SPARTA: Efficient Open-Domain Question Answering via Sparse Transformer Matching Retrieval

Tiancheng Zhao[1], **Xiaopeng Lu**[2] **and Kyusong Lee**[1]
SOCO Inc.
[1]{tianchez,kyusongl}@soco.ai
Language Technologies Institute, Carnegie Mellon University
[2]xiaopen2@andrew.cmu.edu

Abstract

We introduce SPARTA, a novel neural retrieval method that shows great promise in performance, generalization, and interpretability for open-domain question answering. Unlike many neural ranking methods that use dense vector nearest neighbor search, SPARTA learns a sparse representation that can be efficiently implemented as an Inverted Index. The resulting representation enables scalable neural retrieval that does not require expensive approximate vector search and leads to better performance than its dense counterpart. We validated our approaches on 4 open-domain question answering (OpenQA) tasks and 11 retrieval question answering (ReQA) tasks. SPARTA achieves new state-of-the-art results across a variety of open-domain question answering tasks in both English and Chinese datasets, including open SQuAD, Natural Question, CMRC and etc. Analysis also confirms that the proposed method creates human interpretable representation and allows flexible control over the trade-off between performance and efficiency. [1]

1 Introduction

Open-domain Question Answering (OpenQA) is the task of answering a question based on a knowledge source. One promising approach to solve OpenQA is Machine Reading at Scale (MRS) (Chen et al., 2017). MRS leverages an information retrieval (IR) system to narrow down to a list of relevant passages and then uses a machine reading comprehension reader to extract the final answer span. This approach, however, is bounded by its pipeline nature since the first stage retriever is not trainable and may return no passage that contains the correct answer.

To address this problem, prior work has focused on replacing the first stage retriever with a trainable ranker (Chidambaram et al., 2018; Lee et al., 2018; Wang et al., 2018). End-to-end systems have also been proposed to combine passage retrieval and machine reading by directly retrieving answer span (Seo et al., 2019; Lee et al., 2019). Despite of their differences, the above approaches are all built on top of the *dual-encoder architecture*, where query and answer are encoded into fixed-size dense vectors, and their relevance score is computed via dot products. Approximate nearest neighbor (ANN) search is then used to enable real-time retrieval for large dataset (Shrivastava and Li, 2014).

In this paper, we argue that the dual-encoder structure is far from ideal for open-domain QA retrieval. Recent research shows its limitations and suggests the importance of modeling complex queries to answer interactions for strong QA performance. Seo et al. (2019) shows that their best performing system underperforms the state-of-the-art due to query-agnostic answer encoding and its over-simplified matching function. Humeau et al. (2019) shows the trade-off between performance and speed when moving from expressive cross-attention in BERT (Devlin et al., 2018) to simple inner product interaction for dialog response retrieval. Therefore, our key research goal is to develop new a method that can simultaneously achieve expressive query to answer interaction and fast inference for ranking.

We introduce SPARTA (Sparse Transformer Matching), a novel neural ranking model. Unlike existing work that relies on a sequence-level inner product, SPARTA uses token-level interaction between every query and answer token pair, leading to superior retrieval performance. Concretely, SPARTA learns sparse answer representations that model the potential interaction between every query term with the answer. The learned sparse answer representation can be efficiently saved in an

[1]Work done during Lu's internship at SOCO.

Inverted Index, e.g., Lucene (McCandless et al., 2010), so that one can query a SPARTA index with almost the same speed as a standard search engine and enjoy the more reliable ranking performance without depending on GPU or ANN search.

Experiments are conducted on two settings: OpenQA (Chen et al., 2017) that requires phrase-level answers and retrieval QA (ReQA) that requires sentence-level answers (Ahmad et al., 2019). Our proposed SpartaQA system achieves new state-of-the-art results across 15 different domains and 2 languages with significant performance gain, including OpenSQuAD, Open Natural Questions, OpenCMRC and etc.

Moreover, model analysis shows that SPARTA exhibits several desirable properties. First SPARTA shows strong domain generalization ability and achieves the best performance compared to both classic IR method and other learning methods in low-resources domains. Second, SPARTA is simple and efficient and achieves better performance than many more sophisticated methods. Lastly, it provides a human-readable representation that is easy to interpret. In short, the contributions of this work include:

- A novel ranking model SPARTA that offers token-level query-to-answer interaction and enables efficient large-scale ranking.
- New state-of-the-art experiment results on 11 ReQA tasks and 4 OpenQA tasks in 2 languages.
- Detailed analyses that reveal insights about the proposed methods, including generalization and computation efficiency.

2 Related Work

The classical approach for OpenQA depends on knowledge bases (KB)s that are manually or automatically curated, e.g., Freebase KB (Bollacker et al., 2008), NELL (Fader et al., 2014) etc. Semantic parsing is used to understand the query and computes the final answer (Berant et al., 2013; Berant and Liang, 2014). However, KB-based systems are often limited due to incompleteness in the KB and inflexibility to changes in schema (Ferrucci et al., 2010).

A more recent approach is to use text data directly as a knowledge base. Dr.QA uses a search engine to filter to relevant documents and then applies machine readers to extract the final answer (Chen et al., 2017). It needs two stages because all existing machine readers, for example, BERT-based models (Devlin et al., 2018), are prohibitively slow (BERT only processes a few thousands of words per second with GPU acceleration). Many attempts have been made to improve the first-stage retrieval performance (Chidambaram et al., 2018; Seo et al., 2019; Henderson et al., 2019; Karpukhin et al., 2020; Chang et al., 2020). Yet, the information retrieval (IR) community has shown that simple word embedding matching do not perform well for ad-hoc document search compared to classic methods (Guo et al., 2016; Xiong et al., 2017).

To increase the expressiveness of dual encoders, Xiong et al. (2017) develops kernel function to learn soft matching score at token-level instead of sequence-level. Humeau et al. (2019) proposes Poly-Encoders to enable more complex interactions between the query and the answer by letting one encoder output multiple vectors instead of one vector. Dhingra et al. (2020) incorporates entity vectors and multi-hop reasoning to teach systems to answer more complex questions. (Lee et al., 2020) augments the dense answer representation with learned n-gram sparse feature from contextualized word embeddings, achieving significant improvement compared to the dense-only baseline. Chang et al. (2020) explores various unsupervised pretraining objectives to improve dual-encoders' QA performance in the low-resources setting.

Unlike most of the existing work based-on dual-encoders, we explore a different path where we focus on learning sparse representation and emphasizes token-level interaction models instead of sequence-level. This paper is perhaps the most related to the sparse representations from (Lee et al., 2020). However, the proposed approach is categorically different in the following ways. (1) it is stand-alone and does not need augmentation with dense vectors while keeping superior performance (2) our proposed model is architecturally simpler and is generative so that it will understand words that not appear in the answer document, whereas the one developed at (Lee et al., 2020) only models n-grams appear in the document.

3 Proposed Method

3.1 Problem Formulation

First, we formally define the problem of answer ranking for question answering. Let q be the input

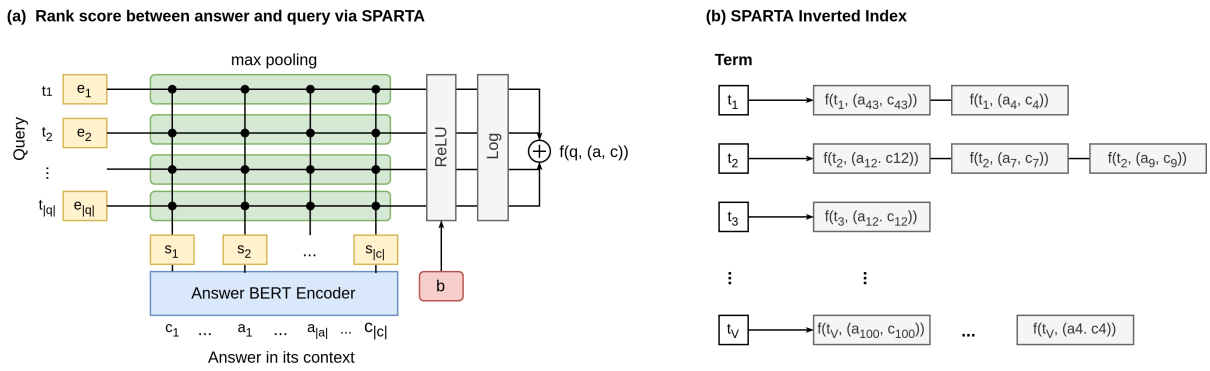

Figure 1: SPARTA Neural Ranker computes token-level matching score via dot product. Each query terms' contribution is first obtained via max-pooling and then pass through ReLU and log. The final score is the summation of each query term contribution.

question, and $A = \{(a, c)\}$ be a set of candidate answers. Each candidate answer is a tuple (a, c) where a is the answer text and c is context information about a. The objective is to find model parameter θ that rank the correct answer as high as possible, .i.e:

$$\theta = \underset{\theta \in \Theta}{\text{argmax}}\, \mathbb{E}[p_\theta((a^*, c^*)|q)] \quad (1)$$

This formulation is general and can cover many tasks. For example, typical passage-level retrieval systems sets the a to be the passage and leaves c empty (Chen et al., 2017; Yang et al., 2019a). The sentence-level retrieval task proposed at sets a to be each sentence in a text knowledge base and c to be the surrounding text (Ahmad et al., 2019). Lastly, the phrase-level QA system sets a to be all valid phrases from a corpus and c to be the surrounding text (Seo et al., 2019). This work focuses on the same sentence-level retrieval task (Ahmad et al., 2019) since it provides a good balance between precision and memory footprint. Yet note that our methods can be easily applied to the other two settings.

3.2 SPARTA Neural Ranker

In order to achieve both high accuracy and efficiency (scale to millions of candidate answers with real-time response), the proposed SPARTA index is built on top of two high-level intuitions.

- Accuracy: retrieve answer with expressive embedding interaction between the query and answer, i.e., token-level contextual interaction.
- Efficiency: create query agnostic answer representation so that they can be pre-computed at indexing time. Since it is an offline operation, we can use the most powerful model for indexing and simplify the computation needed at inference.

As shown in Figure 1, a query is represented as a sequence of tokens $q = [t_1, ...t_{|q|}]$ and each answer is also a sequence of tokens $(a, c) = [c_1, ..a_1, ..a_{|a|}, c_{a+1}, ...c_{|c|}]$. We use a **non-contextualized** embedding to encode the query tokens to e_i, and a **contextualized** transformer model to encode the answer and obtain contextualized token-level embedding s_j:

$$\mathcal{E}(q) = [e_1, ...e_{|q|}] \quad \text{Query Embedding} \quad (2)$$

$$\mathcal{H}(a, c) = [s_1, ...s_{|c|}] \quad \text{Answer Embedding} \quad (3)$$

Then the matching score f between a query and an answer is computed by:

$$y_i = \max_{j \in [1,|c|]}(e_i^T s_j) \quad \text{Term Matching} \quad (4)$$

$$\phi(y_i) = \text{ReLU}(y_i + b) \quad \text{Sparse Feature} \quad (5)$$

$$f(q, (a, c)) = \sum_{i=0}^{|q|} \log(\phi(y_i) + 1) \quad \text{Final Score} \quad (6)$$

where b is a trainable bias. The final score between the query and answer is the summation of all individual scores between each query token and the answer. The logarithm operations normalize each individual score and weaken the overwhelmingly large term score. Additionally, there are two key design choices worth of elaboration.

Token-level Interaction SPARTA scoring uses token-level interaction between the query and the answer. Motivated by bidirectional-attention flow (Seo et al., 2016), relevance between every query and answer token pair is computed via dot product and max pooling in Eq. 4. Whereas in a typical dual-encoder approach, only sequence-level interaction is computed via dot product. Results in our experiment section show that fine-grained interaction is crucial to obtain significant accuracy improvement. Additionally, s_j is obtained from powerful bidirectional transformer encoders, e.g. BERT and only needs to be computed at the indexing time. On the other hand, the query embedding is non-contextual, a trade-off needed to enable real-time inference, which is explained in Section 3.4

Sparsity Control Another key feature to enable efficient inference and memory foot print is sparsity. This is achieved via the combination of log, ReLU and b in Eq. 5. The bias term is used as a threshold for y_i. The ReLU layer forces that only query terms with $y_i > 0$ have impact to the final score, achieving sparse activation. The log operation is proven to be useful via experiments for regularizing individual term scores and leads to better performance and more generalized representation.

Implementation In terms of implementation, we use a pretrained 12-layer, 768 hidden size *bert-base-uncased* as the answer encoder to encode the answer and their context (Devlin et al., 2018). To encode the difference between the answer sequence and its surrounding context, we utilized the segment embedding from BERT, i.e. the answer tokens have segment_id $= 1$ and the context tokens havesegment_id $= 0$. Moreover, the query tokens are embedded via the word embedding from the bert-base-uncased with dimension 768.

3.3 Learning to Rank

The training of SPARTA uses cross entropy learning-to-rank loss and maximizes Eq. 7. The objective tries to distinguish between the true relevant answer $(a^+, c^+)$and irrelevant/random answers K^- for each training query q:

$$J = f(q, (a^+, c^+)) - \log \sum_{k \in K^-} e^{f(q,(a_k,c_k))} \quad (7)$$

The choice of negative samples K^- are crucial for effective learning. Our study uses two types of negative samples: 50% of the negative samples are randomly chosen from the entire answer candidate set, and the rest 50% are chosen from sentences that are nearby to the ground truth answer a. The second case requires the model to learn the fine-grained difference between each sentence candidate instead of only rely on the context information. The parameters to learn include both the query encoder $\mathcal{E}$ and the answer encoder $\mathcal{H}$. Parameters are optimized using back propagation (BP) through the neural network.

3.4 Indexing and Inference

One major novelty of SPARTA is how one can use it for real-time inference. That is for a testing query $q = [t_0, ...t_{|q|}]$, the ranking score between q and an answer is:

$$\text{LOOKUP}(t, (a, c)) = \log(\text{Eq. 5}) \quad t \in V \quad (8)$$

$$f(q, (a, c)) = \sum_{i=1}^{|q|} \text{LOOKUP}(t_i, (a, c)) \quad (9)$$

Since the query term embedding is non-contextual, we can compute the rank feature $\phi(t, (a, c))$ for every possible term t in the vocabulary V with every answer candidate. The result score is cached in the indexing time as shown in Eq. 8. At inference time, the final ranking score can be computed via O(1) look up plus a simple summation as shown in Eq. 9.

More importantly, the above computation can be efficiently implemented via a Inverted Index (Manning et al., 2008), which is the underlying data structure for modern search engines, e.g. Lucene (McCandless et al., 2010) as shown in Figure 1(b). This property makes it easy to apply SPARTA to real-world applications.

3.5 Relation to Classic IR and Generative Models

It is not hard to see the relationship between SPARTA and classic BM25 based methods. In the classic IR method, only the tokens that appeared in the answer are saved to the Inverted Index. Each term's score is a combination of *Term Frequency* and *Inverted Document Frequency* via heuristics (Manning et al., 2008). On the other hand, SPARTA learns which term in the vocabulary should be inserted into the index, and predicts the ranking score directly rather than heuristic calculation. This enables the system to find relevant answers, even when none of the query words appeared in the answer text. For example, if the answer sentence is "Bill Gates founded Microsoft",

a SPARTA index will not only contain the tokens in the answer, but also include relevant terms, e.g. *who*, *founder*, *entrepreneur* and etc.

SPARTA is also related to generative QA. The scoring between (a, c) and every word in the vocabulary V can be understood as the un-normalized probability of $\log p(q|a) = \sum_i^{|q|} \log p(t_i|a)$ with term independence assumption. Past work such as Lewis and Fan (2018); Nogueira et al. (2019) trains a question generator to score the answer via likelihood. However, both approaches focus on auto-regressive models and the quality of question generation and do not provide an end-to-end solution that enables stand-alone answer retrieval.

4 OpenQA Experiments

We consider an Open-domain Question Answering (OpenQA) task to evaluate the performance of SPARTA ranker. Following previous work on OpenQA (Chen et al., 2017; Wang et al., 2019; Xie et al., 2020), we experiment with two English datasets: SQuAD (Rajpurkar et al., 2016), Natural Questions (NQ) (Kwiatkowski et al., 2019); and two Chinese datasets: CMRC (Cui et al., 2018), DRCD (Shao et al., 2018). For each dataset, we used the version of Wikipedia where the data was collected from. Preliminary results show that it is crucial to use the right version of Wikipedia to reproduce the results from baselines. We compare the results with previous best models.

System-wise we follow the 2-stage ranker-reader structure used in (Chen et al., 2017).

Ranker: We split all documents into sentences. Each sentence is treated as a candidate answer a. We keep the surrounding context words of each candidate answer as its context c. We encode at most 512 word piece tokens and truncate the context surrounding the answer sentence with equal window size. For model training, *bert-base-uncased* is used as the answer encoder for English, and *chinese-bert-wwm* is used for Chinese. We reuse the word embedding from corresponding BERT model as the term embedding. Adam (Kingma and Ba, 2014) is used as the optimizer for fine-tuning with a learning rate 3e-5. The model is fine-tuned for at most 10K steps and the best model is picked based on validation performance.

Reader: We deploy a machine reading comprehension (MRC) reader to extract phrase-level answers from the top-K retrieved contexts. For English tasks, we fine-tune on *span-bert* (Joshi et al., 2020). For Chinese tasks, we fine-tune on *chinese-bert-wwm* (Cui et al., 2020). Two additional proven techniques are used to improve performance. First, we use global normalization (Clark and Gardner, 2017) to normalize span scores among multiple passages and make them comparable among each other. Second, distant supervision is used. Concretely, we first use the ranker to find top-10 passages for all training data from Wikipedia corpus. Then every mention of the oracle answers in these contexts are treated as training examples. This can ensure the MRC reader to adapt to the ranker and make the training distribution closer to the test distribution (Xie et al., 2020).

Lastly, evaluation metrics include the standard MRC metric: EM and F1-score.

- Exact Match (EM): if the top-1 answer span matches with the ground truth exactly.
- F1 Score: we compute word overlapping between the returned span and the ground truth answer at token level.

4.1 OpenQA Results

OpenSQuAD		
Model	F1	EM
Dr.QA(Chen et al., 2017)	-	29.8
R^3 (Wang et al., 2018)	37.5	29.1
Par. ranker (Lee et al., 2018)	-	30.2
MINIMAL (Min et al., 2018)	42.5	32.7
DenSPI-hybrid (Seo et al., 2019)	44.4	36.2
BERTserini (Yang et al., 2019a)	46.1	38.6
RE [3] (Hu et al., 2019)	50.2	41.9
Multi-passage (Wang et al., 2019)	60.9	53.0
Graph-retriever (Asai et al., 2019)	63.8	56.5
SPARTA	**66.5**	**59.3**
OpenNQ	EM	
Model	Dev	Test
BERT + BM25 (Lee et al., 2018)	24.8	26.5
Hard EM (Min et al., 2019)	28.8	28.1
ORQA(Lee et al., 2019)	31.3	33.3
Graph-retriever (Asai et al., 2019)	31.7	32.6
SPARTA	**36.8**	**37.5**

Table 1: Results on English Open SQuAD and NQ

Table 1 and 2 shows the SPARTA performance in OpenQA settings, tested in both English and Chinese datasets. Experimental results show that SPARTA retriever outperforms all existing models and obtains new state-of-the-art results on all four

OpenCMRC		
Model	F1	EM
BERTserini(Xie et al., 2020)	60.9	44.5
BERTserini+DS (Xie et al., 2020)	64.6	48.6
SPARTA	**79.9**	**62.9**
OpenDRCD		
Model	F1	EM
BERTserini (Xie et al., 2020)	65.0	50.7
BERTserini+DS (Xie et al., 2020)	67.7	55.4
SPARTA	**74.6**	**63.1**

Table 2: Results on Chinese Open CMRC and DRCD

datasets. For OpenSQuAD and OpenNQ, SPARTA outperforms the previous best system (Asai et al., 2019) by 2.7 absolute F1 points and 5.1 absolute EM points respectively. For OpenCMRC and OpenDRCD, SPARTA achieves a 15.3 and 6.7 absolute F1 points improvement over the previous best system (Xie et al., 2020).

Notably, the previous best system on OpenSQuAD and OpenNQ depends on sophisticated graph reasoning (Asai et al., 2019), whereas the proposed SPARTA system only uses single-hop ranker and require much less computation power. This suggests that for tasks that requires only single-hop reasoning, there is still big improvement room for better ranker-reader QA systems.

5 Retrieval QA Experiments

We also consider Retrieval QA (ReQA), a sentence-level question answering task (Ahmad et al., 2019). The candidate answer set contains every possible sentence from a text corpus and the system is expected to return a ranking of sentences given a query. The original ReQA only contains SQuAD and NQ. In this study, we extend ReQA to 11 different domains adapted from (Fisch et al., 2019) to evaluate both *in-domain performance* and *out-of-domain generalization*. The details of the 11 ReQA domains are in Table 3 and Appendix.

The in-domain scenarios look at domains that have enough training data (see Table 3). The models are trained on the training data and the evaluation is done on the test data. On the other hand, the out-of-domain scenarios evaluate systems' performance on test data from domains not included in the training, making it a zero-shot learning problem. There are two out-of-domain settings: (1) training data only contain SQuAD (2) training data contain only SQuAD and NQ. Evaluation is carried on all the domains to test systems' ability to generalize to unseen data distribution.

Domain	**Data Source**
Has training data	
SQuAD (Rajpurkar et al., 2016)	Wikipedia
News (Trischler et al., 2016)	News
Trivia (Joshi et al., 2017)	Web
NQ (Kwiatkowski et al., 2019)	Google Search
Hotpot (Yang et al., 2018)	Wikipedia
Has no training data	
BioASQ (Tsatsaronis et al., 2015)	PubMed Documents
DROP (Dua et al., 2019)	Wikipedia
DuoRC (Saha et al., 2018)	Wikipedia+IMDB
RACE (Lai et al., 2017)	English Exam
RE (Levy et al., 2017)	Wikipedia
Textbook (Kembhavi et al., 2017)	K12 Textbook

Table 3: 11 corpora included in MultiReQA and their document sources. The top 5 domains contain training data and the bottom 6 domains only have test sets.

For evaluation metrics, we use *Mean Reciprocal Rank* (MRR) as the criteria. The competing baselines include:

BM25: a strong classic IR baseline that is difficult to beat (Robertson et al., 2009).

USE-QA[2]: universal sentence encoder trained for QA task by Google (Yang et al., 2019b). USE-QA uses the dual-encoder architecture and it is trained on more than 900 million mined question-answer pairs with 16 different languages.

Poly-Encoder (Poly-Enc): Poly Encoders improves the expressiveness of dual-encoders with two-level interaction (Humeau et al., 2019). We adapted the original dialog model for QA retrieval: two *bert-base-uncased* models are used as the question and answer encoders. The answer encoder has 4 vector outputs.

5.1 In-domain Performance

Table 4 shows the MRR results on the five datasets with in-domain training. SPARTA can achieve the best performance across all domains with a large main. In terms of average MRR across the five domains, SPARTA is 114.3% better than BM25, 50.6% better than USE-QA and 26.5% better than Poly-Encoders.

Two additional insights can be drawn from the results. First, BM-25 is a strong baseline and does not require training. It performs particularly well in domains that have a high-rate of word-overlapping between the answer and the questions. For example,

[2] https://tfhub.dev/google/universal-sentence-encoder-multilingual-qa

Data	BM25	USE-QA	Poly Enc	SPARTA (ours)
SQuAD	58.0	62.5	64.6	**78.5**
News	19.4	26.2	28.3	**46.6**
Trivia	29.0	41.2	39.5	**55.5**
NQ	19.7	58.2	69.9	**77.1**
HotPot	23.9	25.5	51.8	**63.8**
Avg	30.0	42.7	50.8	**64.3**

Table 4: MRR comparison for the in-domain settings. The proposed SPARTA consistently outperform all the baseline models with large margin. BM25 and USE-QA are unsupervised and pre-trained respectively.

SQuAD's questions are generated by crowd workers who look at the ground truth answer, while question data from NQ/News are generated by question makers who do not see the correct answer. BM25 works particularly well in SQuAD while performing the poorest in other datasets. Similar observations are also found in prior research (Ahmad et al., 2019).

Second, the results in Table 4 confirms our hypothesis on the importance of rich interaction between the answer and the questions. Both USE-QA and Poly Encoder use powerful transformers to encode the whole question and model word-order information in the queries. However, their performance is bounded by the simple dot-product interaction between the query and the answer. On the other hand, despite the fact that SPARTA does not model word-order information in the query, it is able to achieve a big performance gain compared to the baselines, confirming the effectiveness of the proposed token-level interaction method in Eq. 4.

5.2 Out-of-domain Generalization

Table 5 summarized the results for out-of-domain performance comparison. SPARTA trained only on SQuAD outperforms the baselines, achieving 54.1% gain compared to BM25, 26.7% gain compared to USE-QA and 25.3% gain compared to Poly-Encoders in terms of average MRR across 11 different datasets. When SPARTA is trained on SQuAD+NQ, an additional 1.7 MRR improvement is gained compared to SPARTA-SQuAD.

We can observe that Poly-Encoder is able to achieve similar in-domain performance for the domains that are included in the training. However, its performance decreases significantly in new domains, a 25.0% drop compared to its full performance for Poly-Encoder that is trained on SQuAD and 29.2% drop when it's trained on SQuAD+NQ.

Meanwhile, SPARTA generalizes its knowledge from the training data much better to new domains. When trained on SQuAD, its performance on News, Trivia, NQ, and HotPot is only 19.2% lower than the full performance and 18.3% drop when it's trained on SQuAD+NQ. Also, we note that SPARTA's zero-shot performance on News (MRR=41.2) and Trivia (MRR=45.8) is even better than the full performance of Poly-Encoder (News MRR=28.3 and Trivia MRR=39.5).

6 Model Analysis

6.1 Interpreting Sparse Representations

One common limitation of deep neural network models is poor interpretability. Take dense distributed vector representation for example, one cannot directly make sense of each dimension and has to use dimension reduction and visualization methods, e.g. TSNE (Maaten and Hinton, 2008). On the contrary, the resulting SPARTA index is straightforward to interpret due to its sparse nature. Specifically, we can understand a SPARTA vector by reading the top K words with non-zero $f(t,(a,c))$, since these terms have the greatest impact to the final ranking score.

Table 6 shows some example outputs. It is not hard to note that the generated terms for each answer sentence is highly relevant to both a and c, and contains not keywords that appeared in the answer, but also include terms that are potentially in the query but never appear in the answer itself. Two experts manually inspect the outputs for 500 (a,c) data points from Wikipedia, and we summarize the following four major categories of terms that are predicted by SPARTA.

Conversational search understanding: the third row is an example. "Who" appears to the top term, showing it learns Bill Gates is a person so that it's likely to match with "Who" questions.

Keyword identification: terms such as "gates, google, magnate, yellowstone" have high scores in the generated vector, showing that SPARTA learns which words are important in the answer.

Synonyms and Common Sense: "benefactor, investors" are examples of synonyms. Also even though "Utah" does not appear in the answer, it is predicted as an important term, showing that SPARTA leverages the world-knowledge from a pretrained language model and knows Yellowstone is related to Utah.

Model	SQuAD	News	Trivia	NQ	HotPot	Bio	DROP	DuoRC	RACE	RE	Text	Avg
	Unsupervised or pretrained											
BM25	58.0	19.4	29.0	19.7	23.9	8.9	32.6	20.1	14.8	87.4	21.6	30.5
USE-QA	62.5	26.2	41.2	58.2	25.5	7.7	31.9	20.8	25.6	84.8	26.4	37.4
	Trained on SQuAD											
PolyEnc	64.6*	22.2	35.9	57.6	26.5	9.1	32.6	25.4	24.7	88.3	26.0	37.5
SPARTA	78.5*	**41.2**	45.8	62.0	**47.7**	14.5	37.2	35.9	29.7	96.0	28.7	47.0
	Trained on SQuAD + NQ											
PolyEnc	63.9*	19.8	36.9	69.7*	29.6	8.8	30.7	19.6	25.2	72.8	24.6	36.5
SPARTA	**79.0***	40.3	**47.6**	**75.8***	47.5	**15.0**	**37.9**	**36.3**	**30.0**	**97.0**	**29.3**	**48.7**

Table 5: MRR comparison in the out-of-domain settings. The proposed SPARTA is able to achieve the best performance across all tasks, the only learning-based method that is able to consistently outperform BM25 with larger margin in new domains. Results with * are in-domain performance.

Answer (a, c)	**Top terms**
Google was founded in September 1998 by Larry Page and Sergey Brin while they were Ph.D. students at Stanford University in California.	google, when, founded, page, stanford, sergey, larry, founding, established, did, 1998, was, year, formed ...
Yellowstone National Park is an American national park located in the western United States, **with parts in Wyoming, Montana and Idaho.**	montana, yellowstone, wyoming, idaho, park, where, national, western, american, us, utah ...
William Henry Gates is an American business magnate, software developer, investor, and philanthropist. He is best known as the co-founder of Microsoft Corporation.	who, gates, investors, magnate, developer, microsoft, philanthropist, benefactor, investors, ...
Question answering (QA) is a computer science discipline within the fields of information retrieval and natural language processing (NLP).	answering, question, q, computer, information,, retrieval,language, natural, human, nl, science, ...

Table 6: Top-k terms predicted by SPARTA. The text in bold is the answer sentence and the text surrounded it is encoded as its context. Each answer sentence has around 1600 terms with non-zero scores.

Top-K	SQuAD		NQ	
	MRR	R@1	MRR	R@1
50	69.5	61.3	63.2	52.5
100	72.3	64.4	65.6	55.7
500	76.9	69.4	74.4	64.3
1000	78.2	70.8	75.5	65.6
1500	78.6	71.2	75.7	65.7
2000	78.9	71.4	75.9	66.0
Full	79.0	71.6	75.8	66.0

Table 7: Performance on ReQA task with varying sparsity. SPARTA outperforms all baselines with top-50 terms on SQuAD, and with top-500 terms on NQ.

6.2 Sparsity vs. Performance

Sparsity not only provides interpretability, but also offers flexibility to balance the trade-off of memory footprint vs. performance. When there are memory constraints on the vector size, the SPARTA vector can be easily reduced by only keeping the top-K important terms. Table 7 shows performance on SQuAD and NQ with varying K. The resulting sparse vector representation is very robust to smaller K. When only keeping the top 50 terms in each answer vector, SPARTA achieves 69.5 MRR, a better score than all baselines with only 1.6% memory footprint compared to Poly-Encoders (768 x 4 dimension). NQ dataset is more challenging and requires more terms. SPARTA achieves a close to the best performance with top-500 terms.

7 Conclusion

In short, we propose SPARTA, a novel ranking method, that learns sparse representation for better open-domain QA. Experiments show that the proposed framework achieves the state-of-the-art performance for 4 different open-domain QA tasks in 2 languages and 11 retrieval QA tasks. This confirm our hypothesis that token-level interaction is superior to sequence-level interaction for better evidence ranking. Analyses also show the advantages of sparse representation, including interpretability, generalization and efficiency.

Our findings also suggest promising future research directions. The proposed method does not support multi-hop reasoning, an important attribute that enables QA systems to answer more complex questions that require collecting multiple evidence passages. Also, current method only uses a bag-of-word features for the query. We expect further performance gain by incorporating more word-order information.

References

Amin Ahmad, Noah Constant, Yinfei Yang, and Daniel Cer. 2019. Reqa: An evaluation for end-to-end answer retrieval models. *arXiv preprint arXiv:1907.04780*.

Akari Asai, Kazuma Hashimoto, Hannaneh Hajishirzi, Richard Socher, and Caiming Xiong. 2019. Learning to retrieve reasoning paths over wikipedia graph for question answering. *arXiv preprint arXiv:1911.10470*.

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.

Jonathan Berant and Percy Liang. 2014. Semantic parsing via paraphrasing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1415–1425.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250.

Wei-Cheng Chang, Felix X Yu, Yin-Wen Chang, Yiming Yang, and Sanjiv Kumar. 2020. Pre-training tasks for embedding-based large-scale retrieval. *arXiv preprint arXiv:2002.03932*.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading Wikipedia to answer open-domain questions. In *Association for Computational Linguistics (ACL)*.

Muthuraman Chidambaram, Yinfei Yang, Daniel Cer, Steve Yuan, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2018. Learning cross-lingual sentence representations via a multi-task dual-encoder model. *arXiv preprint arXiv:1810.12836*.

Christopher Clark and Matt Gardner. 2017. Simple and effective multi-paragraph reading comprehension. *arXiv preprint arXiv:1710.10723*.

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, Shijin Wang, and Guoping Hu. 2020. Revisiting pre-trained models for chinese natural language processing. In *Findings of EMNLP*. Association for Computational Linguistics.

Yiming Cui, Ting Liu, Wanxiang Che, Li Xiao, Zhipeng Chen, Wentao Ma, Shijin Wang, and Guoping Hu. 2018. A span-extraction dataset for chinese machine reading comprehension. *arXiv preprint arXiv:1810.07366*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Bhuwan Dhingra, Manzil Zaheer, Vidhisha Balachandran, Graham Neubig, Ruslan Salakhutdinov, and William W Cohen. 2020. Differentiable reasoning over a virtual knowledge base. *arXiv preprint arXiv:2002.10640*.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs. *arXiv preprint arXiv:1903.00161*.

Anthony Fader, Luke Zettlemoyer, and Oren Etzioni. 2014. Open question answering over curated and extracted knowledge bases. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1156–1165.

David Ferrucci, Eric Brown, Jennifer Chu-Carroll, James Fan, David Gondek, Aditya A Kalyanpur, Adam Lally, J William Murdock, Eric Nyberg, John Prager, et al. 2010. Building watson: An overview of the deepqa project. *AI magazine*, 31(3):59–79.

Adam Fisch, Alon Talmor, Robin Jia, Minjoon Seo, Eunsol Choi, and Danqi Chen. 2019. Mrqa 2019 shared task: Evaluating generalization in reading comprehension. *arXiv preprint arXiv:1910.09753*.

Jiafeng Guo, Yixing Fan, Qingyao Ai, and W Bruce Croft. 2016. A deep relevance matching model for ad-hoc retrieval. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, pages 55–64.

Matthew Henderson, Iñigo Casanueva, Nikola Mrkšić, Pei-Hao Su, Ivan Vulić, et al. 2019. Convert: Efficient and accurate conversational representations from transformers. *arXiv preprint arXiv:1911.03688*.

Minghao Hu, Yuxing Peng, Zhen Huang, and Dongsheng Li. 2019. Retrieve, read, rerank: Towards end-to-end multi-document reading comprehension. *arXiv preprint arXiv:1906.04618*.

Samuel Humeau, Kurt Shuster, Marie-Anne Lachaux, and Jason Weston. 2019. Poly-encoders: Transformer architectures and pre-training strategies for fast and accurate multi-sentence scoring. *CoRR abs/1905.01969. External Links: Link Cited by*, 2:2–2.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S Weld, Luke Zettlemoyer, and Omer Levy. 2020. Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *arXiv preprint arXiv:1705.03551*.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906*.

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*, pages 4999–5007.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. 2017. Race: Large-scale reading comprehension dataset from examinations. *arXiv preprint arXiv:1704.04683*.

Jinhyuk Lee, Minjoon Seo, Hannaneh Hajishirzi, and Jaewoo Kang. 2020. Contextualized sparse representations for real-time open-domain question answering. *arXiv: Computation and Language*.

Jinhyuk Lee, Seongjun Yun, Hyunjae Kim, Miyoung Ko, and Jaewoo Kang. 2018. Ranking paragraphs for improving answer recall in open-domain question answering. *arXiv preprint arXiv:1810.00494*.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. *arXiv preprint arXiv:1906.00300*.

Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. 2017. Zero-shot relation extraction via reading comprehension. *arXiv preprint arXiv:1706.04115*.

Mike Lewis and Angela Fan. 2018. Generative question answering: Learning to answer the whole question.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605.

Christopher D Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to information retrieval*. Cambridge university press.

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

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

Sewon Min, Victor Zhong, Richard Socher, and Caiming Xiong. 2018. Efficient and robust question answering from minimal context over documents. *arXiv preprint arXiv:1805.08092*.

Rodrigo Nogueira, Jimmy Lin, and AI Epistemic. 2019. From doc2query to doctttttquery.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.

Stephen Robertson, Hugo Zaragoza, et al. 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. *arXiv preprint arXiv:1804.07927*.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2016. Bidirectional attention flow for machine comprehension. *arXiv preprint arXiv:1611.01603*.

Minjoon Seo, Jinhyuk Lee, Tom Kwiatkowski, Ankur Parikh, Ali Farhadi, and Hannaneh Hajishirzi. 2019. Real-time open-domain question answering with dense-sparse phrase index. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4430–4441.

Chih Chieh Shao, Trois Liu, Yuting Lai, Yiying Tseng, and Sam Tsai. 2018. Drcd: a chinese machine reading comprehension dataset. *arXiv preprint arXiv:1806.00920*.

Anshumali Shrivastava and Ping Li. 2014. Asymmetric lsh (alsh) for sublinear time maximum inner product search (mips). In *Advances in Neural Information Processing Systems*, pages 2321–2329.

Adam Trischler, Tong Wang, Xingdi (Eric) Yuan, Justin D. Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2016. Newsqa: A machine comprehension dataset.

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):138.

Shuohang Wang, Mo Yu, Xiaoxiao Guo, Zhiguo Wang, Tim Klinger, Wei Zhang, Shiyu Chang, Gerry Tesauro, Bowen Zhou, and Jing Jiang. 2018. R 3: Reinforced ranker-reader for open-domain question answering. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

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. *arXiv preprint arXiv:1908.08167*.

Yuqing Xie, Wei Yang, Luchen Tan, Kun Xiong, Nicholas Jing Yuan, Baoxing Huai, Ming Li, and Jimmy Lin. 2020. Distant supervision for multi-stage fine-tuning in retrieval-based question answering. In *Proceedings of The Web Conference 2020*, pages 2934–2940.

Chenyan Xiong, Zhuyun Dai, Jamie Callan, Zhiyuan Liu, and Russell Power. 2017. End-to-end neural ad-hoc ranking with kernel pooling. In *Proceedings of the 40th International ACM SIGIR conference on research and development in information retrieval*, pages 55–64.

Wei Yang, Yuqing Xie, Aileen Lin, Xingyu Li, Luchen Tan, Kun Xiong, Ming Li, and Jimmy Lin. 2019a. End-to-end open-domain question answering with bertserini. *arXiv preprint arXiv:1902.01718*.

Yinfei Yang, Daniel Cer, Amin Ahmad, Mandy Guo, Jax Law, Noah Constant, Gustavo Hernandez Abrego, Steve Yuan, Chris Tar, Yun-Hsuan Sung, et al. 2019b. Multilingual universal sentence encoder for semantic retrieval. *arXiv preprint arXiv:1907.04307*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. *arXiv preprint arXiv:1809.09600*.

A Appendices

Size details of multi-domain ReQA task.

Domain	**# of Query**	**# of Candidate Answer**
SQuAD	11,426	10,250
News	8,633	38,199
Trivia	8,149	17,845
NQ	1,772	7,020
Hotpot	5,901	38,906
BioASQ	1,562	13,802
DROP	1,513	2,488
DuoRC	1,568	5,241
RACE	674	10,630
RE	2,947	2,201
Textbook	1,503	14,831

Table 8: Size of the evaluation test set for the 11 corpora included in MultiReQA.