Relevance-guided Supervision for OpenQA with ColBERT

Omar Khattab
Stanford University
okhattab@stanford.edu

Christopher Potts
Stanford University
cgpotts@stanford.edu

Matei Zaharia
Stanford University
matei@cs.stanford.edu

arXiv:2007.00814v2 [cs.CL] 2 Aug 2021

Abstract

Systems for Open-Domain Question Answering (OpenQA) generally depend on a *retriever* for finding candidate passages in a large corpus and a *reader* for extracting answers from those passages. In much recent work, the retriever is a learned component that uses coarse-grained vector representations of questions and passages. We argue that this modeling choice is insufficiently expressive for dealing with the complexity of natural language questions. To address this, we define ColBERT-QA, which adapts the scalable neural retrieval model ColBERT to OpenQA. ColBERT creates fine-grained interactions between questions and passages. We propose an efficient weak supervision strategy that iteratively uses ColBERT to create its own training data. This greatly improves OpenQA retrieval on Natural Questions, SQuAD, and TriviaQA, and the resulting system attains state-of-the-art extractive OpenQA performance on all three datasets.

1 Introduction

The goal of Open-Domain Question Answering (OpenQA; Voorhees and Tice 2000) is to find answers to factoid questions in potentially massive unstructured text corpora. Systems for OpenQA typically depend on two major components: a *retrieval model* to find passages that are relevant to the user's question and a *machine reading model* to try to find an answer to the question in the retrieved passages. At its best, this should combine the power and scalability of current information retrieval (IR) models with recent advances in machine reading comprehension (MRC). However, if the notions of relevance embedded in the IR model fail to align with the requirements of question answering, the MRC model will not reliably see the best passages and the system will perform poorly.

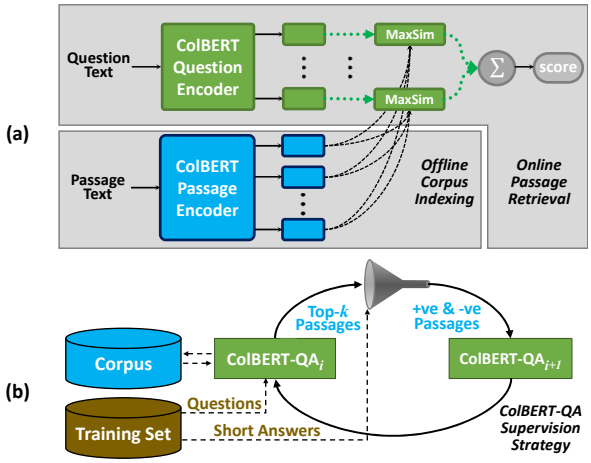

Figure 1: Sub-figure (a) depicts the ColBERT retrieval model (Khattab and Zaharia, 2020). ColBERT encodes questions and passages into multiple embeddings and allows those to interact via a scalable maximum-similarity (MaxSim) mechanism. Sub-figure (b) illustrates our proposed ColBERT-QA training strategy: we use an existing retrieval model to collect the top-k passages for every training question and, with a simple heuristic, sort these passages into positive (+ve) and negative (–ve) examples, using those to train another, more effective retriever. This process is applied thrice, and the resulting ColBERT-QA is used in the OpenQA pipeline.

Many prior approaches to OpenQA rely on classical IR models (e.g., BM25; Robertson et al. 1995) whose notions of relevance are not tailored to questions. In effect, this reduces the OpenQA problem to few-passage MRC, imposing a hard limit on the quality of the passages seen by the MRC model. Recent work has sought to address this problem by *learning* to retrieve passages. For instance, Guu et al. (2020) and Karpukhin et al. (2020) jointly train vector representations of both passages and questions to support similarity-based retrieval. This has led to state-of-the-art performance on multiple OpenQA datasets.

However, existing OpenQA retrievers exhibit two major limitations. First, the representations learned by these models are relatively coarse: they

encode each passage into a *single* high-dimensional vector and estimate relevance via one dot-product. We argue this is not expressive enough for reliably matching complex natural-language questions to their answers. Second, existing systems present substantial tradeoffs when it comes to supervision: they expect hand-labeled "positive" passages, which may not always exist; or they use simple models like BM25 to sample "positives" and "negatives" for training, which may provide weak positives and unchallenging negatives; or they conduct retrieval within the training loop, which requires frequently re-indexing a large corpus (e.g., tens or hundreds of times in REALM; Guu et al. 2020) or a frozen document encoder that cannot adapt to the task (e.g., as in RAG; Lewis et al. 2020). We argue that supervision methods in this space need to be more flexible and more scalable.

We tackle both limitations with ColBERT-QA.[1] To address the first problem, we leverage the recent neural retrieval model ColBERT (Khattab and Zaharia, 2020) to create an end-to-end system for OpenQA. Like other recent neural IR models, ColBERT encodes both the question and the document using BERT (Devlin et al., 2019). However, a defining characteristic of ColBERT is that it explicitly models *fine-grained interactions* (Figure 1(a)) between the question and document representations (§3), in particular by comparing each question term embedding with each passage term embedding. Crucially, ColBERT does so while scaling to millions of documents and maintaining low query latency. We hypothesize that this form of interaction will permit our model to be sensitive to the nature of questions without compromising the OpenQA goal of scaling to massive datasets.

To address the second problem, we propose *relevance-guided supervision* (RGS), an efficient weak-supervision strategy that allows the retriever to guide its own training *without* frequent re-indexing or freezing the document encoder. Instead of expensive pretraining, RGS starts from an existing weak retrieval model (e.g., BM25) to collect the top-k passages for every training question and uses a provided weak heuristic to sort these passages into positive and negative examples, relying on the ordering imposed by the retriever. These examples are then used to train a more effective retriever, and this process is applied 2–3 times, with the resulting retriever deployed in the OpenQA pipeline. Crucially for scaling to large corpora, RGS only requires re-indexing the corpus once or twice during training and correspondingly only retrieves positives and negatives in 2–3 large batches. In doing so, RGS permits fine-tuning of the document encoder during all of training, freeing it to co-adapt with the query encoder to the task's complexities.

To assess ColBERT-QA, we report on experiments with Natural Questions (Kwiatkowski et al., 2019), SQuAD (Rajpurkar et al., 2016a), and TriviaQA (Joshi et al., 2017a). We adopt the OpenQA formulation in which the passage is not given directly as gold evidence, but rather must be retrieved, including during training. Further, we focus on the standard *extractive* OpenQA setup, where the reader model extracts the answer string from one of the retrieved passages. On all three datasets, ColBERT-QA achieves state-of-the-art retrieval and extractive OpenQA results.

In summary, our contributions are:

1. We propose relevance-guided supervision (RGS), an efficient iterative strategy for fine-tuning a retriever given a weak heuristic.

2. We conduct the first systematic comparison between ColBERT's fine-grained interaction and recent single-vector retrievers like DPR. We find that ColBERT exhibits strong transfer learning performance to new OpenQA datasets and that fine-tuned ColBERT delivers large gains over existing OpenQA retrievers.

3. We apply RGS to ColBERT and a single-vector retriever, and find that each improves by up to 2.3 and 3.2 points in Success@20, respectively. Our resulting ColBERT-QA system establishes state-of-the-art retrieval and downstream performance.

2 Background & Related Work

2.1 Machine Reading Comprehension

MRC refers to a family of tasks that involve answering questions about about text passages (Clark and Etzioni, 2016). In recent work, the potential answers are usually selected from a set of pre-defined options (Hirschman et al., 1999; Richardson et al., 2013; Iyyer et al., 2014) or guaranteed to be a substring of the passage (Yang et al., 2015; Rajpurkar et al., 2016a; Kwiatkowski et al., 2019; Joshi et al., 2017a). Here, we start from the version of the task

[1] https://github.com/stanfordnlp/ColBERT-QA

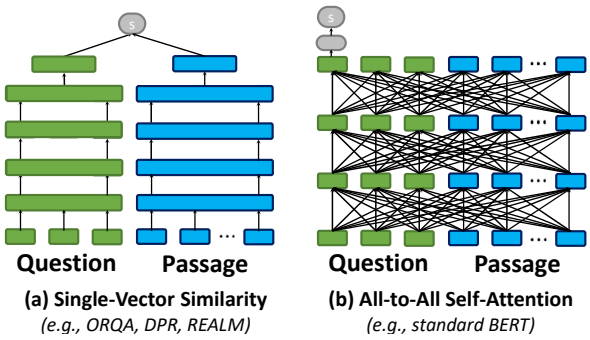

Figure 2: A comparison between two extremes of building neural retrievers with Transformers. On the left, single-vector models independently encode each question and passage into a vector and model relevance as one dot-product. On the right, all-to-all attention models feed a sequence concatenating the question and each passage through the encoder, using layers of self-attention to estimate a relevance score. Diagrams adapted from Khattab and Zaharia (2020) with permission.

established by Yang et al. and Rajpurkar et al., in which the passage p and question q are MRC inputs and the correct answer $\hat{a}$ is a literal substring of p.

2.2 OpenQA

In its most general formulation, the OpenQA task is defined as follows: given a large corpus C of documents (e.g., Wikipedia or a fraction of the Web) and a question q, the goal is to produce the correct short answer $\hat{a}$. Much of the earliest work in question answering adopted this paradigm (Voorhees and Tice, 2000; Ferrucci et al., 2010; Kushman et al., 2014). Our focus is on the specific version of OpenQA that is a relaxation of the standard MRC paradigm (Chen et al., 2017; Kratzwald and Feuerriegel, 2018): the passage is no longer given, but rather needs to be retrieved from a corpus, and the MRC component must seek answers in the retrieved passages without a guarantee that an answer is present in any of them.

2.3 Retrieval Models for OpenQA

Mainstream approaches use sparse retrieval models (e.g., BM25; Robertson and Zaragoza 2009) and various heuristics (e.g., traversal of Wikipedia hyperlinks) to retrieve a set of k passages that are relevant to the question q. This (closed) set of passages is often *re-ranked* with a Transformer encoder (Vaswani et al., 2017) to maximize precision, as in Wang et al. (2019) and Yang et al. (2019). Subsequently, mainstream approaches deploy a reader to read each of these k passages and return the highest-scoring answer, $\hat{a}$, present as a span in one or more of the k passages. This approach is represented in Chen et al. (2017), Min et al. (2019a), and Asai et al. (2020), among many others.

More recent literature has shown that there is value in learning the retriever, including ORQA (Lee et al., 2019), REALM (Guu et al., 2020), and DPR (Karpukhin et al., 2020). Common to all three is the BERT two-tower architecture depicted in Figure 2(a). This architecture encodes every question q or passage d into a single, high-dimensional vector and models relevance via a single dot-product. However, these models differ substantively in supervision.

Lee et al. (2019) propose the Inverse Close Task (ICT), a self-supervised task to pretrain these encoders for retrieval. The question encoder is fed a random sentence q from the corpus and the passage encoder is fed a number of context passages, one of which is the true context of q, and the model learns to identify the correct context. Guu et al. (2020) extend this task to retrieval-augmented language modeling (REALM), where the encoders are optimized to retrieve passages that help with a Masked Language Modeling task (Devlin et al., 2019). After pretraining, both ORQA and REALM freeze the passages index and encoder, and fine-tune the question encoder to retrieve passages that help a jointly-trained reader model extract the correct answer. Though self-supervised, the pretraining procedures of ORQA and especially REALM are computationally expensive, owing to the amount of data they must see and as REALM re-indexes the corpus during pretraining every 500 steps (out of 200k).

Very recently, Karpukhin et al. (2020) propose a dense passage retriever (DPR) that directly trains the architecture in Figure 2(a) for retrieval, relying on a simple approach to collect positives and negatives. For every question q in the training set, Karpukhin et al. (2020) recover the hand-labeled evidence passages (if available; otherwise they use BM25-based weak supervision) as positive passages and sample negatives from the top-k results by BM25. Importantly, they also use in-batch negatives: every positive passage in the training batch is a negative for all but its own question. Using this simple strategy, DPR considerably outperforms both ORQA and REALM and established a new state-of-the-art for extractive OpenQA.

The single-vector approach taken by ORQA, REALM, and DPR can be very fast but allows for only shallow interactions between q and d. Khattab

and Zaharia (2020) describe a spectrum of options for encoding and scoring in neural retrieval, within which this single-vector paradigm is one extreme. At the other end of this spectrum is the model shown in Figure 2(b). Namely, we could feed BERT a concatenated sequence $\langle q, d \rangle$ for every passage d to be ranked and fine-tune all BERT parameters against the ranking objective. This allows for very rich interactions between the query and document. However, such a model is prohibitively expensive beyond *re-ranking* a small set of passages already retrieved by much cheaper models.

ColBERT (Figure 3) seeks a middle ground: it separately encodes the query and document into token-level representations and relies on a scalable scoring mechanism that creates rich interactions between q and d. Central to their efficiency, the interactions are *late* in that they involve just the output states of BERT. Essential to their quality, they are *fine-grained* in that they cross-match token representations of q and d against each other.

Our work differs from Khattab and Zaharia (2020) in two major ways. First, they assume gold-evidence positives, which may not exist in OpenQA, and use BM25 negatives, which we argue is insufficient for an end-to-end retriever. We propose a simple and efficient strategy, namely RGS, that improves training quality. Second, Khattab and Zaharia (2020) report gains against a single-vector ablation of their IR system, but we ask if these gains hold against (concurrent) state-of-the-art single-vector models in OpenQA, where a reader could in principle overshadow retrieval gains. Our work confirms that late interaction is superior even (if not especially) in OpenQA: we report considerable gains when using traditional supervision and even larger gains with RGS. We also report strong results when using an out-of-domain transfer learning setting from IR, which work by Akkalyoncu Yilmaz et al. (2019) considers but in the context of neural re-rankers and between standard IR tasks.

2.4 Supervision Paradigms in OpenQA

Broadly, there are two paradigms for training and evaluating OpenQA models.

Gold-Evidence Supervision. Some OpenQA datasets supply annotated *evidence passages* or human-curated contexts from which the gold answers can be derived. For such datasets, it is natural to rely on these labels to train the retriever, akin to typical supervision in many IR tasks. For example, Natural Questions contains labeled "long answers", which DPR uses as positive passages.

However, using gold-evidence labels is not possible with OpenQA datasets that only supply question–answer string pairs, like TriviaQA and WebQuestions (Berant et al., 2013).[2] Moreover, manual annotations might fail to reflect the richness of passages answering the same question in the corpus, possibly due to biases in how passages are found and annotated.

Weak Supervision. Addressing these limitations, weakly-supervised OpenQA (Lee et al., 2019; Guu et al., 2020) supplies its own evidence passages during training. To do so, it exploits a question's short answer string as a crucial supervision signal. For a question q in the training set, a passage that contains q's answer string $\hat{a}$ is treated as a potential candidate for a positive passage. To weed out *spurious* matches, additional strategies are often introduced. We categorize those into *retrieve-and-filter* and *inner-loop retrieval* strategies. In retrieve-and-filter, an existing retriever and a weak heuristic are essentially intersected to find promising passages for training. For instance, Wang et al. (2019) and Karpukhin et al. (2020) consider only passages highly-ranked by BM25 for q as candidates, where the answer string can be a more reliable signal.

In inner-loop retrieval, the training loop retrieves passages for each example using the model being fine-tuned. This is conducted by ORQA, REALM, and RAG (Lewis et al., 2020), approaching "end-to-end" training of the retriever. However, such inner-loop retrieval requires major approximations, since it is infeasible to compute forward and backward passes over the entire collection for every training batch. Here, ORQA, REALM, and RAG *freeze* their document encoder (and the indexed vectors) when fine-tuning for OpenQA, which restricts the adaptability of the model to this task and/or to new corpora. During pretraining, REALM does not freeze the document encoder, but then it has to very frequently re-index the corpus with training, and the method suffers quality loss if the index is allowed to become stale.

While existing retrieve-and-filter approaches reflect the naive term-matching biases of BM25, and existing inner-loop retrieval strategies restrict training the document encoder or require frequent re-indexing and repeated retrieval, RGS offers a scal-

[2] TriviaQA provides automatic "distant supervision" labels.

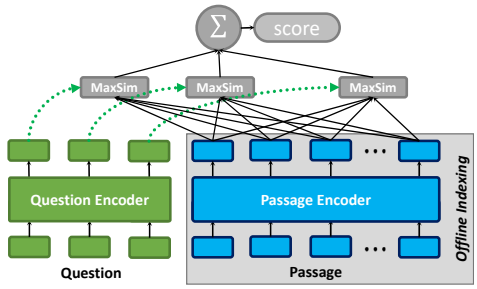

Figure 3: The general architecture of ColBERT given a question and a passage. Diagram adapted from Khattab and Zaharia (2020) with permission.

able and effective alternative that allows the retriever itself to collect the training examples while fine-tuning the document encoder and only re-indexing 1–2 additional times.

Weak supervision is also a topic in IR. For instance, Dehghani et al. (2017) explore using BM25 as a teacher model to train a neural ranker. Other weak signals in IR include anchor text (Zhang et al., 2020) and headings text (MacAvaney et al., 2019), treated as queries for which the target content is assumed relevant. Lastly, user interactions have long been used for supervision in search and recommendation. However, the gold answer string is unique to OpenQA and we show it can lead to large quality gains when combined with an effective retriever.

3 ColBERT-QA

We now describe our ColBERT-QA system. We propose *relevance-guided supervision* (RGS) (§3.2), a scalable strategy that uses the retriever being trained and a weak heuristic to gather training examples in a few discrete rounds. As Figure 1 illustrates, we use RGS to fine-tune ColBERT models in three stages arriving at **ColBERT-QA**$_1$, **ColBERT-QA**$_2$, and **ColBERT-QA**$_3$.

3.1 The ColBERT Model

ColBERT capitalizes on BERT's capacity for contextually encoding text into token-level output representations. Given a question q, we tokenize it as $[q_1, \ldots, q_n]$. The token sequence is truncated if it is longer than N (e.g., $N = 32$) or padded with `[MASK]` tokens if it is shorter. Khattab and Zaharia (2020) refer to this padding as *query augmentation* and show that it improves ColBERT's effectiveness. These tokens are processed by BERT to obtain a sequence of output vectors $\mathbf{q} = [\mathbf{q}_1, \ldots, \mathbf{q}_N]$.

The encoding of a passage d given tokens $[d_1, \ldots, d_m]$ follows the same pattern but no augmentation is performed. BERT processes these into output vectors $\mathbf{d} = [\mathbf{d}_1, \ldots, \mathbf{d}_m]$. For both $\mathbf{q}$ and $\mathbf{d}$, we apply a linear layer on top of each representation to control the output dimension. Each vector is finally rescaled into unit length.

Let E_q (length N) and E_d (length m) be the final vector sequences derived from $\mathbf{q}$ and $\mathbf{d}$. The ColBERT scoring mechanism is given as follows:

$$S_{q,d} = \sum_{i=1}^{N} \max_{j=1}^{m} E_{q_i} \cdot E_{d_j}^T \quad (1)$$

As illustrated in Figure 3, for every query embedding, ColBERT computes its maximum-similarity (MaxSim) score over all embeddings in the passage, then sums these scores. This mechanism "softly" matches each query term with a passage term, producing a similarity score. Crucially, this computation scales to billions of tokens corresponding to many millions of passages, permitting ColBERT to retrieve passages directly from a massive corpus, not restricted by the limitations of a "first-stage" retriever (e.g., BM25 or single-vector models).

Overall, ColBERT balances the scalability of single-vector retrievers and the richness of BERT's query–document interaction (§2.3). For IR, Khattab and Zaharia (2020) show that ColBERT outperforms a single-vector retriever and that, while ColBERT's precision is comparable to that of BERT when re-ranking a closed set of passages (e.g., top-1000 passages retrieved by BM25), its scalability (allowing full-corpus retrieval) boosts recall.

ColBERT is trained to give positive passages higher scores than negative passages. More specifically, it requires triples $\langle q, d^+, d^- \rangle$, where q is a short query, d^+ is a positive passage, and d^- is a negative passage. We score each of d^+ and d^- according to Equation 1 given q, and treat this as a binary classification task, optimizing the model parameters using a cross-entropy loss. For generic IR applications, negative passages can be sampled uniformly from the top-k (e.g., $k = 1000$) results for an existing IR system. Positive passages are more difficult to obtain; IR evaluations generally require *labeled* positives as provided by datasets like MS MARCO Ranking (Nguyen et al., 2016).

3.2 Relevance-Guided Supervision

OpenQA often lacks gold-labeled passages and instead has *short answer strings* as supervision signals. This signal is known to lead to many spurious matches and existing work tackles this by either

Algorithm 1: Relevance-guided Supervision, given a weak heuristic H

Input: Training questions Q and corpus C
Input: Supervision heuristic H
Input: Retrieval model R_0, number of rounds n
Output: Stronger retriever R_n

1 **for** round $t \leftarrow 1$ ***to*** n **do**
2 Index corpus C for retrieval with R_t.
3 $Rank_t \leftarrow \{R_i.retrieve(q) : q \in Q\}$
4 $P_t = \{H.getPositives(r[0:k^+])[0:t]$
5 $: r \in Rank_t\}$
6 $N_t = \{H.getNegatives(r[0:k^-])$
7 $: r \in Rank_t\}$
8 $T_t = \{\langle q, p, n\rangle : q \in Q,$
9 $p \in P_t[q], n \in N_t[q]\}$
10 Train a new retriever R_{t+1} using triples T_t.
11 **end**
12 **return** *Final retriever* R_n

BM25-based retrieve-and-filter, which can be simplistic, or inner-loop retrieval methods, which can be expensive and restrictive (§2.4).

Algorithm 1 outlines the proposed RGS, which seeks to alleviate these limitations by *outer-loop* retrieval. RGS takes as input a set of training question Q, a corpus of passages C, and an initial weak retrieval model (e.g., BM25). In general, RGS assumes a task-specific weak heuristic H, which in this work indicates whether a passage contains a question's short-answer string. RGS proceeds in n discrete rounds (we set $n = 3$ in this work). Each round uses as a building block the retrieve-and-filter mechanism. For every question q in the *training* set, we retrieve R's top-k results (Line 3). Then, we take as positives (Line 4) the highest-ranked t (e.g., $t = 3$) passages *that contain q's short answer*, restricted to the top-k^+ (e.g., $k^+ = 20$) results. Every passage in the top-k^- (potentially with $k^- \gg k^+$) that does not contain the short answer is treated as a negative (Line 6). We train a new retriever with these examples (Line 10), and repeat until all rounds are complete.

Our key hypothesis is that more effective retrieval would collect more accurate and diverse positives and more challenging and realistic negatives. Importantly, RGS collects positives and negatives entirely outside the training loop. As a result, we can flexibly sample positives and negatives from large depths and we can fine-tune the entire model (i.e., including the document encoder) without having to frequently re-index the corpus. However, applying this process more than once on the training set risks overfitting in a way that rewards a narrow set of positives and drifts in the negatives it samples. To avoid this, we use a deterministic split of the training set such that no model sees the same question during both training and positive/negative retrieval. To this end, we create arbitrary equal-sized splits of all training sets before starting RGS and train each ColBERT-QA model with 50% of the training questions.[3]

To bootstrap ColBERT-QA, §4 selects R_0 as the bag-of-words model BM25. Training yields ColBERT-QA$_1$, which enables us to establish ColBERT's high quality even with simple weak supervision. We then apply RGS in two rounds, leading to ColBERT-QA$_2$ and ColBERT-QA$_3$, whose retrieval quality consistently outperforms ColBERT-QA$_1$ and ultimately surpasses existing retrievers by large margins.

3.3 Reader Supervision for OpenQA

Like many OpenQA systems, ColBERT-QA uses a BERT encoder as a reader. After retrieval, the reader takes as input a concatenation of a question q and a passage d: `[CLS]` q `[SEP]` d `[SEP]`, for each passage d in the top-k set retrieved. The reader scores each individual short span s in each passage d. Similar to Lee et al. (2019), we model the probability of each span s, namely $P(s|d)$, as $P(s|d) \propto \texttt{MLP}(h_{\texttt{start}(s)}; h_{\texttt{end}(s)})$ for the output embeddings at the start and end tokens of span s.

To train the reader, we use a set of triples $\langle q, d^+, d^-\rangle$ for every question q in the training set with positive d^+ and negative d^-. We collect these triples using the same general heuristic used for retrieval training, extracting them from the ranking model R whose passages we feed to the reader during inference. We treat every span matching the answer in d^+ as correct for optimization and minimize the maximum marginal likelihood of the correct answer, normalizing the span probabilities over the spans in both documents. For the subset of spans $\hat{S}$ that match the gold answer in d^+, we minimize the loss $-\log \sum_{s\in\hat{S}} P(s|d^+)$.

[3] We note that, if desired, it is easy to extend this procedure to train a ColBERT-QA model on all questions: train two independent copies of the supervising model R, one for each half of the training set, and combine their rankings to supervise the final model. We leave such improvements to future work.

Name	Source(s) of Positives / Negatives	Success@20 (test) NQ	TQ	SQ	EM (dev) NQ	TQ	SQ
Baseline Retrievers							
BM25 (Anserini)	n/a	64.0	77.3	71.4	-	-	-
DPR (core variant) (Karpukhin et al., 2020)	Gold / BM25	78.4	79.4	63.2	-	-	-
DPR (best variant) (Karpukhin et al., 2020)	Gold / BM25	79.4	79.9	71.5	-	-	-
Single-vector ColBERT (ablation)	BM25	78.3	82.8	72.9	-	-	-
ColBERT-guided Supervision Retrievers (ours)							
Out-of-Domain ColBERT-QA	MS MARCO	79.1	80.3	76.5	-	-	-
ColBERT-QA$_1$	BM25	82.9	84.7	82.1	47.1	69.5	49.4
ColBERT-QA$_2$	ColBERT-QA$_1$	85.0‡	85.3‡	**83.9**‡	**48.6**	**70.2**	51.0
ColBERT-QA$_3$	ColBERT-QA$_2$	**85.3**‡	**85.6**‡	83.7‡	47.9	69.8	**51.8**

Table 1: A comparison between baseline retrieval models and approaches for training ColBERT-QA. A subscript 1, 2, or 3 on ColBERT-QA indicates the number of fine-tuning rounds for OpenQA retrieval. We report Success@20 on the test sets for comparison with Karpukhin et al. (2020)'s results. We reserve the test-set EM evaluation for §5. For Success@20, ‡ indicates significant improvement over ColBERT-QA$_1$; details in the main text.

Metric	NQ		TQ		SQ	
	C3	Δ_{C1}	C3	Δ_{C1}	C3	Δ_{C1}
k=1	52.9	+5.7	68.0	+2.6	56.0	+2.1
k=5	75.6	+4.2	80.7	+1.6	75.1	+2.1
k=10	81.4	+2.8	83.6	+1.3	79.6	+1.4
k=20	85.3	+2.3	85.6	+0.9	83.7	+1.6
k=50	88.6	+1.6	87.4	+0.5	87.7	+1.8
k=100	90.1	+1.1	88.4	+0.2	89.4	+1.4
MRR@100	62.8	+4.8	73.7	+2.2	64.5	+2.1

Table 2: Test-set retrieval quality of ColBERT-QA$_3$ (with gains over ColBERT-QA$_1$) at various depths.

4 Evaluating ColBERT-QA's Retrieval

ColBERT-QA presents many options for training the retriever so it extracts the best passages for the reader. This section explores a range of these options to address the following key questions.

Our first question concerns ColBERT's retrieval modeling capacity. Can ColBERT's fine-grained interactions improve the accuracy of OpenQA retrieval when tuned for this task? We consider this under a weak supervision paradigm: training based on BM25 ranking. In §4.2, we find that ColBERT is highly effective in this setting, considerably outperforming classical and single-vector retrieval.

Our second question concerns an out-of-domain version of our model. ColBERT is standardly trained to perform IR tasks. Is this form of transfer learning from IR sufficient for effective OpenQA? If so, then this might be an appealingly modular option for many applications, allowing system designers to focus on the reader. In §4.3, we show that ColBERT succeeds in this setting.

Our third set of questions concerns how best to supervise ColBERT-QA for OpenQA. In particular, can relevance-guided supervision improve on standard BM25-based supervision by capitalizing on the structure and effectiveness of ColBERT? We find that the answer is "yes"; ColBERT-QA with RGS consistently proves to be the best method (§4.4), while only requiring 1–2 additional iterations of indexing and training.

4.1 Methods

Similar to Chen et al. (2017), Guu et al. (2020), and Karpukhin et al. (2020), we study the retrieval quality in OpenQA by reporting Success@k (S@k for short), namely, the percentage of questions for which a retrieved passage (up to depth k) contains the gold answer string. This metric reflects an upper bound on the performance of an extractive reader that reads k passages, assuming it always locates the answer if present. However, readers in practice are affected by the quality and number of passages that contain the answer string and by passages that do *not* contain the answer. To evaluate this more directly, we suggest employing a BERT-large model (with "whole-word-masking" pretraining) as a reader, trained for each retriever as described in §3.3. We report the exact-match (EM) score corresponding to each retriever–reader combination on the validation set. Hyperparmeter tuning is described in §B.

We conduct these experiments on the open versions of the Natural Questions (NQ), SQuAD (SQ), and TriviaQA (TQ) datasets. Additional details on these datasets are in our appendices. Our retrieval results are summarized in Table 1. To describe how each model was trained, we report its sources of positive and negative passages.

4.2 Baselines

The top of Table 1 shows the retrieval quality of our baselines: BM25 and the recent state-of-the-art DPR. As Karpukhin et al. (2020) report, DPR has a considerable edge over BM25 on NQ and TQ but not SQuAD. The authors attribute the weak performance on SQuAD to the presence of high lexical matching between its questions and evidence passages. Indeed, as Lee et al. (2019) argues, this might stem from the SQuAD annotators writing the questions after reading the passage.

Unlike single-vector models, ColBERT is capable of fine-grained contextual matching between the words in the question q and the passage d so, by design, it can *softly* match contextual representations of words across q and d. The results on NQ, TQ, and SQ confirm that fine-grained modeling exceeds the quality attained with single-vector representations, consistently outperforming the baselines across all three datasets.

As described in §2.3, DPR is trained using in-batch negatives and differs from ColBERT in a number of ways (e.g., weak vs. gold supervision on NQ, the depth of negatives used in training, etc.). To better account for differences, we include a "single-vector ColBERT" ablation trained in the same manner as our ColBERT-QA$_1$ model. This ablation is similar to the ablation model evaluated by (Khattab and Zaharia, 2020) on MS MARCO and it resembles DPR in that it uses a dot-product of a 768-dimensional vector extracted from BERT's `[CLS]` token for each query or passage. Unlike DPR, we do not use in-batch negatives for this model to more directly compare with ColBERT.[4] The results validate that fine-grained interaction is a stronger retrieval choice for OpenQA.

4.3 Evaluating ColBERT out of domain

We now directly examine out-of-domain ColBERT-QA in Table 1. For this, we train ColBERT as in Khattab and Zaharia (2020) on the MS MARCO Ranking dataset, a natural choice for transfer learning to OpenQA. To elaborate, it is originally derived from the MRC dataset with the same name and contains many question-like search queries as well as other "free-form" queries. Unless otherwise stated, we evaluate ColBERT-based models by indexing and retrieving from the full corpus.

The results in Table 1 show that out-of-domain ColBERT-QA outperforms BM25 and, in fact, is already competitive with the DPR retriever across the three datasets.[5] To further contextualize these results, we note that out-of-domain ColBERT-QA is more effective than "zero-shot" ORQA and REALM as reported in (Guu et al., 2020): its development-set Success@5 exceeds 65% on NQ whereas both zero-shot ORQA and REALM are below 40%, on a different random train/dev split of the same data. We conclude that transfer learning with models having strong inductive biases, like ColBERT's matching, offers a promising alternative to expensive unsupervised retrieval pretraining.

4.4 Evaluating Relevance-Guided Supervision

We now turn our attention to the proposed RGS strategy, defined in §3.2. RGS seeks to exploit ColBERT's high precision to collect high quality positives and challenging negative and to leverage its recall to collect richer positives with a wider coverage of questions. Below, we test how these hypotheses fare after one, two, and three stages of fine-tuning. We first focus on Success@20 then consider the EM metric. We start the fine-tuning of each ColBERT retriever from BERT-base.

We bootstrap ColBERT-QA in a weakly-supervised fashion on each target dataset: **ColBERT-QA$_1$** relies on BM25 for collecting its positive and negative passages. As reported in Table 1, it already outperforms every baseline considered so far, including DPR, by considerable margins. This strong performance emphasizes that ColBERT's capacity to learn from *noisy* supervision examples can lead to more accurate OpenQA retrieval. It also suggests that adapting to the characteristics of OpenQA-oriented retrieval, the corpus, and the target QA datasets can beat high-quality retrieval training for generic IR. This is evidenced

[4] Unlike our ColBERT models, we found this single-vector ablation performs better under *re-ranking* BM25 top-1000 than under full-corpus retrieval on the validation sets; hence, we report its re-ranking performance.

[5] If we plug ColBERT-QA$_1$'s retriever over this out-of-domain retrieval, we get 45.8, 66.5, and 44.8 EM (dev) on NQ, TQ, and SQ, respectively.

by the consistent gains of ColBERT-QA$_1$ over the already-effective out-of-domain model.

Next, we apply RGS by using ColBERT-QA$_1$ itself to create data for further training, yielding **ColBERT-QA$_2$**, and take this a step further in using ColBERT-QA$_2$ to supervise **ColBERT-QA$_3$**. As shown, ColBERT-QA$_2$ and ColBERT-QA$_3$ are consistently the most effective, while only requiring that we index and retrieve positives and negatives one or two times. In particular, they raise test-set Success@20 over the next best, ColBERT-QA$_1$, by up to 0.9–2.4 points on the three datasets, a sizeable increase in the fraction of questions that can be answered by an extractive reader. We conduct a more granular comparison between ColBERT-QA$_3$ and ColBERT-QA$_1$ in Table §2, which shows that the gains are even larger at smaller depths (e.g., up to 5.7 points in S@1), a property useful in applications where a reader only has the budget to consume a few passages per question.[6]

As mentioned in §3, relying on ColBERT's scalability permits us to leverage its accuracy by applying it to the entire corpus, enabling improved recall. For instance, if we use ColBERT-QA$_3$ for re-ranking BM25 top-1000 instead of end-to-end, full-corpus retrieval, its Success@20 score drops on the development set from 84.3% to just 80.7%.

It is standard to evaluate OpenQA retrievers using the occurrence of answers in the retrieved passages. Having done this, we consider another evaluation paradigm: the *development-set EM scores* resulting from training a reader corresponding to each retriever. The EM scores in Table §1 reinforce the value of RGS (i.e., ColBERT-QA$_2$ and ColBERT-QA$_3$ outperforming ColBERT-QA$_1$ by up to 0.7–2.4 EM points). Simultaneously, the EM scores show that a higher fraction of answerable questions (i.e., higher Success@k) does not always imply higher EM with a particular reader. We hypothesize that this depends on the correlation of the retriever and reader mistakes: a stronger retriever may find more "positive" passages but it can also retrieve more challenging negatives that could confuse an imperfect reader model. Considering this, we next evaluate the full ColBERT-QA system against existing end-to-end OpenQA systems.

[6]With the Wilcoxon signed-rank test (with $p < 0.05$ and Bonferroni correction) on Success@20, we find that ColBERT-QA$_2$ and ColBERT-QA$_3$ are always statistically significantly better than ColBERT-QA$_1$ (pre-correction p-values all less than 0.0001) and detect no such difference between ColBERT-QA$_2$ and ColBERT-QA$_3$.

5 End-to-end OpenQA Evaluation

Our primary question is at this point is whether ColBERT-QA's retrieval improvements lead to better end-to-end OpenQA quality. We find that the answer is consistently positive: ColBERT-QA leads to state-of-the-art extractive OpenQA results.

5.1 Datasets

We continue to evaluate on NQ, SQuAD, and TQ, using their *open* versions (Lee et al., 2019), which discard the evidence passages of the validation and test questions. As is standard, for each of the three datasets, we randomly split the original training set for training and validation and use the original development set for testing.

Like the results in §4, for all three datasets, we primarily conduct our evaluation on the December 2018 Wikipedia dump used by the majority of our baselines. However, Asai et al. (2020) argue that for SQuAD, comparisons should be made using another common 2016 dump closer to the creation date of this dataset. Thus, for SQuAD, Table 3 additionally includes our end-to-end results on this 2016 Wikipedia dump (details in §A).

5.2 Baselines

The results are shown in Table 3. We report Exact Match (EM) of the gold answer on the test sets of the OpenQA versions of NQ, TQ, and SQuAD. We compare against both mainstream OpenQA approaches that rely on heuristic retrieval and recent systems that use learned vector-based retrieval.

Under mainstream OpenQA, we first report the purely BM25-retrieval-based results of Karpukhin et al. (2020). Next, we report GraphRetriever (Min et al., 2019b) and PathRetriever (Asai et al., 2020), both of which propose graph-based augmentation mechanisms to classical bag-of-words IR models (in particular, BM25 and TF-IDF, respectively). Unlike most other OpenQA systems included, PathRetriver uses a BERT-large reader (pretrained with whole-word masking or WWM) with 330M parameters.

Under learned-retrieval OpenQA, we report results of ORQA, REALM, and DPR, which are extractive OpenQA models like ColBERT-QA. We also include the concurrent work RAG by Lewis et al. (2020): unlike all other models included here, RAG is a *generative* model that uses a BART-large reader (406M parameters; Lewis et al. 2019).

Name	Retriever (# parameters)	Reader (# parameters)	NQ	TQ	SQuAD
Mainstream OpenQA					
BM25 + BERT Karpukhin et al. (2020)	BM25	BERT-base (110M)	32.6	52.4	38.1 / -
GraphRetriever Min et al. (2019b)	BM25 +graph-based retr.	BERT-base (110M)	34.5	56.0	-
PathRetriever Asai et al. (2020)	TFIDF +graph-based retr.	BERT-WWM (330M)	32.6	-	- / 56.5
Learned-Retrieval OpenQA					
ORQA Lee et al. (2019)	ORQA (2×110M)	BERT-base (110M)	33.3	45.0	20.2 / -
REALM Guu et al. (2020)	REALM (2×110M)	BERT-base (110M)	40.4	-	-
DPR (best variant) Karpukhin et al. (2020)	DPR (2×110M)	BERT-base (110M)	41.5	57.9	36.7 / -
RAG (generative) Lewis et al. (2020)	DPR (joint) (2×110M)	BART-large (406M)	44.5	56.1	-
Our Models					
ColBERT-QA (base)	ColBERT-QA$_3$ (110M)	BERT-base (110M)	<u>42.3</u>	<u>64.6</u>	<u>47.7</u> / <u>53.5</u>
ColBERT-QA (large)	ColBERT-QA$_3$ (110M)	BERT-WWM (330M)	**47.8**	**70.1**	**54.7 / 58.7**

Table 3: End-to-end OpenQA results, reporting exact-match (EM) on the open-domain test sets. For SQuAD, we report EM on the main 2018 Wikipedia dump (left) and on the 2016 Wikipedia dump (right). Underlined are the best results using only a BERT-base reader; bold is overall best.

5.3 Our Models

We use ColBERT-QA$_3$ as our retriever and test two reader models: (a) **ColBERT-QA (base)** with a BERT-base reader and (b) **ColBERT-QA (large)** with a BERT-large reader pretrained with whole-word masking (WWM). The former allows us to compare directly with approaches that use BERT-base. The latter allows us to evaluate ColBERT-QA's performance given a powerful reader that can better leverage its retrieval and permits comparing against the recent RAG model, which uses over 626M parameters, and PathRetriever, which also uses BERT-WWM. We train our readers separately from the retrievers, but use the retriever to collect triples for reader supervision.

As Table 3 shows, ColBERT-QA (base) outperforms the existing models that use a BERT-base reader on all three datasets. Moreover, it also outperforms the generative RAG on TQ, despite using far fewer parameters and being restricted to extractive reading. In fact, ColBERT-QA (base) already attains state-of-the-art extractive OpenQA performance on TQ and SQuAD.

Looking at ColBERT-QA (large) next, we observe that it also outperforms all of our baselines, including RAG, and establishes state-of-the-art performance for extractive OpenQA on all three datasets. These results reinforce the impact of improved, weakly-supervised retrieval as explored in §4. Importantly, ColBERT-QA's gains are consistent and substantial: every baseline—besides REALM, which does not evaluate on SQuAD or TQ—trails ColBERT-QA by several EM points on at least one dataset.

6 Conclusion

We proposed ColBERT-QA, an end-to-end system for OpenQA that employs the recent ColBERT retrieval model to improve both retriever modeling and supervision in OpenQA. To this end, we developed a simple yet effective training strategy that

progressively improves the quality of the OpenQA training data, even without hand-labeled evidence passages. Our results show that ColBERT is highly effective for OpenQA retrieval; with the proposed training paradigm, ColBERT-QA can improve retrieval quality over existing methods by over five Success@20 points and the resulting OpenQA pipeline attains state-of-the-art extractive OpenQA results across three popular datasets.

Acknowledgments: We would like to thank Ashwin Paranjape for valuable discussions and feedback. This research was supported in part by affiliate members and other supporters of the Stanford DAWN project—Ant Financial, Facebook, Google, Infosys, NEC, and VMware—as well as Cisco, SAP, a Stanford HAI Cloud Credit grant from AWS, and the NSF under CAREER grant CNS-1651570. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

Appendices

A Datasets

Natural Questions (NQ; Kwiatkowski et al. 2019) contains real questions submitted to Google by multiple searchers each, filtered such that a Wikipedia article is among the top-5 results. Answers are short spans in the Wikipedia article. Owing to its large scale and organic nature, this is the main dataset in our experiments.

SQuAD v1.1 (Rajpurkar et al., 2016b) is the popular QA dataset whose questions were written by crowdworkers over Wikipedia articles.

TriviaQA (TQ; Joshi et al. 2017b) is a set of trivia questions originally created by enthusiasts.

Each of the three datasets has approx. 79k training questions and 9k validation questions; NQ has 4k test questions and both of SQuAD and TQ have 11k test questions. For all three datasets, we use the train/validation splits released by Karpukhin et al. (2020). We note that the *test sets* are the same for all methods across both datasets.

B Implementation Details

Corpus & Preprocessing. Similar to related work, we use the full English Wikipedia, excluding tables, lists, and disambiguation pages. To facilitate comparisons with the state of the art, we use the preprocessed passages released by Karpukhin et al. (2020) for the 20 December 2018 dump.[7] We use this corpus unless otherwise stated. For the experiments that need Wikipedia 2016, we preprocess the 2016 dump released by Chen et al. (2017) in the same manner.

Following standard practice (e.g., as in Karpukhin et al. 2020 and Lee et al. 2019), we prepend the title of each Wikipedia page to all of its passages. For retrieval evaluation, we treat a passage as relevant if it contains the short answer string one or more times. Unlike ours, Karpukhin et al. (2020)'s open-source implementation does not take into account the title when evaluating whether a passage contains the answer or not. We found that this has a minimal impact on Success@k for various depths k. For instance, while ColBERT-QA$_3$ outperforms the best DPR results reported by the authors by 5.7–12.2 points in S@20, this delta would be reduced by only 0.1–0.6 points if we did not consider answer matches in passage titles.

Hyperparameters. For BM25 retrieval, we use the Anserini (Yang et al., 2018) toolkit with its default MSMARCO-tuned k_1 and b for passage search. We train our models using Python 3 and PyTorch 1.6, relying on the popular HuggingFace transformers library for the pretrained BERT models (Wolf et al., 2020). We apply PyTorch's built-in automatic mixed precision for faster training and inference. For training ColBERT models, we use a batch size of 64 triples (i.e., a question, a positive passage, and a negative passage each) with the default learning rate of 3×10^{-6} and default dimension $d = 128$ for each embedding. Our implementation starts from the ColBERT code.[8]

For our retrievers, we select the best checkpoint among $\{10k, 20k, ..., 50k\}$ via Success@20 on a sample of 1500 questions from each validation set. For ColBERT-QA$_2$ and ColBERT-QA$_3$, the last checkpoint was consistently the best-performing, suggesting more training could lead to even higher accuracy. For RGS, we set the depth k^- to 1000 passages (for sampling negatives) and sample the highest-ranked $t = 5$ positives from the top $k^+ = 50$ passages. If no positives exist in the top-k^+, we take the highest-ranked positive from the top-1000. For training the readers, we set $t = 3$ and $k^+ = k^- = 30$. For training our reader models, we use batches of 32 triples with a learning rate of 1×10^{-5}. For the base and large reader, we select the best checkpoint among $\{10k, 20k, ..., 50k\}$ and $\{10k, 20k\}$, respectively, via EM on the validation

[7] https://github.com/facebookresearch/DPR

[8] https://github.com/stanford-futuredata/ColBERT/

Slice	Size	S@1 C3	Delta of C3 over C1	Delta of C3 over DPR	Delta of C3 over BM25	Example (where C3 outperforms all baselines)
all	8757	53	+5	+8	+30	who sang i won't give up on you
misc.	1144	45	+2	+7	+21	poems that use the first letter of a word
what	1116	47	+5	+9	+25	what 's the major league baseball record for games won in a row
who	3651	59	+6	+9	+33	who wrote the song ruby sung by kenny rogers
where	712	53	+4	+5	+33	where does the united states keep an emergency stockpile of oil quizlet
when	1818	50	+7	+6	+34	when did wales last win the 6 nations
superlative	628	52	+7	+20	+35	who formed the highest social class of republican and early imperial rome

Table 4: Retrieval results by "slice" on the NQ validation set. For each slice, the table reports the number of queries (Size), Success@1 on NQ of ColBERT-QA3 (C3), and the improvement of this model over three baselines: ColBERT-QA1 (C1), DPR, and BM25. Each row contains an example query, for which C3 gets a correct passage and all three baselines fail. Rows are sorted by delta over BM25.

Ablation Name	Success@20 (test) NQ	TQ	SQ
SV Round 1 (end-to-end)	74.0	78.1	59.2
SV Round 2 (end-to-end)	76.6	78.4	**61.7**
SV Round 3 (end-to-end)	**77.2**	**79.4**	61.5

Table 5: The results of applying RGS to a *single-vector* ablation of the ColBERT model.

set, and tune the number of passages fed to the reader during inference in $\{5, 10, 15, 20, 30, 50\}$.

C Retrieval Analysis

Across a number of simple "slices" (Chen et al., 2019) of NQ-dev, Table 4 compares ColBERT-QA$_3$ (C3 for short) against ColBERT-QA$_1$ (C1), DPR, and BM25, revealing gains due to RGS (vs. C1), ColBERT (vs. DPR), and neural retrieval (vs. BM25). We use the Success@1 metric, stressing each model's precision. We find that C3's gains are stable and consistent across all models and slices, with gains of 2–7 points against C1, 7–20 points against DPR, and 21–35 against BM25. Interestingly, the largest gains against all three can be seen on the *superlative* slice, which contains over 600 queries that have comparative markers such as "best", "most", or "-est" (e.g., "largest").

D RGS for Single-Vector Retrieval

In this experiment, we apply RGS to a *single-vector* (SV) ablation of ColBERT. Table §5 shows the results, extending those in Table 1. Like DPR but unlike our ablation in Table 1, we test with end-to-end retrieval (and not by re-ranking BM25) and use in-batch negatives (but unlike DPR, only collect those on a per-GPU basis in a four-GPU setting). We observe very similar gain patterns to applying RGS over ColBERT, with +3.2, +1.3, and +2.3 point gains between the first and third rounds. The absolute results are considerably weaker than ColBERT-QA's, pointing to the value of late interaction, and are on average 1.0-point weaker than DPR's core setting, suggesting better tuning or implementation of single-vector modeling.

E Computational Cost & Latency

We conducted our experiments using servers with four Titan V (12GB) or 4–8 V100 (16/32GB) GPUs. We report running times for using four Titan V GPUs with our latest implementation. Each round of retriever training and validation requires 7–8 hours. Precomputing passage representations and indexing into FAISS (Johnson et al., 2019) requires approximately 6 hours. We apply our retrieval in batch mode: retrieval with all NQ train/dev/test questions takes 2–3 hours. Python scripts for pre- and post-processing (e.g., labeling as positive/negative, creating triples) add up to a few hours but leave much room for optimizations.

We also compare the (one-query) retrieval latency of ColBERT against our single-vector ablation, noting our Python-based research implementations are not optimized for production. Both models use the same underlying testbed, which treats single-vector representations as a special case of ColBERT. For questions in NQ-dev, retrieval latency is 71ms and 36ms per query for ColBERT and for the single-vector ablation, respectively.

References

Zeynep Akkalyoncu Yilmaz, Wei Yang, Haotian Zhang, and Jimmy Lin. 2019. Cross-domain modeling of sentence-level evidence for document retrieval. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3490–3496, Hong Kong, China. Association for Computational Linguistics.

Akari Asai, Kazuma Hashimoto, Hannaneh Hajishirzi, Richard Socher, and Caiming Xiong. 2020. Learning to retrieve reasoning paths over wikipedia graph for question answering. *ICLR 2020*.

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*, pages 1533–1544, Seattle, Washington, USA. Association for Computational Linguistics.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading wikipedia to answer open-domain questions. In *55th Annual Meeting of the Association for Computational Linguistics, ACL 2017*, pages 1870–1879. Association for Computational Linguistics (ACL).

Vincent S Chen, Sen Wu, Zhenzhen Weng, Alexander Ratner, and Christopher Ré. 2019. Slice-based learning: A programming model for residual learning in critical data slices. *Advances in neural information processing systems*, 32:9392.

Peter Clark and Oren Etzioni. 2016. My computer is an honor student – but how intelligent is it? Standardized tests as a measure of AI. *AI Magazine*, 37(1):5–12.

Mostafa Dehghani, Hamed Zamani, Aliaksei Severyn, Jaap Kamps, and W Bruce Croft. 2017. Neural ranking models with weak supervision. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 65–74.

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)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

David Ferrucci, Eric Brown, Jennifer Chu-Carroll, James Fan, David Gondek, Aditya A. Kalyanpur, Adam Lally, J. William Murdock, Eric Nyberg, John Prager, Nico Schlaefer, and Chris Welty. 2010. Building Watson: An overview of the deepQA project. *AI Magazine*, 31(3):59–79.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Retrieval Augmented Language Model Pre-Training. *ICML 2020*.

Lynette Hirschman, Marc Light, Eric Breck, and John D. Burger. 1999. Deep read: A reading comprehension system. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pages 325–332, College Park, Maryland, USA. Association for Computational Linguistics.

Mohit Iyyer, Jordan Boyd-Graber, Leonardo Claudino, Richard Socher, and Hal Daumé III. 2014. A neural network for factoid question answering over paragraphs. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 633–644, Doha, Qatar. Association for Computational Linguistics.

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

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017a. 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)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. 2017b. 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), pages 1601–1611.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense Passage Retrieval for Open-Domain Question Answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781.

Omar Khattab and Matei Zaharia. 2020. ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 39–48.

Bernhard Kratzwald and Stefan Feuerriegel. 2018. Adaptive document retrieval for deep question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 576–581, Brussels, Belgium. Association for Computational Linguistics.

Nate Kushman, Yoav Artzi, Luke Zettlemoyer, and Regina Barzilay. 2014. Learning to automatically solve algebra word problems. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 271–281, Baltimore, Maryland. Association for Computational Linguistics.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

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*, pages 6086–6096.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. ArXiv:1910.13461.

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. *NeurIPS 2020*.

Sean MacAvaney, Andrew Yates, Kai Hui, and Ophir Frieder. 2019. Content-Based Weak Supervision for Ad-Hoc Re-Ranking. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 993–996.

Sewon Min, Danqi Chen, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2019a. A discrete hard em approach for weakly supervised question answering. *arXiv preprint arXiv:1909.04849*.

Sewon Min, Danqi Chen, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2019b. Knowledge guided text retrieval and reading for open domain question answering. *arXiv preprint arXiv:1911.03868*.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. Ms marco: A human generated machine reading comprehension dataset. In *CoCo@ NIPS*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016a. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

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

Matthew Richardson, Christopher J.C. Burges, and Erin Renshaw. 2013. MCTest: A challenge

dataset for the open-domain machine comprehension of text. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 193–203, Seattle, Washington, USA. Association for Computational Linguistics.

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

Stephen E Robertson, Steve Walker, Susan Jones, Micheline M Hancock-Beaulieu, Mike Gatford, et al. 1995. Okapi at TREC-3. *NIST Special Publication*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Ellen M. Voorhees and Dawn M. Tice. 2000. Building a question answering test collection. In *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 200–207, New York, NY, USA. Association for Computing Machinery.

Zhiguo Wang, Patrick Ng, Xiaofei Ma, Ramesh Nallapati, and Bing Xiang. 2019. Multi-passage BERT: A Globally Normalized BERT Model for Open-domain Question Answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5881–5885.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Peilin Yang, Hui Fang, and Jimmy Lin. 2018. Anserini: Reproducible ranking baselines using lucene. *Journal of Data and Information Quality (JDIQ)*, 10(4):1–20.

Wei Yang, Yuqing Xie, Aileen Lin, Xingyu Li, Luchen Tan, Kun Xiong, Ming Li, and Jimmy Lin. 2019. End-to-end open-domain question answering with bertserini. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 72–77.

Yi Yang, Wen-tau Yih, and Christopher Meek. 2015. WikiQA: A challenge dataset for open-domain question answering. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2013–2018, Lisbon, Portugal. Association for Computational Linguistics.

Kaitao Zhang, Chenyan Xiong, Zhenghao Liu, and Zhiyuan Liu. 2020. Selective Weak Supervision for Neural Information Retrieval. In *Proceedings of The Web Conference 2020*, pages 474–485.