large configuration, we obtain a gain of 0.7 points on TriviaQA.

Our analysis in Figure 5 portrays that the EM scores improve with more retrieved documents. This highlights that in contrast to *Individual Top-k*, the *Joint Top-k* better aggregates the information

Model	NQ	TriviaQA
Base Configuration		
FiD (Izacard and Grave, 2020)	48.2	65.0
Joint Top-k	**49.2**	64.8
Large Configuration		
FiD (Izacard and Grave, 2020)	51.4	67.6
Joint Top-k	**51.4**	**68.3**

Table 7: Results on answer extraction using *Joint Top-k* approach.

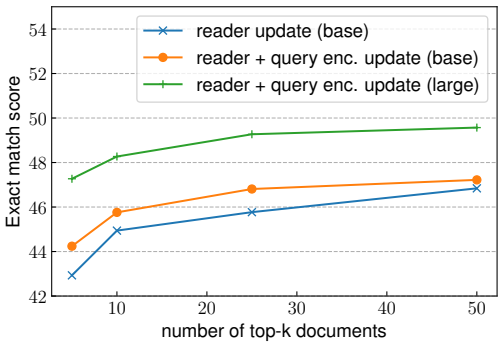

Figure 5: Effect of increasing top-k documents on answer generation for *Joint Top-k* approach.

contained in the retrieved documents. This Figure also illustrates the effect of similarity enriched attention on answer extraction for the base configuration. For values of top-k=5, 10, and 25, using retrieval-similarity enriched encoder-decoder attention, we consistently observe a gain of 0.8-1 EM points (comparing orange plot and blue plot in Figure 5), while there is a smaller gain when top-k=50. This signifies that with more retrieved documents, the utility of end-to-end training tends to diminish, thus explaining the lower gains observed in retrieval performance for *Joint Top-k* in Table 4.

6.3 Overall Comparison

Based on the discussions in Sec. 5.4 and Sec. 6, we remark that end-to-end training using the two approaches has a complementary effect on the retrieval accuracy and answer extraction. While the *Individual Top-k* approach helps to significantly improve the retrieval performance, the *Joint Top-k* approach is more useful for answer extraction.

7 Related Work

(Yih et al., 2011) proposed a discriminative approach to train a retriever by learning dense representations of query and context documents based on word frequency. However, this approach was data-hungry and not scalable. Recently, (Lee et al.,

2019; Karpukhin et al., 2020) address this by leveraging pre-trained BERT weights (Devlin et al., 2019) to train a dual-encoder retriever by using smaller amounts of question-context pairs. In particular, (Lee et al., 2019) first pre-train the retriever in an unsupervised manner using ICT and then jointly train the retriever and reader for OpenQA. On the other hand, (Karpukhin et al., 2020) perform supervised training of the retriever using hard-negative examples, yielding impressive results on several retrieval benchmarks.

To improve the retrieval accuracy of the dual-encoder model, (Chang et al., 2020) explore several paragraph-level pre-training strategies including the application of ICT. They demonstrated the effectiveness of pre-training over sparse-retrieval approaches such as BM25. Their evidence consisted of the training documents that was further increased to 1M documents for OpenQA. Our work differs from them in several ways. First, our OpenQA setup is more challenging as the evidence consists of 21M documents. Second, we pre-train with two strategies consisting of ICT and masked salient-spans and finetune using strong supervised methods, which leads to much improved results. Third, we further update the retriever with end-to-end training leveraging question-answer pairs, which further improves the retrieval accuracy leading to new state-of-the-art results.

A new line of work investigates task-specific pre-training of language models. For example, (Guu et al., 2020) predicts masked salient spans consisting of named entities to pre-train the reader and retriever components for OpenQA. Similarly, (Lewis et al., 2020a) perform cross-lingual pre-training where the objective is to predict a sequence using its paraphrases in different languages, demonstrating improved zero-shot performance in document translation tasks.

8 Conclusion

We propose approaches to improve the retrieval accuracy of the dual-encoder model for the OpenQA task. We first perform a systematic investigation of the importance of pre-training with ICT and masked salient spans tasks for supervised training of the retriever. We then present two approaches for end-to-end training of the reader and retriever components in OpenQA. In one approach, the reader considers each retrieved document individually while in the other approach where the reader considers all the retrieved documents jointly. Overall, these methods help achieve state-of-the-art results on both retrieval and answer extraction.

Acknowledgements

This work was done during the first author's internship at NVIDIA. It was also partially supported by Canada CIFAR AI Chair held by Prof. Hamilton. We would like to thank the anonymous reviewers for providing valuable feedback and recommendations. We would also like to thank the administrators of the Selene supercomputer for their assistance in facilitating the large-scale runs.

Broader Impact and Ethics Statement

To understand the ethical context of our work on open-domain question answering, it is important to consider the real-world use cases and potential individuals who may interact with systems developed based on our proposed methods. The potential real-world applications could be search engines or virtual assistants, where our techniques can improve the question-answering ability. However, it is worthwhile to mention that our trained systems can not be deployed off-the-shelf for such applications, given that our models were trained on the Natural Questions and TriviaQA datasets with the goal of matching the specific training data distribution. Real-world applications building on our work should be re-trained using a custom training dataset that is relevant to the kind of queries that originates in practice.

Our system represents a prototype model for answering questions over Wikipedia and can easily be extended to be used in sensitive contexts such as legal or health-care settings. However, extensive and robust quality assurance testing will be needed as our system was not designed to meet those criteria. More generally, there is the possibility of social biases which could be introduced by the training data. Since we did not control or regularize our model to remove such biases, we would urge the users to undertake the necessary quality-assurance testing to evaluate and understand the extent to which such biases might be present. User should also understand how much these biases are impacting their trained system and to make modifications to their training data and procedures accordingly.

References

Alexandr Andoni, Piotr Indyk, Thijs Laarhoven, Ilya Razenshteyn, and Ludwig Schmidt. 2015. Practical and optimal lsh for angular distance. In *Advances in Neural Information Processing Systems*.

Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. 1994. Signature verification using a "siamese" time delay neural network. In *Advances in Neural Information Processing Systems*.

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*.

Danqi Chen. 2018. *Neural Reading Comprehension and Beyond*. Ph.D. thesis, Stanford University.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading Wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. 2020. Retrieval augmented language model pre-training. In *Proceedings of the 37th International Conference on Machine Learning*.

Gautier Izacard and Edouard Grave. 2020. Leveraging passage retrieval with generative models for open domain question answering. *arXiv preprint arXiv:2007.01282*.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

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

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *The 2015 International Conference for Learning Representations*.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Matthew Kelcey, Jacob Devlin, Kenton Lee, Kristina N. Toutanova, Llion Jones, Ming-Wei Chang, Andrew Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association of Computational Linguistics*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*.

Mike Lewis, Marjan Ghazvininejad, Gargi Ghosh, Armen Aghajanyan, Sida Wang, and Luke Zettlemoyer. 2020a. Pre-training via paraphrasing. In *Advances in Neural Information Processing Systems*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020b. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*.

Patrick Lewis, Ethan Perez, Aleksandara Piktus, F. Petroni, V. Karpukhin, Naman Goyal, Heinrich Kuttler, M. Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020c. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Advances in Neural Information Processing Systems*.

Shen Li, Yanli Zhao, Rohan Varma, Omkar Salpekar, Pieter Noordhuis, Teng Li, Adam Paszke, Jeff Smith, Brian Vaughan, Pritam Damania, and Soumith Chintala. 2020. Pytorch distributed: Experiences on accelerating data parallel training. *Proc. VLDB Endow.*

Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, and Hao Wu. 2018. Mixed precision training. In *International Conference on Learning Representations*.

Mike Mintz, Steven Bills, Rion Snow, and Daniel Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the*

Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. CoNLL-2012 shared task: Modeling multilingual unrestricted coreference in OntoNotes. In *Joint Conference on EMNLP and CoNLL - Shared Task*.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A Python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*.

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

Stephen Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends in Information Retrieval*.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. 2019. Megatron-lm: Training multi-billion parameter language models using gpu model parallelism. *arXiv preprint arXiv:1909.08053*.

Anshumali Shrivastava and Ping Li. 2014. Asymmetric lsh (alsh) for sublinear time maximum inner product search (mips). In *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*.

Wen-tau Yih, Kristina Toutanova, John C. Platt, and Christopher Meek. 2011. Learning discriminative projections for text similarity measures. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning*.

A Training Details

We provide the training details of all the experiments below. We use the same training settings for both the base and large model configurations and use the open-source Megatron-LM toolkit (Shoeybi et al., 2019) to implement the models.[1] To train the models, we employed mixed-precision training (Micikevicius et al., 2018) and leveraged distributed training feature as implemented in the Pytorch framework (Li et al., 2020). All of our experiments were performed on the Selene cluster which consists of NVIDIA A100 GPUs.

A.1 Language Models Training

We train BERT (Devlin et al., 2019; Lan et al., 2020) and T5 (Raffel et al., 2020) language models from scratch, whose hyperparameters for both the base and large configurations are detailed in Table 8. We used 32 GPUs to train the BERT-large (330M) model and 128 GPUs to train the T5-large (770M) model.

A.2 Retriever Training

Supervised: We use Adam optimizer (Kingma and Ba, 2015), a batch size of 128, learning rate of 2e-5 with a linear decay, and train for 80 epochs. Training was performed on 16 GPUs.

ICT training: We initialize the parameters of both the question and context encoders using BERT weights trained with Megatron-LM. We train the model on Wikipedia paragraphs with maximum length of 256 tokens. We use a batch size of $4,096$, learning rate of 1e-4 with linear decay, and train the model for $100,000$ steps using Adam optimizer. This corresponds to training the model for roughly 20 epochs over the Wikipedia dataset. We set the weight decay to 0.01 and the warmup ratio of the optimizer to 0.01. With a probability of 0.1, we also keep the query sentence in the context. We train the large ICT model using 128 GPUs.

Masked salient spans generative training: We initialize the retriever with ICT training and pretrain the T5 reader on an aggregated dataset from (Shoeybi et al., 2019). We use the pre-trained models provided by the Stanza toolkit (Qi et al., 2020) to segment Wikipedia paragraphs into sentences and extract named entities.[2] The masked sentence is used as a query to retrieve evidence documents with the help of which the reader predicts the masked words. The model is trained according to Equation 5 and 6. We train the model for $100,000$ steps with Adam optimizer using a learning rate of 2e-5 and a warmup ratio of 0.05. Similar to (Guu et al., 2020), we also compute the evidence embeddings asynchronously and update the evidence index every 500 steps. Training was performed on 240 GPUs.

A.3 End-to-End Supervised Training

As the performance of the ICT pre-trained retriever and masked salient spans pre-trained retriever is similar when all the training data is used (Sec. 5.2), we select the retriever pre-trained with ICT initialization and finetuned with supervised data. For the reader, we use a pre-trained T5 model. For all experiments, we train for 10 epochs using a batch size of 64, learning rate of 2e-5 with linear decay, and weight regularization of 0.1. For *Individual Top-k* approach, during training, the evidence embeddings index is refreshed after every 500 steps. The number of retrieved evidence documents for the reader is considered as a hyperparameter and is selected via performance on the dev set. Training of Individual Top-k was performed on 240 GPUs while training of Joint Top-k was performed on 64 GPUs.

For retrieving the top-k documents from our evidence ($\sim$21M documents), we perform exact search. Specifically, we utilize matrix multiplication and top-k functionalities as provided by the PyTorch framework. This matrix multiplication operation is highly optimized for GPU computations and we observed that performing exact search was not a bottleneck during training. We therefore did not optimize or approximate the similarity search using LSH (Andoni et al., 2015) or efficient maximum inner product search (Shrivastava and Li, 2014).

NQ and TriviaQA Specific Details: For both datasets, we uniformly sample the target answer from the list of provided answers during the training process. For answer extraction, similar to (Guu et al., 2020), we did not append the title of the Wikipedia article with the corresponding top-k retrieved document as the reader's input.

[1] https://github.com/NVIDIA/Megatron-LM

[2] We use the model trained on OntoNotes (Pradhan et al., 2012) to extract named entities for 10 selected categories.

Hyperparameter	BERT	T5
Dataset	Wikipedia, BookCorpus	Wikipedia, CC-Stories, RealNews, OpenWebText
Hidden Size	{768, 1024}	{768, 1024}
Attention Heads	{12, 16}	{12, 16}
Dropout	0.1	0.1
Attention Dropout	0.1	0.1
Optimizer	Adam	Adam
Training Steps	1M	1M
Warmup Steps	10k	10k
Peak Learning Rate	1e-4	1e-4
Weight Decay	1e-2	1e-2
Batch Size	256	2048
Learning Rate Decay	Linear	Linear
Gradient Clipping	1.0	1.0

Table 8: Hyperparameters for pre-training BERT and T5 models.

A.4 Individual Top-k Inference

During inference, the reader model first greedily generates an answer for each retrieved document. We then score each generated answer using Eq. 5 and finally select the answer with the highest likelihood score.

A.5 Example Outputs from Retriever

We present few examples in Table 9 when the ICT + Supervised retriever is evaluated on the NQ test dataset.

B Reproducibility Checklist

B.1 For all reported experimental results

- *A clear description of the mathematical setting, algorithm, and/or model*: This is provided in the main paper in Sec. 2 and Sec. 3.

- *A link to a downloadable source code, with specification of all dependencies, including external libraries (recommended for camera ready, though welcome for initial submission)*: As mentioned previously, we have developed our codebase over the open-source Megatron-LM library (https://github.com/NVIDIA/Megatron-LM). Our implementations over this codebase are currently organized in different branches, that are better suited for walk-through with a git-based tool. To preserve anonymity and in good faith, we are submitting the source codes from one branch of our codebase, with the caution that the codebase doesn't contain an exhaustive README file.

- *A description of computing infrastructure used*: We run experiments on Nvidia's Selene cluster where each node's specifications are: Number of CPUs: 256, Physical Memory: 2.2TB, GPU model: 8 x Nvidia A100, GPU architecture and memory: Ampere/80GB, Arch: x86_64, and Disk size: 10TB.

- *The average runtime for each model or algorithm, or estimated energy cost*: We provide the average runtime and compute used for training different models in Appendix A. However, we want to highlight that our codes were not carefully optimized to minimize runtime or to make optimal use of the hardware resources.

- *The number of parameters in each model*: We provide number of parameters in models in Sec. 4.2.

- *Corresponding validation performance for each reported test result*: Validation set performance is currently not reported in the main paper. However, we followed rigorous experimentation protocol, and selected the best models by its performance on the validation set. If the program committee or reviewers require the validation set performance, we will include it in the final version of the paper.

- *A clear definition of the specific evaluation measure or statistics used to report results*: Our evaluation metrics are standard and widely used by the question answering community. We provide their details in the main paper in Sec. 5 and Sec. 6.

Question from NQ test	Answer	Top-1 Document Retrieved by ICT + Supervised
what parts make up the peripheral nervous system	autonomic nervous system	...The connection between CNS and organs allows the system to be in two different functional states: sympathetic and parasympathetic. The peripheral nervous system is divided into the somatic nervous system, and the **autonomic nervous system**. The somatic nervous system is under voluntary control, and transmits signals from the brain to end organs such as muscles. The sensory nervous system is part of the somatic nervous system and transmits signals from senses such as taste and touch (including fine touch and gross touch) to the spinal cord and brain...
when is the new season of wentworth coming out	19 June 2018	...In a similar manner, a 12-episode fourth season was announced before the airing of the third season on 27 February 2015. It began airing from 10 May 2016. Cormack confirmed a fifth season had been commissioned on 19 July. The twelve-part series premiered on 4 April 2017. On 9 May 2017, Showcase announced that the series has been renewed for a sixth season, which premiered on **19 June 2018**. A seventh season was commissioned in April 2018, before the sixth-season premiere, with filming commencing the following week and a premiere set for 2019...
who challenged the aristotelian model of a geocentric universe	Copernicus	...("On the Revolutions of the Heavenly Spheres"), which posited that the Earth and the other planets instead revolved around the Sun. The geocentric system was still held for many years afterwards, as at the time the Copernican system did not offer better predictions than the geocentric system, and it posed problems for both natural philosophy and scripture. The Copernican system was no more accurate than Ptolemyś system, because it still used circular orbits. This was not altered until Johannes Kepler postulated that they were elliptical (Keplerś first law of planetary motion). ...

Table 9: Examples of top-1 retrieved documents from the NQ test as outputted from the ICT + Supervised retriever. If the answer exists in the document, it is highlighted in bold.

B.2 For all results involving multiple experiments, such as hyperparameter search

- *The exact number of training and evaluation runs:* We provide training details for all models in Appendix A. Specifically, for the fine-tuning experiments, we train the models until convergence, which is 80 epochs for retriever models and 10 epochs for answer extraction models. We evaluate the model after each epoch on the validation set and save the best checkpoint according to their performance on the corresponding evaluation metric.

- *Hyperparameter configurations for best-performing models*: We provide the hyperparameter settings in Appendix A.

- *The bounds for each hyperparameter*: As described in Appendix A, our model and training setting uses standard hyperparameters such as different dropouts $\in [0, 1)$, warmup ratio of optimizer $\in [0.01, 0.05]$, weight regularization $\in [0, 1]$, and learning rate $\in [1e^{-4}, 1e^{-5}]$. The model hyperparameters includes model dimensions $d \in \{768, 1024\}$, number of layers $\in \{12, 24\}$.

- *The method of choosing hyperparameter values (e.g., uniform sampling, manual tuning, etc.) and the criterion used to select among them (e.g., accuracy)*: We performed manual hyperparameter tuning. We also performed

tuning of the number of warmup steps for the Adam optimizer. We selected the best hyperparameter using performance on the validation set.

- *Summary statistics of the results (e.g. mean, variance, error bars, etc.)*: All of our experiments are compute expensive large-scale runs utilizing a lot of resources such as CPUs, GPUs and take time ranging from tens of hours to several days. Therefore, due to computational and time constraints performing multiple runs for each experiment was not feasible. Therefore, we adopted the approach of using the same seed value (1234) for all the training runs including both pre-training and finetuning experiments.

B.3 For all datasets used

- *Details of train/validation/test splits*: We use the standard training / dev / test splits whose details are provided in Sec. 4.
- *Relevant statistics such as number of examples and label distributions*: We provide dataset statistics details in Table 1.
- *An explanation of any data that were excluded, and all pre-processing steps*: We include the relevant details in Sec. 4.
- *For natural language data, the name of the language(s)*: Our datasets are in English language.
- *A link to a downloadable version of the dataset or simulation environment*: Both the datasets of NQ and TriviaQA are open-source and widely used by the community. NQ is available at: https://ai.google.com/research/NaturalQuestions/download. TriviaQA is available at: http://nlp.cs.washington.edu/triviaqa/. We make use of the NQ, TriviaQA, and Wikipedia datasets as open-sourced by the DPR authors (Karpukhin et al., 2020) here: https://github.com/facebookresearch/DPR/blob/master/data/download_data.py.
- *For new data collected, a complete description of the data collection process, such as instructions to annotators and methods for quality control*: This is not applicable to this work.