# Sparse Neural Retrieval Model for Efficient Cross-Domain Retrieval-Based Question Answering

**Kosuke Nishida**[†‡] **Naoki Yoshinaga**[§] **Kyosuke Nishida**[†]

[†]NTT Human Informatics Laboratories, NTT Corporation
[‡]The University of Tokyo
[§]Institute of Industrial Science, the University of Tokyo
[†]{kosuke.nishida.ap, kyosuke.nishida.rx}@hco.ntt.co.jp
[§]ynaga@iis.u-tokyo.ac.jp

## Abstract

Retrieval-based question answering (QA) is the problem of answering a question from a large corpus. Here, we tackle cross-domain retrieval QA (ReQA); a sentence-level retrieval for QA in which a model is trained in a general corpus and used in private settings to deal with private documents (*e.g.*, personal e-mails and confidential documents). This task requires a model that has domain robustness and works in real-time on resource-constrained devices (*e.g.*, mobiles). The current accurate neural ReQA models, however, require GPUs for real-time processing and work poorly on domains other than the domains used to train the models. In this study, aiming to address the two challenges, we improve the out-of-domain performance of the index-based efficient sparse neural ReQA model. The proposed model replaces the embedding matrix with those trained in the target domain to improve the performance on the user's corpus and has a learnable sparsity parameter to tune the sparsity of the output vectors. To evaluate the cross-domain ReQA, we revise the MultiReQA dataset to estimate the model performance correctly. Experimental results showed that SPARC improves both in-domain and out-of-domain performance by considering the token importance in the target corpus.

## 1 Introduction

There is an increasing interest in answering user's questions written in natural language by leveraging documents stored on the user's own device (*e.g.,* project documents). Hence, retrieval-based question answering (QA), which is the problem of answering a question via a large corpus (Voorhees and Tice, 2000; Chen et al., 2017), has attracted attention. Recently, Ahmad et al. (2019) proposed retrieval QA (ReQA) as one of retrieval-based QA tasks, in which the model does not generate or extract a concise answer but instead retrieves a sentence mentioning the answer directly from a corpus. Although retrieval-based QA has been studied in various settings that differ in the granularity of outputs (namely, documents, paragraphs, and ultimately sentences), sentence-level retrieval becomes popular since it subsumes the other settings and is most useful.

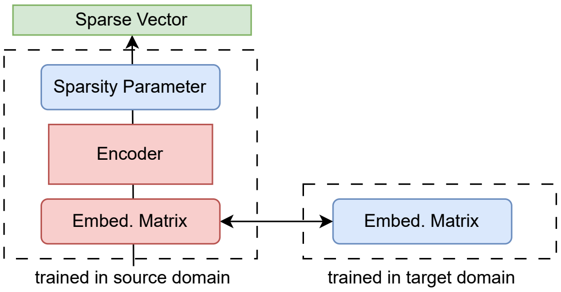

Figure 1: Overview of our model, which is based on a sparse neural retrieval model that encodes a text as a sparse vector. We introduce a learnable sparsity parameter and replacement of the embedding matrix with one trained in the target domain.

In this paper, we focus on the cross-domain application of sentence-level retrieval QA, (cross-domain ReQA), in which models are evaluated on out-of-domain datasets, by developing quick and lightweight unsupervised adaptation of a trained ReQA model. This task comprises a realistic setting in which users must process their corpora, which may include personal information or internal company information, on their own devices because of security and privacy concerns. Hence, we assume that a service provider trains a QA model on GPUs without any information on the target domain, after which users deploy the model for inference. This task involves desktop search. Such cross-domain ReQA faces two challenges. First, the model must work in a user's computer environment. Second, there are discrepancies in the distribution of tokens in the retrieval corpus between the training and inference phase.

Previous studies tackled the first challenge by proposing models that encode sentences in a corpus via scores for all tokens in a vocabulary and then use part of them as a sparse representation. These sparse neural retrieval models work efficiently as an inverted index on a mono-CPU when using search engines such as Lucene (McCandless et al., 2010). Unlike bag-of-words models (*e.g.*, BM25 (Robertson and Zaragoza, 2009)), these models consider implicit tokens that do not occur in a sentence but are related to the sentence, such as synonyms. Moreover, unlike dense retrieval using a sentence-level inner product, these models represent the interaction between all the tokens in a query and in a sentence as a sparse representation. Sparse neural retrieval models are also suitable for the cross-domain ReQA. Whereas the model training requires GPUs, a CPU is sufficient at the inference phase for the pre-computation of sparse representations of the sentences in a user's corpus and real-time retrieval with the pre-computed inverted index. In contrast, the second challenge has remained.

Here, to overcome both challenges in cross-domain ReQA, we propose a sparse neural retrieval model for cross-domain retrieval (SPARC). Specifically, we introduce two techniques to the sparse neural retrieval model, as illustrated in Figure 1. First, we replace the embedding matrix of the model trained in the source domain with one in the target domain. Because both sparse neural retrieval models and language models are based on the dot product of hidden states and token embeddings, we can share the embedding matrix trained on the language modeling task in the target domain to capture the statistics of the corpus in the target domain. By restricting the trainable parameters and training steps, we can perform the training of the embedding matrix in a user's computer environment. Second, we integrate a learnable parameter that makes the model's output states sparse. This approach is novel because the previous sparse neural models achieved sparsity via top-K filtering at only inference time or an unstable regularization term that is manually tuned.

To evaluate the model on cross-domain ReQA, we focus on the MultiReQA dataset (Guo et al., 2021; Zhao et al., 2021), which involves a sentence-level retrieval-based QA task in multiple domains.

The contributions of this paper are threefold:

- We propose SPARC, which seeks to overcome the two challenges in cross-domain ReQA by introducing a learnable parameter to induce sparsity and replacing the embedding matrix.
- We observed that MultiReQA required further pre-processing to estimate the model performance correctly, and we thus introduce a new pre-processing method.
- Our experimental results showed that SPARC improves the retrieval performance on MultiReQA in both in-domain and out-of-domain settings by overcoming the two challenges.

## 2 Problem Setting

We define the problem of cross-domain retrieval QA. We have source and target domains, respectively denoted as $D_s$ and $D_t$. Given a sentence set $S_d$ in a domain $d \in D_t$, the goal is to retrieve a sentence that answers a question $q$ from $S_d$. First, a service provider trains a model with the source domain datasets on GPUs. The provider does not use any data in the target domain for training. Then, a user pre-computes the sentences in the target domains on only a CPU. Lastly, the user runs a real-time QA system on a CPU. This task is not constrained to domain-level adaptation but involves personalization based on a given corpus.

## 3 Related Work

### 3.1 Retrieval-Based QA

Open-domain QA is one of the most common problems in retrieval-based QA (Voorhees and Tice, 2000). The main approach for this task is a two-stage approach comprising fast retrieval of passages from a corpus and fine-grained extraction of an answer from those passages (Chen et al., 2017; Wang et al., 2018; Nishida et al., 2018).

The first stage of fast retrieval in open-domain QA can be applied to the ReQA task. For such fast retrieval, classic non-neural inverted-index methods are used, such as TF-IDF (Jones, 1972) and BM25 (Robertson and Zaragoza, 2009). Recently, approximate nearest neighbor search (ANNS), which uses dense vectors as keys and queries (Shrivastava and Li, 2014; Johnson et al., 2017), was introduced for fast retrieval in open-domain QA (Xiong et al., 2021; Karpukhin et al., 2020; Lee et al., 2019; Qu et al., 2021). Such methods are called dual-encoder models because the question and passages are encoded in dense vectors. The main advantage of dual-encoder models is that they can benefit from pre-trained language models

(PLMs) (Devlin et al., 2019). However, ANNS methods often assume multi-threaded computing with multiple CPUs or even GPUs, whereas traditional sparse models work well on a mono-CPU and are computationally efficient (Lassance and Clinchant, 2022). In addition, dual-encoder models require encoding of the question with PLM at the inference phase, which often takes a larger computation time than the retrieval.

Two sparse neural retrieval models, SPARTA (Zhao et al., 2021) and SPLADE (Formal et al., 2021b,a, 2022; Lassance and Clinchant, 2022), combine both lines of work. They obtain sparse representations comprising the scores of the relevant tokens to each sentence, and they store the representations as an inverted index for fast retrieval while using a PLM. Unlike dual-encoder models, they can model the interaction between the tokens in a query and those in a sentence beyond the dot product of dense vectors (Zhao et al., 2021). These models can be viewed as performing implicit document expansion. Lassance and Clinchant (2022) reported that these models outperformed previous sparse models using the document expansion, such as DocT5Query (Nogueira et al., 2019) and DeepImpact (Mallia et al., 2021). Dai and Callan (2020) and Bai et al. (2020) also used PLM to calculate the importance of terms appearing in the input and expanded context. Here, we extend these sparse neural retrieval models to the cross-domain setting.

### 3.2 Cross-Domain QA

Apart from ReQA, the MRQA dataset (Fisch et al., 2019) focuses on cross-domain QA for evaluation of a QA model's generalization capability on out-of-domain data. This dataset includes six source domain datasets (Rajpurkar et al., 2016; Trischler et al., 2017; Joshi et al., 2017; Dunn et al., 2017; Yang et al., 2018; Kwiatkowski et al., 2019) and six target domain datasets (Tsatsaronis et al., 2015; Dua et al., 2019; Saha et al., 2018; Lai et al., 2017; Levy et al., 2017; Kembhavi et al., 2017). Guo et al. (2021) and Zhao et al. (2021) simultaneously developed the MultiReQA dataset for cross-domain ReQA by splitting all passages in the dataset into the sentences and merging them into a sentence set.

A major approach in cross-domain QA is to generate questions as pseudo training datasets in the target domain (Golub et al., 2017; Shakeri et al., 2020; Luo et al., 2022). However, question generation and training of a PLM on the pseudo dataset is too computationally expensive to run in user environments. Also, as in domain adaptation of the language models in general natural language processing tasks (Gururangan et al., 2020), we can apply masked language models in the target domain for question answering (Nishida et al., 2020). However, that approach assumes that the target domain is determined in the training phase; thus, it is not applicable in a domain-agnostic setting.

## 4 Baseline Models

We propose SPARC to enable effective execution of a pre-trained, off-the-shelf ReQA model in a user environment. In this paper, we implemented SPARC by extending SPARTA. However, the technique proposed for SPARC can be applied to other sparse neural retrieval models such as SPLADE.

### 4.1 SPARTA

Here, we introduce the SPARTA model (Zhao et al., 2021). Let $\boldsymbol{H}$ be a sentence encoder, and let $\boldsymbol{H}(s) \in \mathbb{R}^{l \times d}$ be contextualized token embeddings of a sentence $s$, where $l$ is the token length and $d$ is the hidden size. $f$ denotes the following token-to-sequence (tok2seq) matching score function between $s$ and a token $v$ in a vocabulary $V$:

$$f(v, s) = \log(\mathrm{ReLU}(\max_i \boldsymbol{H}(s)_i^\top \boldsymbol{e}_v) + 1), \quad (1)$$

where $\boldsymbol{e}_v$ is the 0-th layer embedding of token $v$ in the encoder $\boldsymbol{H}$, and $\boldsymbol{H}(s)_i$ is the i-th token representation of $\boldsymbol{H}(s)$.

Here, the tok2seq matching score $f(v, s)$ can be decomposed into three functions. First, $\{\boldsymbol{H}(s)_i^\top \boldsymbol{e}_v\}_{1 \le i \le l}$ indicates how strongly the input tokens are related to $v$. Note that this operation is similar to the masked language modeling (Devlin et al., 2019). Second, the max operation over the input tokens outputs the highest scores. Third, $\log(\mathrm{ReLU}(\cdot) + 1)$ outputs a non-negative score, which is essential for a sparse neural retrieval model because we expect that most scores are close to zero.

To retrieve a sentence, we calculate the sequence-to-sequence (seq2seq) matching score between a question $q$ and a sentence $s$:

$$g(q, s; f) = \sum_{v \in q} f(v, s). \quad (2)$$

At the training phase, following the standard setting, we sample questions and the corresponding ground-truth and negative sentences from the source domain. We use the cross-entropy loss to distinguish ground-truth sentence and in-batch negatives.

At the inference phase, we can pre-compute the tok2seq matching scores $\{f(v, s)\}_{v \in V}$ for all sentences $s \in S_d$ because $f$ does not depend on the contextualized information of $q$. Then, we store the top-$K$ tokens and their scores as an inverted index. Finally, we efficiently retrieve a sentence with respect to $g(q, s; f)$ on a CPU by using the pre-computed inverted index.

### 4.2 SPLADE

SPLADE (Formal et al., 2021b,a, 2022; Lassance and Clinchant, 2022) is another major sparse neural retrieval model. It computes the tok2seq matching score $f(v, q)$ of a question $q$ in addition to the tok2seq matching score $f(v, s)$ of a sentence $s$. However, $f(v, q)$ cannot be pre-computed, and SPLADE is thus not suitable for our task setting of running a real-time QA system in a user's private environment with a single CPU for inference.

Lassance and Clinchant (2022) proposed an efficient variant of SPLADE that uses BERT-tiny (Turc et al., 2019) as the question encoder. However, our pilot experiments showed that using BERT-tiny degraded the performance. The MultiReQA dataset is small than the MSMARCO dataset (more than 1M questions) (Bajaj et al., 2016), which the authors used, and we consider that the small BERT-tiny model (total 4.4M parameters, the number of layers is two, the hidden size is 128) requires a large dataset to learn retrieval capability. Formal et al. (2021a) proposed another efficient variant, SPLADE-doc, which removes the question encoder from the model. Its seq2seq matching score is equivalent to that of SPARTA.

Note also SPLADE models used the following FLOPS regularization (Paria et al., 2020):

$$R_{\text{FLOPS}} = \sum_{v \in V} \left( \frac{1}{B} \sum_{1 \leq b \leq B} f(v, s_b) \right)^2 , \quad (3)$$

where $B$ is the batch size, and $s_b$ is the b-th sentence in a batch. The use of a FLOPS regularizer in a sparse neural retrieval model has two problems. First, because the $l_2$ norm reduces the value but does not cause it to reach zero, the FLOPS regularizer assumes that any token in a vocabulary is related to one of the sentences in a batch. This assumption is strong because of the presence of domain- and language-specific tokens. Second, the FLOPS regularizer requires the scheduling of a strength hyperparameter. Formal et al. (2021a) quadratically increased it until 50k steps and then kept it as constant. However, we could not reproduce those results in our experiments on the MultiReQA dataset.

As a result, we used the SPARTA model, which is equivalent to SPLADE-doc without FLOPS regularization, as a baseline. Other SPLADE models are not comparable with respect to computational efficiency, or their training were unstable.

## 5 SPARC

Here, to develop SPARC, we introduce two new techniques into SPARTA: replacement of the embedding matrix with a domain-specific one, and a learnable sparsity parameter in the tok2seq score $f(v, s)$.

### 5.1 Embedding Matrix Replacement

Generally, to obtain knowledge in a target domain, the common approach is to apply a masked language modeling (MLM) task (Devlin et al., 2019) in the target domain before fine-tuning in the source domain (Gururangan et al., 2020). This is because the model suffers from the catastrophic forgetting of the capability to solve a downstream task if it is first trained with the downstream task in the source domain and then trained with the MLM in the target domain. However, we are agnostic of the target domain in the fine-tuning phase and thus cannot follow this strategy.

Instead, we propose to replace the embedding matrix with one that is trained with the MLM task in the target domain after fine-tuning. This fine-tuning step uses the sentence set of the target domain just before the pre-computing of invert index and does not use any information of questions. Therefore, this step matches our problem setting. First, we fine-tune the model in the source domain while fixing the embedding matrix. Second, to capture the knowledge in the target domain into the embedding matrix, we train the original PLM in the target domain with the MLM task while fixing all parameters but the embedding matrix. Finally, we replace the fine-tuned model's embedding matrix with one trained with the MLM. A sparse neural retrieval model solves a task that is similar to the MLM task, because it maximizes the dot product of the contextualized embeddings $H(s)$ and the

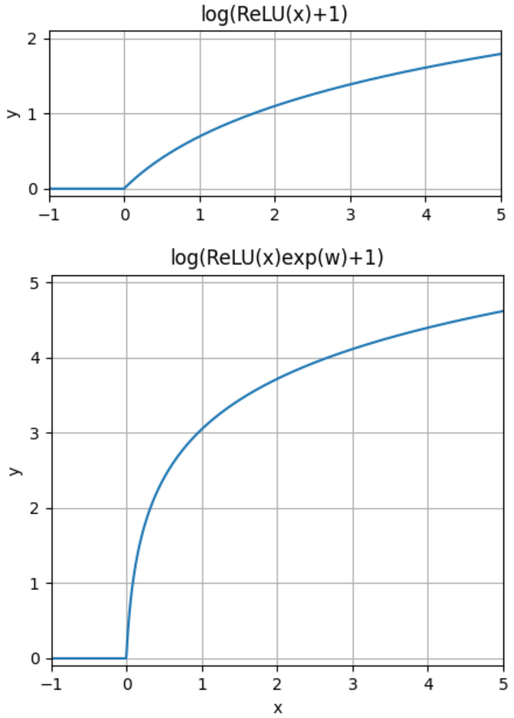

Figure 2: Graphs of $y = \log(\mathrm{ReLU}(x) + 1)$ and $y = \log(\mathrm{ReLU}(x) \exp(w) + 1)$, with $\exp(w) = 20$.

static embedding of related tokens $e_v$. Hence, we can replace the embedding matrix without performance degradation. By restricting the learnable parameters and the step size, we confirmed that the training of the embedding matrix in the target domain works on a CPU.

### 5.2 Learnable Sparsity Parameter

In our pilot experiments, we observed a problem with the tok2sec matching score $f(v, s)$ (Eq. 1). The advantage of using $\log(\mathrm{ReLU}(x) + 1)$ is that it sets the score to zero for negative $x$. Also, it is natural to use the log function, because Eq. 2 performs a sum operation over $f(v, s)$, and then the scores are fed to the loss function as logits. However, the log activation causes small gradients, which results in small differences among token scores, as shown in the top row of Figure 2. Therefore, to obtain a sufficiently large difference in the seq2seq matching score $g(q, s; f)$ among sentences, many tokens must have positive values.

Therefore, we introduce a learnable parameter $w$ as follows:

$$f(v, s) = \log(\mathrm{ReLU}(\max_i \boldsymbol{H}(s)_i^\top \boldsymbol{e}_v)\exp(w) + 1). \tag{4}$$

We have found that the value of $\exp(w)$ is typically over 20 after training, which shows the importance of the learnable sparsity parameter. As shown in the bottom row of Figure 2, this formulation causes a large difference among token scores; moreover, the gradient is large in the neighborhood of zero, which causes sparsity. As a result, SPARC assigns large values only to important tokens, and zero to most tokens in Eq. 1.

## 6 Experiments

### 6.1 Dataset and Metrics

**Dataset.** We used MultiReQA (Guo et al., 2021; Zhao et al., 2021) for these experiments. MultiReQA was created from a cross-domain QA dataset, MRQA, by splitting passages into sentences. Following Guo et al. (2021), we removed four domains from the 12 domains in the original MRQA dataset because most of the questions in those domains were unclear when detached from their original passage.

**Our pre-processing of MultiReQA.** However, we found that the previously published MultiReQA underestimated the model performance. For instance, the performance in the biomedical domain was reported to be less than 20 points in MRR@10. We observed that identical questions (*e.g.*, “Q: What type of enzyme is peroxiredoxin 2 (PRDX2)?”, “A: antioxidant”) tied to different passages caused the underestimation. Because these are factoid questions, the answers in all tied passages should be allowed. Hence, we solved this problem by merging identical questions and their answers into an instance. An answer was sampled at the training phase, and all answers were allowed at the inference phase. Moreover, we removed three domains, as follows. HotpotQA (Yang et al., 2018) requires multi-hop reasoning, and its questions thus cannot be answered by a single sentence. TextBookQA (Kembhavi et al., 2017) requires an understanding of visual elements to answer. Relation Extraction (Levy et al., 2017) is a synthetic dataset for relation extraction and the scores on that dataset are saturated.

As a result, our version of MultiReQA had five domains. Table 1 lists the dataset statistics. We used the training and evaluation splits of SQuAD as our training dataset and development dataset. Then, we used the evaluation splits of the other datasets as our test dataset.

**Metrics.** Here, we report the median of MRR and Recall@1 scores over three runs.

### 6.2 Implementation

Following Zhao et al. (2021), to encode a sentence, we used the other sentences in the same passage as the surrounding context by concatenating the original sentence and the surrounding context. That is, an input of the encoder $s$ is '<Original Sentence> `[SEP]` <Other Sentences>'. This was truncated to the first 256 tokens. We used the segment ids to distinguish the sentence and the context.

To train the model with the cross-entropy loss, we extracted hard-negative sentences for a question. First, we randomly extracted a sentence from a passage that included the ground-truth sentence except for the ground-truth sentence. Second, we randomly extracted a sentence from the top-100 sentences in the sentence set $S_d$, whose scores were calculated with BM25 (Robertson and Zaragoza, 2009). We then removed sentences including the question's original answer. Thus, in a batch of size $B$, a question had one ground-truth sentence, two hard-negative sentences, and $3(B-1)$ negative sentences.

We used the pre-trained DistilBERT-base-uncased model (66M parameters) (Sanh et al., 2019) as the encoder. Note that we could pre-compute the sentence score $\{f(s,v)\}_{v \in V}$ in 0.9s with DistilBERT on a MacBookAir (M1, 16GB memory). For implementation, we used PyTorch (ver. 1.12.1) (Paszke et al., 2019)[1] and transformers (ver. 4.21.1) (Wolf et al., 2020).[2] The model was trained for 20 epochs without early stopping. The Adam optimizer (Kingma and Ba, 2015) was used with a learning rate of 1e-5, a linear learning-rate decay, and 500 warmup steps. The learning rate for the sparsity parameter was 1e-3 to obtain a sufficiently large value. We used four NVIDIA Quadro RTX 8000 (48GB) GPUs. The batch size was 96. Gradient Cache (Luyu Gao and Callan, 2021) was used to backpropagate the gradients of in-batch negatives in multiple GPUs. Following Zhao et al. (2021), we set $K = 2000$. Finally, the number of steps for pre-training in the target domain for embedding matrix replacement was 500. The batch size is 32. We used the Adam optimizer with a learning rate of 5e-5. The training took less than five hours.

### 6.3 Compared Models

**Unsupervised sparse model.** We used **BM25** (Robertson and Zaragoza, 2009).

**Sparse neural models.** We used **SPARTA** (Zhao et al., 2021) as the baseline and implemented **SPARC** on the basis of SPARTA. The inference was implemented with SciPy (Virtanen et al., 2020).

**Dense model.** We used Sentence-BERT (**S-BERT**)[3] (Reimers and Gurevych, 2019) as the dual-encoder baseline. Although several dense models, such as DPR (Karpukhin et al., 2020), have the same architecture as S-BERT, the main differences among them are the training data and the selection of negative samples. We selected S-BERT as the initial point for a fair comparison in terms of the training data because its pre-training does not use any QA datasets, and we then trained the model in our setting. The inference was implemented with faiss (Johnson et al., 2017).

### 6.4 Results and Discussion

**Does SPARC improve the retrieval performance?** Table 1 summarizes the main results. SPARC outperformed SPARTA except on the biomedical domain. Even without the embedding replacement, SPARC improved the performance. We conclude that the learnable sparsity parameter improved the performance by emphasizing the scores of important tokens and compressing the scores of unimportant ones.

In particular, we observed that the embedding matrix replacement improved the performance on TriviaQA, SearchQA and BioASQ. In contrast, it decreased the performance on SQuAD and Natural Questions because SQuAD was used for training, while the domain of Natural Questions entailed Wikipedia with a few HTML tags such as '`<p>`' and '`<li>`'. The embedding matrix replacement reduced discrepancies between the sentence sets in the training and inference phases. Therefore, it would be effective for users to develop retrieval systems from their own documents in private environments.

Regarding the biomedical domain, SPARTA outperformed SPARC. SPARC increases the sparsity and improved the performance on the other domains. However, the biomedical domain has a lot

[1] `https://pytorch.org/`

[2] `https://github.com/huggingface/transformers`

[3] `https://huggingface.co/sentence-transformers/distilbert-base-nli-mean-tokens`

|  | SQuAD | Natural Questions | TriviaQA | SearchQA | BioASQ | Average |
|---|---|---|---|---|---|---|
| Domain | Wikipedia | HTML Wikipedia | Web snippets | Web snippets | Science articles | — |
| Queries | Crowdsourced | Search logs | Trivia | Jeopardy | Domain experts | — |
| $\lvert S_d \rvert$ for Train | 95658 | (448354) | (1893673) | (3163800) | — | — |
| $\lvert q \rvert$ for Train | 86355 | (104065) | (61687) | (117219) | — | — |
| $\lvert S_d \rvert$ for Eval. | 10641 | 22117 | 238338 | 454835 | 14157 | — |
| $\lvert q \rvert$ for Eval. | 10477 | 4176 | 7784 | 16978 | 223 | — |
| Unsupervised sparse model |  |  |  |  |  |  |
| BM25 | 55.44/48.15 | 27.42/20.45 | 32.77/24.06 | 48.61/33.20 | 41.76/31.70 | 45.98/36.90 |
| DistilBERT trained on SQuAD |  |  |  |  |  |  |
| S-BERT | 67.30/57.13 | 27.76/19.19 | 39.53/27.51 | 43.74/29.90 | 51.95/41.07 | 48.76/38.21 |
| SPARTA | 84.31/77.47 | 42.92/32.70 | 53.71/40.78 | 55.52/37.94 | **75.84/64.73** | 62.46/50.72 |
| SPARC - Emb.Rep. | **84.97/78.12** | **45.09/34.85** | 55.11/42.01 | 57.95/40.28 | 74.62/62.95 | 63.55/51.64 |
| SPARC | 84.96/78.02 | 44.97/34.62 | **55.42/42.57** | **58.15/40.53** | 75.10/63.39 | **63.72/51.83** |

Table 1: Data statistics and the results of cross-domain evaluation. The neural models were trained on the SQuAD dataset, and the datasets in parentheses were not used for training. The scores indicate MRR/Recall@1.

|  | SQuAD | Natural Questions | TriviaQA | SearchQA | BioASQ | Average | Std. |
|---|---|---|---|---|---|---|---|
| SPARTA | 5316 | 3923 | 4167 | 5353 | 5805 | 4912.8 | 733.3 |
| SPARC | 1246 | 1120 | 1108 | 1189 | 1087 | 1150.0 | 58.9 |

Table 2: Numbers of non-zero tokens in the sentence vectors generated by each method, reported as the median among all sentence vectors.

of domain-specific words. We thus consider that SPARC compressed the scores of important tokens comprising domain-specific words. Although sparsity is effective for computational efficiency and performance in general domains, there is a trade-off between sparsity and performance in domains involving domain-specific words.

SPARC outperformed the sparse non-neural model BM25 and the dense neural model S-BERT. This was because it better captured contextual information and interaction among tokens as compared to BM25 based on the token frequency, and S-BERT based on a dense representation.

**Does the learnable sparsity parameter make the vectors sparse?** To investigate the sparsity of the sparse neural retrieval models, we counted the number of non-zero tokens in the sentence vectors generated by each method. Table 2 lists the results. Although the learnable sparsity parameter in Eq. 4 tuned only the scale of the dot product of $\boldsymbol{H}(s)_i$ and $\boldsymbol{e}_v$, SPARC increased the sparsity of the sentence vectors. Also, we found that the standard deviation was smaller for SPARC than for SPARTA. This constant sparsity is one of the pieces of evidence of the SPARC's robustness in out-of-domains. However, we found a large number of non-zero tokens for SPARTA on the BioASQ dataset. Because the token distribution and the sense of tokens in the biomedical domain were different from the other domains, a large number of nonzero tokens avoid removing important tokens in the biomedical domain.

Note also that SPARC outperformed SPARTA in terms of the MRR and Recall@1, even though its vectors were sparser than those of SPARTA.

**Why does the learnable sparsity parameter make the vectors sparse by tuning the scale?** Next, we investigated the reason for the sparsity. Figures 3 and 4 show histograms of the values of $\max_i(\boldsymbol{H}(s)_i^\top \boldsymbol{e}_v)\exp(w)$ for all sentence vectors with SPARTA (without $\exp(w)$) and SPARC, respectively. As seen in these figures, the SPARC values were larger than those with SPARTA because of the learnable sparsity parameter. Moreover, we confirmed that the number of positive values, which were not removed by the $\mathrm{ReLU}$ function, was smaller with SPARC than with SPARTA. We conclude that, because of the values' large scale, the final score (Eq. 2) could distinguish sentence vectors with small numbers of tokens. SPARC thus obtained sparse vectors by tuning the scale.

**What tokens are removed by the learnable sparsity parameter?** We next investigated what tokens were removed by the learnable sparsity param-

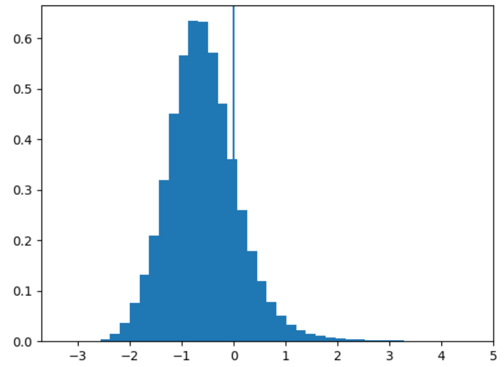

Figure 3: Histogram of the values for all tokens in the vocabulary of all sentence vectors with SPARTA. The vertical axis is normalized to the proportion. That is, the values are divided by $|S_d| \times |V|$.

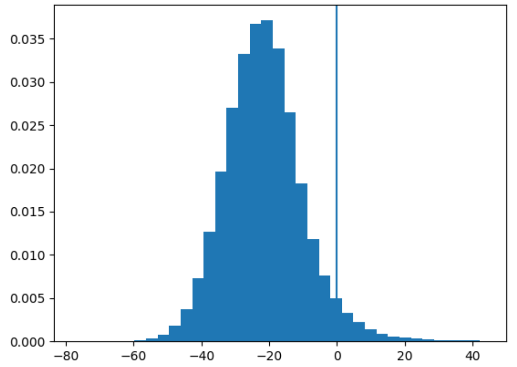

Figure 4: Histogram of the values for all tokens in the vocabulary of all sentence vectors with SPARC.

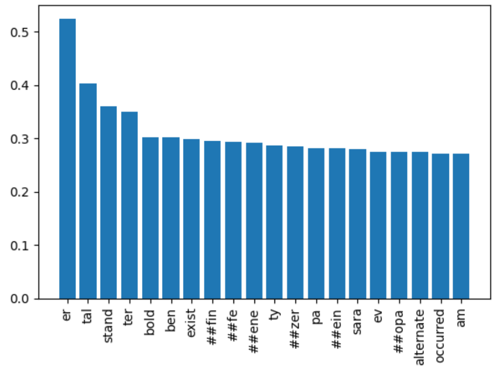

Figure 5: Histogram of the number of sentences in which each token had a non-zero value with SPARTA but zero with SPARC. The vertical axis is normalized to the proportion. That is, the number is divided by $|S_d|$.

| | MRR | Recall@1 | Latency[ms] |
|---|---|---|---|
| SPARC ($K = 100$) | 59.48 | 69.68 | 0.69 |
| SPARC ($K = 500$) | 83.89 | 77.96 | 0.77 |
| SPARC ($K = 1000$) | 84.82 | 77.99 | 0.77 |
| SPARC ($K = 2000$) | 84.96 | 78.02 | 0.77 |
| S-BERT ($d = 768$) | 67.30 | 57.13 | 41.5 |

Table 3: Retrieval performance and latency. We retrieved the top-1000 sentences to calculate the MRR and latency. The computational cost of SPARTA is the same as SPARC except for the sparsity.

eter. Specifically, we counted the number of sentences in which each token had a positive score with SPARTA but zero with SPARC. Figure 5 shows a histogram of the top-20 tokens, with the vertical axis normalized to the proportion with respect to all sentences.

First, we observed that most of the top 20 tokens were subwords. We expect that SPARC learned not to assign scores to all the tokens in a word, which was effective under the constraint of sparsity.

Second, words indicating presence (e.g., “stand”, “exist”, “occured”, “am”) were removed. We assume that SPARC reduces the scores of such words because they infrequently affected the meaning of a sentence.

**How fast does SPARC answer a query?** Finally, Table 3 lists the performance and the latency on SQuAD with DistilBERT on a MacBook Air. We observed that SPARC outperformed S-BERT in terms of both retrieval performance and latency. Although S-BERT is not a state-of-the-art model and more sophisticated models such as Poly-Encoder (Humeau et al., 2020) and ColBERT (Khattab and Zaharia, 2020) exist, they have a larger computational cost than the S-BERT model. Therefore, we confirmed that the sparse neural retrieval models have an advantage over the dense models for the private QA systems targeted here.

## 7 Conclusion

In this paper, we focused on cross-domain retrieval question answering (ReQA), which is motivated by a realistic setting in which users do not have rich computational resources and cannot submit their corpora because of privacy concerns. For this purpose, we proposed the SPARC model by incorporating the replacement of the embedding matrix and a learnable sparsity parameter in a sparse neural retrieval model. We found that SPARC outperformed the neural retrieval baselines in both in-domain and out-of-domain settings. In addition to improving the retrieval performance, SPARC increased the sparsity of the sentence representation.

SPARC can be used for the personalization of a QA model, which requires adaptation to a very large number of target domains (*i.e.*, users). We believe that this work will contribute to the implementation of efficient, secure private QA systems.

## Acknowledgements

This work (second author) was partially supported by the special fund of Institute of Industrial Science, The University of Tokyo, and by JSPS KAKENHI Grant Number JP21H03494.

## References

Amin Ahmad, Noah Constant, Yinfei Yang, and Daniel Cer. 2019. ReQA: An evaluation for end-to-end answer retrieval models. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 137–146. Association for Computational Linguistics.

Yang Bai, Xiaoguang Li, Gang Wang, Chaoliang Zhang, Lifeng Shang, Jun Xu, Zhaowei Wang, Fangshan Wang, and Qun Liu. 2020. Sparterm: Learning term-based sparse representation for fast text retrieval. *CoRR*, abs/2010.00768.

Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, et al. 2016. Ms marco: A human generated machine reading comprehension dataset. *CoRR*, abs/1611.09268.

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)*, pages 1870–1879. Association for Computational Linguistics.

Zhuyun Dai and Jamie Callan. 2020. Context-aware document term weighting for ad-hoc search. In *Proceedings of The Web Conference 2020*, WWW '20, page 1897–1907. Association for Computing Machinery.

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. Association for Computational Linguistics.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. 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 2368–2378. Association for Computational Linguistics.

Matthew Dunn, Levent Sagun, Mike Higgins, V. Ugur Güney, Volkan Cirik, and Kyunghyun Cho. 2017. Searchqa: A new q&a dataset augmented with context from a search engine. *CoRR*, abs/1704.05179.

Adam Fisch, Alon Talmor, Robin Jia, Minjoon Seo, Eunsol Choi, and Danqi Chen. 2019. MRQA 2019 shared task: Evaluating generalization in reading comprehension. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 1–13. Association for Computational Linguistics.

Thibault Formal, Carlos Lassance, Benjamin Piwowarski, and Stéphane Clinchant. 2022. From distillation to hard negative sampling: Making sparse neural ir models more effective. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '22, page 2353–2359. Association for Computing Machinery.

Thibault Formal, Carlos Lassance, Benjamin Piwowarski, and Stéphane Clinchant. 2021a. Splade v2: Sparse lexical and expansion model for information retrieval. *CoRR*, abs/2109.10086.

Thibault Formal, Benjamin Piwowarski, and Stéphane Clinchant. 2021b. Splade: Sparse lexical and expansion model for first stage ranking. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, page 2288–2292. Association for Computing Machinery.

David Golub, Po-Sen Huang, Xiaodong He, and Li Deng. 2017. Two-stage synthesis networks for transfer learning in machine comprehension. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 835–844, Copenhagen, Denmark. Association for Computational Linguistics.

Mandy Guo, Yinfei Yang, Daniel Cer, Qinlan Shen, and Noah Constant. 2021. MultiReQA: A cross-domain evaluation forRetrieval question answering models. In *Proceedings of the Second Workshop on Domain Adaptation for NLP*, pages 94–104. Association for Computational Linguistics.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online. Association for Computational Linguistics.

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

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2017. Billion-scale similarity search with gpus. *CoRR*, abs/1702.08734.

Karen Spärck Jones. 1972. A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, 28(1):11–21.

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)*, pages 1601–1611. Association for Computational Linguistics.

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. Association for Computational Linguistics.

Aniruddha Kembhavi, Minjoon Seo, Dustin Schwenk, Jonghyun Choi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Are you smarter than a sixth grader? textbook question answering for multimodal machine comprehension. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

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*, SIGIR '20, page 39–48. Association for Computing Machinery.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations (ICLR)*.

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*, 7:452–466.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. 2017. RACE: Large-scale ReAding comprehension dataset from examinations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 785–794. Association for Computational Linguistics.

Carlos Lassance and Stéphane Clinchant. 2022. An efficiency study for splade models. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '22, page 2220–2226. Association for Computing Machinery.

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. Association for Computational Linguistics.

Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. 2017. Zero-shot relation extraction via reading comprehension. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 333–342. Association for Computational Linguistics.

Hongyin Luo, Shang-Wen Li, Mingye Gao, Seunghak Yu, and James Glass. 2022. Cooperative self-training of machine reading comprehension. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 244–257, Seattle, United States. Association for Computational Linguistics.

Jiawei Han Luyu Gao, Yunyi Zhang and Jamie Callan. 2021. Scaling deep contrastive learning batch size under memory limited setup. In *Proceedings of the 6th Workshop on Representation Learning for NLP*.

Antonio Mallia, Omar Khattab, Torsten Suel, and Nicola Tonellotto. 2021. Learning passage impacts for inverted indexes. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '21, page 1723–1727. Association for Computing Machinery.

Michael McCandless, Erik Hatcher, Otis Gospodnetić, and O Gospodnetić. 2010. *Lucene in action*, volume 2. Manning Greenwich.

Kosuke Nishida, Kyosuke Nishida, Itsumi Saito, Hisako Asano, and Junji Tomita. 2020. Unsupervised domain adaptation of language models for reading comprehension. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 5392–5399, Marseille, France. European Language Resources Association.

Kyosuke Nishida, Itsumi Saito, Atsushi Otsuka, Hisako Asano, and Junji Tomita. 2018. Retrieve-and-read: Multi-task learning of information retrieval and reading comprehension. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, CIKM '18, page 647–656. Association for Computing Machinery.

Rodrigo Nogueira, Jimmy Lin, and AI Epistemic. 2019. From doc2query to docttttquery. *Online preprint*.

Biswajit Paria, Chih-Kuan Yeh, Ian En-Hsu Yen, Ning Xu, Pradeep Ravikumar, and Barnabás Póczos. 2020. Minimizing flops to learn efficient sparse representations. *CoRR*, abs/2004.05665.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Yingqi Qu, Yuchen Ding, Jing Liu, Kai Liu, Ruiyang Ren, Wayne Xin Zhao, Daxiang Dong, Hua Wu, and Haifeng Wang. 2021. RocketQA: An optimized training approach to dense passage retrieval for open-domain question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5835–5847. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. 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. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. 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 3982–3992. Association for Computational Linguistics.

Stephen Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389.

Amrita Saha, Rahul Aralikatte, Mitesh M. Khapra, and Karthik Sankaranarayanan. 2018. DuoRC: Towards complex language understanding with paraphrased reading comprehension. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1683–1693. Association for Computational Linguistics.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108.

Siamak Shakeri, Cicero Nogueira dos Santos, Henghui Zhu, Patrick Ng, Feng Nan, Zhiguo Wang, Ramesh Nallapati, and Bing Xiang. 2020. End-to-end synthetic data generation for domain adaptation of question answering systems. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5445–5460, Online. Association for Computational Linguistics.

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.

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2017. NewsQA: A machine comprehension dataset. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 191–200. Association for Computational Linguistics.

George Tsatsaronis, Georgios Balikas, Prodromos Malakasiotis, Ioannis Partalas, Matthias Zschunke, Michael R Alvers, Dirk Weissenborn, Anastasia Krithara, Sergios Petridis, Dimitris Polychronopoulos, et al. 2015. An overview of the bioasq large-scale biomedical semantic indexing and question answering competition. *BMC bioinformatics*, 16(1):1–28.

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Well-read students learn better: The impact of student initialization on knowledge distillation. *CoRR*, abs/1908.08962.

Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. 2020. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272.

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*, SIGIR '00, page 200–207. Association for Computing Machinery.

Shuohang Wang, Mo Yu, Xiaoxiao Guo, Zhiguo Wang, Tim Klinger, Wei Zhang, Shiyu Chang, Gerry Tesauro, Bowen Zhou, and Jing Jiang. 2018. Reinforced ranker-reader for open-domain question answering. volume 32.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi 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 M. 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. Association for Computational Linguistics.

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul N. Bennett, Junaid Ahmed, and Arnold Overwijk. 2021. Approximate nearest neighbor negative contrastive learning for dense text retrieval. In *International Conference on Learning Representations*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380. Association for Computational Linguistics.

Tiancheng Zhao, Xiaopeng Lu, and Kyusong Lee. 2021. SPARTA: Efficient open-domain question answering via sparse transformer matching retrieval. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 565–575. Association for Computational Linguistics.