NLP Project Report

Huang Ziqing,[1] **Gao Tian,** [1] **Wang Wenhao,** [1] **Zhang Bingliang,** [1]
[1] IIIS, Tsinghua University, Beijing, China
{hzq19, gaot19, wangwh19, zhangbl19}@mails.tsinghua.edu.cn

Abstract

Open-domain Question Answering is an active and important field of research in NLP, and in recent years its popularity has surged dramatically. The modern research of Open-domain QA suggests a 2-stage framework consisting of Passage Retrieval and Machine Reading Comprehension. In an attempt to improve the retrieval accuracy by augmenting existing passage retrievers, we propose two different approaches, where the first one focuses on improving the robustness of queries through data augmentation and the second one places emphasis on making the queries more information-dense by adding related texts to it. For the first scheme, we use data augmentation based on back-translation and synonyms together with an auxiliary NLL loss using hard negative samples. In the second method, we use generation-augmented queries for passage retrieval. We discover that the two different approaches can both go beyond state-of-the-art results.

Introduction

Open-domain Question Answering is a task to answer questions given in natural language based on a large collection of documents. The modern research of Open-domain QA suggests a 2-stage framework consisting of Passage Retrieval and Machine Reading Comprehension.

Recent dense passage retrievers outperform traditional sparse retrieval methods like BM25 on popular question answering datasets like Natural Question (NQ). These dense models are trained using supervised datasets and the dense passage retriever (DPR) model (Karpukhin et al. 2020).

In this work, we first identify that the bottleneck of the task is the performance of the retriever, in which the retrieval accuracy has a large room for improvement. Then we proposed two approaches that can augment the retrieval stage. The first one focuses on improving the robustness of queries through data augmentation and the second one places emphasis on making the queries more information-dense by adding related texts to them. Finally, we tested different methods on SmallNQ dataset (cut from NQ, described in sections below), discovering that these methods both outperform the original DPR retriever.

Background And Related Work

Open-domain QA is a long-standing task that aims to give an accurate answer to a given question based on large-scale unstructured documents. Different from a search engine, the Open-domain QA system requires to give the final answers instead of the raw retrieved documents (Zhu et al. 2021).

Traditional architecture involves *question analysis*, *document retrieval* and *answer extraction* steps, where each part required tons of manual efforts and feature engineering. With the recent emergence of deep learning, the *"Retriever-Reader"* architecture has become a new modern paradigm for Open-domain QA. Retriever is aimed to retrieve a small subset of relevant documents given question while the reader is aimed to further extract and aggregate answers from them.

Specifically, the retriever can be regarded as a function $R : (q, C) \rightarrow C_q$ (Karpukhin et al. 2020), where $|C|$ can be millions to billions while $|C_q|$ is restricted to be tens or hundreds. It requires a retriever to have good efficiency. The reader $E : (q, C_q) \rightarrow a$ takes questions and retrieved documents as input, and finds the answer to the question. In principle, a much finer comparison and comprehension in relation and interaction between questions and documents are required in the reading phase.

Passage Retrieval

Sparse Retrieval In classical IR (Information Retrieval) system, sparse vectors, such as TF-IDF and BM25, are used to represent documents, passages, or questions. These mature methods were combined with modern neural MRC (Machine Reading Comprehension) to build an end-to-end Open-domain QA system in the past few years. DrQA (Chen et al. 2017) adopted bi-gram hashing and TF-IDF matching to search over a large corpus of Wikipedia documents. BERTerini (Yang et al. 2019) took use of retrieval software based on Lucene. However, one of the intrinsic drawbacks of sparse retrieval is its term-based feature, which makes it difficult to capture much implicit semantic information.

Dense Retrieval As the counterpart of a sparse one, dense retrieval is aimed to represent the semantic meaning of words or documents. Also, a learnable encoding function enables more flexibility for dense retrieval. This novel idea can

be traced back to Word2Vec model (Mikolov et al. 2013). ORQA (Lee, Chang, and Toutanova 2019) was the first work to show that *retriever* and *reader* could be jointly learned from question-answer string pairs without any existing IR system. They designed a pretraining procedure, called ICT (Inverse Cloze Task), for *retriever*. The paper showed that the trained retriever outperformed BM25 over a large margin. What's more, even the pretraining for *retriever* is unnecessary shown by (Karpukhin et al. 2020). A metrics learning framework was proposed to create a vector space so that relevant pairs of questions and documents wound have a smaller distance, e.g. standard inner product.

Iterative Retrieval As the name suggests, the documents are retrieved in multiple steps. The core operation is the *question reformulation* after each iteration based on currently retrieved documents. Instead of using original questions to search all documents at once, it gives the retriever opportunity to rephrase the question based on essential context. The method for reformulation is flexible. In (Das et al. 2019), the Gate Recurrent Unit was adopted as a reformulation model and trained by Reinforcement Learning. MDR (Xiong et al. 2020) encoded one-step retrieved passages as well as previous ones as a new query for further retrieval. Since this scheme allows more question and documents interaction during retrieval, the performance may have great potential in future research.

Generation-Augmented Retrieval In this method, it is proposed that generation-augmented questions (Mao et al. 2020) increase the performance of various retrieval schemes, including BM25, where comparable top-k retriever accuracy is achieved. Specifically, this method proposes that if we use *Seq2Seq* model to generate some contexts such as the answer of the query, the title of the target passage, etc., and concatenate the generated results to the query, the augmented query is more retriever-friendly. One potential problem with this method is that it is not better than DPR in top-k accuracy when k is small, e.g., $k = 5$.

Contrastive learning of sentence embeddings Contrastive learning is popular and impressively successful on representation learning. The critical part of contrastive learning for sentence embedding is the construction of positive instances. One previous work (Fang et al. 2020) uses back-translation as positive instances and performs better than traditional token-level augmentations like word deletion and replacement since back-translation can capture the global semantic meaning of sentences. Another work (Gao, Yao, and Chen 2021) proposes a continuous augmentation method that uses random dropout of Transformers to construct positive pairs and also outperforms traditional discrete augmentations.

Alignment and Uniformity (Wang and Isola 2020) proposes two critical metrics of representation learning. One is *alignment* which evaluates the closeness of positive embedding pairs and the other is *uniformity* which evaluates the distribution of embeddings. Directly optimizing these two metrics can learn a comparable or even better representation on downstream tasks than contrastive loss. Inspired by this, we propose paraphrasing augmentations to get more robust alignment and hard negative instances to get higher uniformity for query embeddings.

Machine Reading Comprehension

Unlike canonical MRC where passage-answer pairs are usually available, *reader* in Open-domain QA is required to not only extract but also aggregate answers from a set of retrieved candidate documents. Answers may appear once or multiple times, making it difficult to extract and aggregate. Depending on answering methods, there are mainly two types of *reader*.

Extractive Reader This method relies on the assumption that the correct answer to the given question appears as a span in one or some of the documents. Thus *reader* only needs to predict which span should be the correct answer. In DS-QA (Lin et al. 2018), Paragraph Selector first applied to filter documents, and Paragraph Reader was then used to extract the answer. Instead of regarding candidate documents as isolation, Graph Retrieval (Min et al. 2019) tried to propagate information among related documents based on a BERT reader through a documents graph, where edges represent the relationship between documents. Also, the prominent MRC model such as SpanBERT (Joshi et al. 2020), which is better at predicting a span from a given document, could be also plugged in for better performance.

Generative Reader Compared to predicting a span in documents, the generative reader tends to directly generate the answer to the given question in natural language form based on retrieved documents. This method is more flexible, less restrictive. Recent model Fusion-in-Decoder (Izacard and Grave 2020) directly adopted a pretrained Seq2Seq model, BART or T5, as generative reader. The output answer is conditioned to the concatenation of token-wised encoding in the entire candidate documents. It outperformed several previous works over a large margin in datasets like TriviaQA and NQ (Natural Question).

Method

Overview

On one hand, we add an auxiliary loss on original DPR cross-entropy loss, where positive instances are generated by paraphrasing models and (hard) negative instances are selected simply in a batch or by the pretrained DPR query encoder. On the other hand, we concatenate contexts generated by pretrained T5 to original queries in order to introduce more information. Then we change the DPR pipeline in that we randomly choose one query from the augmented corpus for the query encoder input and that we introduce to the training loss an auxiliary loss term where the similarity between different encoded vectors of augmented queries are taken into account.

The complete pipeline is shown in Figure 1.

Query Augmentation with Auxiliary Loss

Query Augmentation Different questions with the same semantic meaning are expected to have similar embeddings.

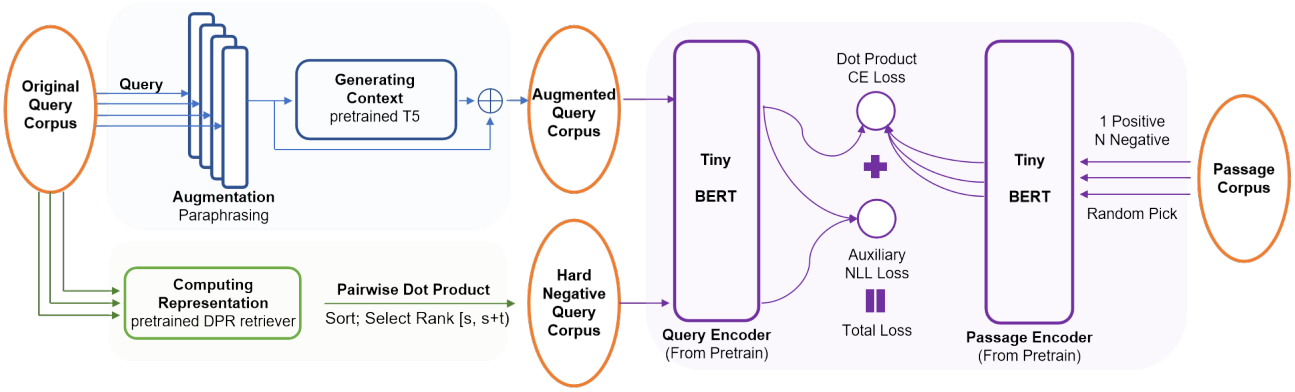

Figure 1: Our complete pipeline with query augmentation and context generative model. Original CE loss is combined with an auxiliary NLL loss for entire training process. The negative queries for auxiliary loss are other queries within same batch together with hard negative queries computed by a pretrained DPR retriever.

With this assumption, we attempt to paraphrase questions by back translations and train the DPR encoder to learn similar embeddings among paraphrased questions. We use a machine translation model to translate questions from English to German, French, and Spanish and translate them back to English. Thus, we derive three paraphrased questions for each original query.

Also, we tried a synonym replacement method, provided by NLPAUG (Ma 2019), to generate query augmentation, inspired by the intuition that simply replacing words in the query by their corresponding synonym will result in the query with similar meaning.

Auxiliary Loss Let $D = \{\langle q_i, p_i^+, p_{i,1}^-, ..., p_{i,n}^- \rangle\}_{i=1}^m$ denote training data, where q_i is the query, p_i^+ is the corresponding positive passage and $p_{i,j}^-$ are the negative passages. Also, we have

$$sim(q, q') = f(q) \cdot f'(q') \tag{1}$$

where $f(\cdot), f'(\cdot)$ are the BERT encoder of the texts. Then we have the original DPR loss described by the following equation:

$$L(D) = -\log \frac{e^{sim(q_{i,1}^*, p_i^+)}}{e^{sim(q_{i,1}^*, p_i^+)} + \sum_{j=1}^n e^{sim(q_{i,1}^*, p_{i,j}^-)}} \tag{2}$$

Let E_q denote the query encoder. We modify the original DPR loss by introducing paraphrased questions $q'_{i,j}$ and add an auxiliary loss to encourage the similarity of paraphrased questions' embeddings. Let Q_i be the set where the auxiliary loss of q_i is computed from, then the new augmented DPR loss is

$$L(D) = -\log \frac{e^{sim(q_{i,1}^*, p_i^+)}}{e^{sim(q_{i,1}^*, p_i^+)} + \sum_{j=1}^n e^{sim(q_{i,1}^*, p_{i,j}^-)}} + \lambda \cdot L'(q_i, Q_i)$$

where $L'(\cdot)$ is the auxiliary loss, and λ is the coefficient of the auxiliary loss which is properly chosen to make the two loss terms comparable in scale.

We consider two kinds of auxiliary loss that pull queries with similar meaning together, i.e., MSE loss and NLL loss.

The MSE auxiliary loss is defined as follows:

$$L'_{\text{MSE}}(q_i, q'_i) = ||q_i - q'_i||_2^2 \tag{3}$$

where q'_i is uniformly drawn from the augmented query corpus.

The NLL auxiliary loss is defined as

$$L'_{\text{NLL}}(q_i, r_{i,1}^-, r_{i,2}^-, \cdots, r_{i,t}^-) = -\log \frac{e^{sim(q_i, q'_i)}}{e^{sim(q_i, q'_i)} + \sum_{j=1}^t e^{sim(q_i, r_{i,j}^-)}}$$

where the $r_{i,j}$ are the negative samples and $sim(\cdot)$ is given in Equation 1. We mention here that the selection of these negative samples are crucial to the performance of the NLL auxiliary loss, and we consider several different negative selection schemes.

1. **In-Batch Negatives**: The first kind of negative samples are the other queries in the same batch in the training stage.
2. **Static Hard Negatives**: The second kind of negative samples are generated from the trained DPR model's query encoder. For a query q_i, we rank all the queries q' according to $sim(q_i, q')$ and pick the results that rank $2, 3, \cdots, t+1$ as the hard negatives, where $sim(\cdot)$ is given in Equation 1
3. **Dynamic Hard Negatives**: The third kind of negative samples are generated as the training of the model proceeds. After some given epoch, for each query q_i we pick the top queries that are similar to q_i as the hard negative samples.

The details of experiments and results are described in the next section .

Generation-Augmented DPR

Mao et al. (2020) propose a method that combines generative augmentation with sparse retriever BM25 and achieves

Methods	Top 5	Top 10	Top 20	Top 50	Top 100
DPR btz = 16	0.118	0.163	0.211	0.286	0.332
DPR btz = 32	0.122	0.162	0.212	0.285	0.335
NLL+BN	0.124	0.176	0.217	0.288	0.339
NLL+S-HN	0.126	0.168	0.217	0.290	**0.345**
NLL+D10-HN	0.113	0.166	0.214	0.281	0.333
MSE+BN	0.110	0.152	0.198	0.263	0.321
GAR	0.159	0.210	0.260	0.327	**0.379**
GAR+NLL+D1-HN	0.163	0.212	0.263	0.328	0.374
GAR+NLL+D10-HN	0.161	0.201	0.261	0.327	0.371
GAR+NLL+S-HN	0.160	0.208	0.258	0.329	**0.378**

Table 1: The top-k retrieval accuracy results. *DPR btz = k* stands for our baseline DPR model of batch size k. BN stands for in-batch negatives and HN means enhanced BN by hard negatives. "S" and "Dk" represents fixed hard negative and dynamic hard negative updated every k epochs respectively. In GAR, queries are augmented by generated answers.

a comparable top-k accuracy with DPR. The intuition is that query augmentation (answer/title/context) can bring more information for a better location during retrieval. Motivated by the similar intuition, we believe that combining query augmentation with dense retriever DPR can also enhance the performance. Each query is augmented with an answer, title, context, or all of them and input to DPR retriever. We find that all of these augmentations can improve the performance of DPR retriever, especially for small top-k accuracy. Among these augmentations, DPR retriever with answer augmentation brings impressive enhancement compared with the original DPR retriever.

See the next section for the experimental results of Generation-Augmented method.

Experiments

As a compromise of limitation of resources, we use a smaller dataset, corpus, and BERT model. Specifically, 1% wiki documents are used as Corpus (around 210K passages), referred to as "SmallWiki". Similarly, a smaller size of NQ dataset is used (10% training set, 20% validation set), refers as "SmallNQ". For BERT model, Tiny-BERT (`huawei-noah/TinyBERT_General_4L_312D`) is used to save both storage and computation resources (Jiao et al. 2019).

In the training stage, since the original DPR model does not provide pretraining on Tiny-BERT for the NQ dataset, all the models are trained from pretrained Tiny-BERT (unless specifically mentioned) and using a fixed random seed. Also, all the models are on SmallNQ for 60 epochs on Tiny-BERT with a batch size of 16 or 32, using Adam Optimizer that starts with learning rate 1e-5. The experimental results are shown in Table 1.

Query Augmentation with Auxiliary Loss

We tested two different genres of auxiliary loss, MSE loss, and NLL loss. Changing the auxiliary loss coefficient λ and the choice of negative samples, the models are trained using the previously mentioned configuration with a batch size of 16 when using dynamic hard negatives and with a batch size of 32 for the others.

After training, we test the top-k retrieval accuracy on the original test set and SmallWiki as a measure of performance. We find that the NLL auxiliary loss that uses static hard negative works best together with a coefficient of $\lambda = 1.0$. The retrieval accuracy of the MSE and NLL auxiliary loss methods are presented in Table **??** and in the ablation study subsection that follows.

Generation-Augmented DPR

For the Generation-augmented method, we use pretrained GAR model (Mao et al. 2020) to generate three different kinds of information, i.e., generated "answer", "title" and "context", before training. Also, we tested to concatenate them together as a fourth augmentation, which is denoted as "all". The four different schemes are trained using the previously mentioned configuration with batch size 32.

Still, the top-k results are tested as a performance measure. We find that the augmentation of "answer" works best, while all the four generation-augmentation schemes improve the performance of the DPR baseline. The retrieval accuracy of Generation-Augmented DPR is presented in Table 1 and the ablation study subsection that follows.

Combination of Methods

We also tested the performance when the query augmentation method is combined with the generation-augmentation method. We trained the models of the four generation schemes (answer/title/context/all) and used the NLL auxiliary loss with static/dynamic hard negatives using the previously mentioned training configuration.

The retrieval accuracy of the different methods is shown in Table 1. We notice that when the two methods are combined, the performance does not further increase than the one using only "answer" Generation-Augmentation, but is still better than the ones using only auxiliary loss.

Ablation Study

Tiny-BERT Alternative: SimCSE We tried to start from another pretrained Tiny BERT using the method proposed in SimCSE (Gao, Yao, and Chen 2021), where the training stage of the alternative BERT uses augmentation methods similar to ours. We trained SimCSE with auxiliary NLL

Method	Top 5	Top 10	Top 20	Top 50	Top 100
SimCSE+NLL+S-HN	0.126	0.168	0.217	0.290	0.343
SimCSE	0.122	0.160	0.212	0.285	**0.337**
NLL+S-HN	0.126	0.168	0.217	0.290	**0.345**
DPR (Baseline)	0.122	0.162	0.212	0.285	0.335

Table 2: The ablation study over different pretrain schemes. Retrieval accuracy using SimCSE-pretrained TinyBERT, compared with original version of pretrained TinyBERT, is tested. Also, we did the experiments with and without the NLL loss.

loss using static hard negative on our pipeline. The model is trained for 60 epochs, with a batch of 32, and loss coefficient of $\lambda = 1.0$. The results are shown in Table 2. We find that despite the fact that it has similar intuition, pretrained SimCSE does not steadily improve the performance of our pipeline, so it is not utilized as a part of our method.

Choice of Auxiliary Loss Coefficient Experiments are conducted to seek for best balance coefficients between auxiliary loss and original DPR CE loss. We remark that the coefficient is robust within a large range of values, from $1e$-2 to 10. They are all trained for 60 epochs with a batch of 32 together with the static hard negatives. The evaluation results of $top - k$ retrieval accuracy are shown in Table 3.

Coeff.	Top 5	Top 10	Top 20	Top 50	Top 100
0.01	0.125	0.168	0.217	0.289	**0.345**
0.1	0.126	0.168	0.217	0.290	**0.345**
1	0.126	0.168	0.217	0.291	0.343
10	0.124	0.169	0.218	0.287	0.342

Table 3: The ablation study over balance coefficient. Coeff. stands for coefficient. The selection are robust in a wide range of coefficient

Choice of Generation-Augmentation Target There are several possible options for generation augmentation, such as "answer", "title", "sentence" (including answer). The same targets are used as original GAR (Mao et al. 2020) since they are all available in the training dataset. We conduct experiments with different augmentation targets on DPR with a batch of 32 and epochs of 60. In these experiments, we do not add auxiliary loss. The performances are dramatically different as shown in Table 4. As we can notice, the query augmented by generated answers outperforms others over a large margin. Intuitively, the generated answer is directly related to downstream retrieval tasks, hence leading to better performance. Another observation is that there may be some training-test overlapping and augmented by answers makes these questions trivial.

Conclusion

In this work, we propose two methods to learn more retriever-friendly query representations from two aspects and both outperform DPR retriever. On one hand, to enhance the robustness and distinguishability of query embeddings, we use augmentation transformers to generate positive instances and select hard negatives by pretrained DPR retriever for contrastive representation learning. On the other hand, to introduce more information about queries to the retriever, we propose a generation-augmented DPR which uses pretrained generation model to generate answer/title/context of queries as attached augmentations. However, when combining these two methods together, the performance is better than the original DPR retriever but only comparable with generation-augmented DPR. One tentative solution might be excluding queries with similar generated augmentation when selecting hard negatives during preprocessing. Also, another possible direction of improvement is using other forms of data augmentation such as entity in our pipeline. These are left as future work.

Targets	Top 5	Top 10	Top 20	Top 50	Top 100
answer	0.159	0.21	0.26	0.327	**0.379**
title	0.13	0.173	0.215	0.287	0.341
sentence	0.127	0.173	0.221	0.29	0.338
all	0.148	0.193	0.245	0.314	0.363

Table 4: The ablation study on augmentation targets. The retriever augmented by answer outperforms others over a large margin.

References

Chen, D.; Fisch, A.; Weston, J.; and Bordes, A. 2017. Reading wikipedia to answer open-domain questions. *arXiv preprint arXiv:1704.00051* .

Das, R.; Dhuliawala, S.; Zaheer, M.; and McCallum, A. 2019. Multi-step retriever-reader interaction for scalable open-domain question answering. *arXiv preprint arXiv:1905.05733* .

Fang, H.; Wang, S.; Zhou, M.; Ding, J.; and Xie, P. 2020. CERT: Contrastive Self-supervised Learning for Language Understanding.

Gao, T.; Yao, X.; and Chen, D. 2021. SimCSE: Simple Contrastive Learning of Sentence Embeddings.

Izacard, G.; and Grave, E. 2020. Leveraging passage retrieval with generative models for open domain question answering. *arXiv preprint arXiv:2007.01282* .

Jiao, X.; Yin, Y.; Shang, L.; Jiang, X.; Chen, X.; Li, L.; Wang, F.; and Liu, Q. 2019. Tinybert: Distilling bert for natural language understanding. *arXiv preprint arXiv:1909.10351* .

Joshi, M.; Chen, D.; Liu, Y.; Weld, D. S.; Zettlemoyer, L.; and Levy, O. 2020. Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics* 8: 64–77.

Karpukhin, V.; Oğuz, B.; Min, S.; Lewis, P.; Wu, L.; Edunov, S.; Chen, D.; and Yih, W.-t. 2020. Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906* .

Lee, K.; Chang, M.-W.; and Toutanova, K. 2019. Latent retrieval for weakly supervised open domain question answering. *arXiv preprint arXiv:1906.00300* .

Lin, Y.; Ji, H.; Liu, Z.; and Sun, M. 2018. Denoising distantly supervised open-domain question answering. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 1736–1745.

Ma, E. 2019. NLP Augmentation. https://github.com/makcedward/nlpaug.

Mao, Y.; He, P.; Liu, X.; Shen, Y.; Gao, J.; Han, J.; and Chen, W. 2020. Generation-augmented retrieval for open-domain question answering. *arXiv preprint arXiv:2009.08553* .

Mikolov, T.; Chen, K.; Corrado, G.; and Dean, J. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* .

Min, S.; Chen, D.; Zettlemoyer, L.; and Hajishirzi, H. 2019. Knowledge guided text retrieval and reading for open domain question answering. *arXiv preprint arXiv:1911.03868* .

Wang, T.; and Isola, P. 2020. Understanding Contrastive Representation Learning through Alignment and Uniformity on the Hypersphere.

Xiong, W.; Li, X. L.; Iyer, S.; Du, J.; Lewis, P.; Wang, W. Y.; Mehdad, Y.; Yih, W.-t.; Riedel, S.; Kiela, D.; et al. 2020. Answering complex open-domain questions with multi-hop dense retrieval. *arXiv preprint arXiv:2009.12756* .

Yang, W.; Xie, Y.; Lin, A.; Li, X.; Tan, L.; Xiong, K.; Li, M.; and Lin, J. 2019. End-to-end open-domain question answering with bertserini. *arXiv preprint arXiv:1902.01718* .

Zhu, F.; Lei, W.; Wang, C.; Zheng, J.; Poria, S.; and Chua, T.-S. 2021. Retrieving and reading: A comprehensive survey on open-domain question answering. *arXiv preprint arXiv:2101.00774* .