Constructing a Closed-Domain Question Answering System With Generative Language Models

Hung Le
Japan Advanced Institute of Science and Technology
Nomi, Japan
hungle@jaist.ac.jp

Le-Minh Nguyen
Japan Advanced Institute of Science and Technology
Nomi, Japan
nguyenml@jaist.ac.jp

Jiaying Ni
Japan Advanced Institute of Science and Technology
Nomi, Japan
jiaying@jaist.ac.jp

Shogo Okada
Japan Advanced Institute of Science and Technology
Nomi, Japan
okada-s@jaist.ac.jp

2023 15th International Conference on Knowledge and Systems Engineering (KSE) | 979-8-3503-2974-2/23/$31.00 ©2023 IEEE | DOI: 10.1109/KSE59128.2023.10299437

***Abstract*—Generative language models such as ChatGPT are good at producing answers for questions whose answers are publicly available. For questions about private organization documents, such models cannot perform well due to the fact that they were not trained on these documents, hence the knowledge was not embedded in their parameters. On the other hand, traditional Question Answering (QA) systems with a Retriever and a Reader model are interpretable and can be trained quickly on private documents, but they require laborious annotation. The output of QA systems is a span of text, which is not friendly to the end users as well. In this paper, we proposed a framework for generating closed-domain QA data set in a semi-automatic manner, reducing human efforts. An organization-specific QA data set was created based on this framework. Additionally, we fine-tuned a traditional open-domain QA model on the newly created data set, and combine the output of this model with recently released language models to improve the naturalness of the response. Finally, we present our results and discuss the findings.**

***Index Terms*—closed-domain question answering, generative language models, natural language processing**

I. Introduction

The goal of a Question Answering (QA) system is to return accurate answers for given questions in natural language from a user. Generally speaking, two main types of QA systems are open-domain QA and closed-domain QA. In an open-domain QA, the users expect the system to be able to answer questions about any domain whose topics may not be related to each other. Wikipedia articles are the most popular data source for developing this kind of system. At the other end of the spectrum, closed-domain QA models are expected to answer questions about a much narrower domain, for example, questions about the internal documents of an organization. Due to the specialized and sensitive aspect of the documents, it is difficult to obtain the annotation necessary to train machine learning models for closed-domain QA systems.

There are three main paradigms proposed to solve the QA task. In DrQA [1], the authors proposed an open-domain QA system that contains a Document Retriever and a Document Reader. In this paradigm, the retriever is responsible for determining the small subset of relevant documents from the large corpus, while the reader is responsible for finding the answer spans within this subset of documents. With the advent of the Transformer architecture, a new paradigm, where the retriever and the reader could be jointly trained in an end-to-end manner, emerged. Some works which could be classified within this paradigm are R^3 [2], ORQA [3], REALM [4], and DPR [5]. Lastly, the third paradigm is composed of only a generative model, and the answers are no longer a span of text within a document. Instead, the knowledge is learned during training and embedded within the weights of models. OpenAI's ChatGPT is currently perhaps the most prominent system that falls into this category.

While generative large language models can store some factual knowledge in their parameters and produce responses in a natural tone, they have their own flaws. First, the answers from these models are not tractable - there is no simple way to determine the original source of the answers since the models fused all the knowledge together when next word prediction is the objective function. One problem that arises from this flaw is known as "hallucination" - the model produces plausible but incorrect answers. Second, these models are very expensive to train or even fine-tune, hence their knowledge cannot be easily updated to fit domain-specific needs.

Identifying these gaps, this work aims to develop a closed-domain QA system combining the advantages of traditional QA and generative language models while minimizing their flaws. In particular, our main contributions are:

- We proposed a framework for semi-automatically generating the data set needed to train a traditional QA. The framework reduced human efforts by using a generative language model for making the questions.
- Applying the framework, we constructed an organization-specific data set and fine-tuned a Reader model on this data set. The experiment results show that leveraging a fine-tuned open-domain QA model is more beneficial than fine-tuning from scratch.
- By combining the output of the Reader model with a

generative language model using the prompting technique, we show that domain-specific factual-grounded answers can be obtained without the need to fine-tune the generative language model. In addition, the naturalness of the responses was partly verified by human evaluators.

II. Related Work

A. Question Answering systems

Traditionally, QA research focuses on extracting answers from unstructured documents. In DrQA [1], the authors proposed a Retriever and a Reader as basic components for the QA system. Since then, much research has been dedicated to improving the performance of each component and the overall structure. More recently, BERTserini [6] fine-tuned a BERT model as the Reader model and combined it with the Anserini [7] toolbox to create a QA system over a large corpus of Wikipedia articles. In [8], the paragraphs in the same article are linked together, and a graph-based approach with a Graph Retriever and a Graph Reader was developed. Graph-based approaches for QA were further explored in [9], [10].

B. Large Language Models

Building on top of the Transformer architecture [11], large language models such as BERT [12] and its variants (e.g., [13], [14]) have to bring many successes to natural language processing applications. Two main types of language models are masked language models for natural language understanding [12] and auto-regressive language models for natural language generation [15]. As researchers scale the sizes of the auto-regressive models, some new properties appear, referred to as "emergent abilities" of language models [16]. This finding aligns with the "scaling law of language models" - scaling the models' sizes improves the models' qualities to some extent. As these models are getting larger and larger, it becomes more and more expensive to train or fine-tune them.

C. Memory-based Architectures

Memory-based architectures refer to architectures with an external memory supporting the neural networks model. This design shares some similarities with memory networks [17]. Some work has studied the utility of using external memory to support dialog or QA systems (e.g. [18], [19]). In this paper, the memory is the BM25 [20] index of the data set. Similar to [21], two key features of our memory are (i) human-readable, the memory is in natural language, and (ii) human-writable, it is possible to edit the document index.

D. Retrieve-and-Edit approaches

Some of the previous works have attempted to first retrieve the answers and then modify them for the final output (e.g., [22], [23]). This approach is known as the Retrieve-and-Edit approach [22]. Some successful applications of this approach include Machine Translation, Semantic Parsing, and Question Answering [21]. Our work can be seen as under the category of this framework.

III. Methods

Figure 1 shows the overall architecture of our approach. Broadly speaking, the architecture could be separated into three distinct components: Information Retrieval (IR), Reading Comprehension (RC), and Retelling (RT). The main purpose of the IR component is to find top-k relevant contexts to a question from a large pool of contexts. In our system, the contexts are closed-domain knowledge - paragraphs obtained from *JAIST's Handbook for Students* and some of the faculties' web pages. We use the terms *paragraph* and *context* interchangeably in this paper. Once relevant contexts are obtained, the RC component will read each of the relevant contexts and return the span of text with the highest probability of being the answer. Most of the time, the span of text is not a full sentence, so the last RT component will take this text span and paraphrase it to create a user-friendly answer. From another point of view, the extractive answer from the RC could be seen as an external knowledge source that directs the RT's generative model to generate fact-based answers.

As in the traditional QA system, the Retriever in IR and the Reader in RC need to be trained in a supervised manner. Constructing a data set such as the SQuAD data set [24] is a laborious (and expensive) task, so we resolved to a question generation model to aid the construction of our data set.

A. Models and algorithm

1) Data set construction: As mentioned above, we use a question generator and a pre-trained Reader model to assist us in the creation of a closed-domain QA data set. Fig. 2 shows the pipeline to generate the data set. Given a domain-specific paragraph, the Question Generator will generate a set of questions related to the paragraph. A pre-trained Reader then extracts the answers to the questions from the paragraph. Finally, a human is responsible for making sure that the questions and the answers are correct, as well as adding additional questions/answers if necessary.

For the Question Generator, we used a T5-for-question-generation model[1]. Given a context, this model was trained end-to-end to generate multiple questions simultaneously, as suggested in [25]. We utilized Haystack's Question Generation implementation for this model[2]. For the Reader, we used a fine-tuned BERT-based model, which is described in more detail in section III-A3.

We developed a web-based interface to help with the Human Verification step during the data set generation process. This interface is an extension of the QA-Annotator GitHub project[3] with additional features added by us: the ability to import SQuAD's style data set, the ability to mark a question/answer pair as correct, ability to quickly modify the auto-generated pairs. Our experience showed that using the interface significantly reduced the time needed for verifying the question/answer pairs.

[1]https://huggingface.co/valhalla/t5-base-e2e-qg
[2]https://docs.haystack.deepset.ai/docs/question˙generator
[3]https://github.com/impyadav/QA-Annotator

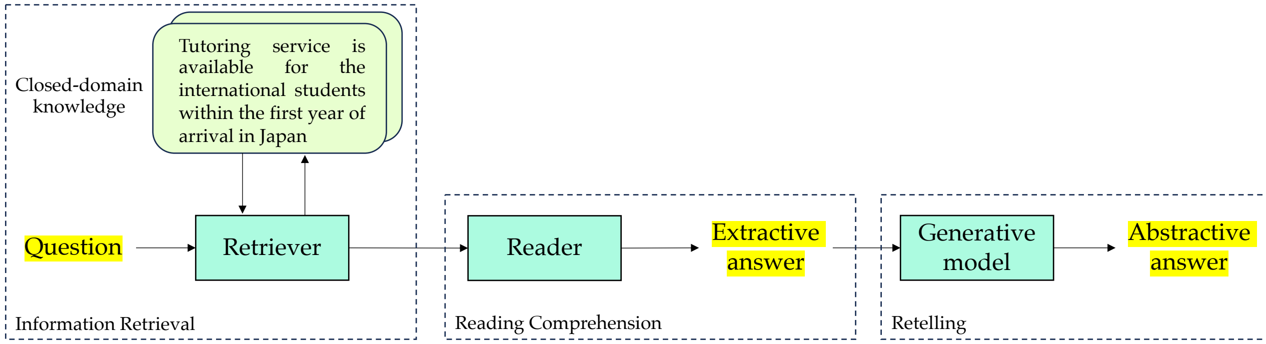

Fig. 1. Overall architecture of our proposed method

2) Retriever: For the Retriever, we used the classical BM25 algorithm [20]. This algorithm is fast and does not require the encoding of the query at inference time like Dense Passage Retrieval [5]. We briefly describe the BM25 algorithm below.

Given a query Q, in order to calculate the BM25 score of a document D, we first need to calculate the inverse document frequency (IDF) for each keyword q_i in Q:

$$IDF(q_i) = ln(\frac{N - n(q_i) + 0.5}{n(q_i) + 0.5} + 1) \tag{1}$$

where N is the total number of documents and $n(q_i)$ is the number of documents containing the keyword q_i. Then, the BM25 score of a document D given the query Q can be calculated according to the following formula:

$$\begin{aligned} score(D, Q) = \sum_{i=1}^{n} IDF(q_i) \cdot \\ \frac{f(q_i, D) \cdot (k_i + 1)}{f(q_i, D) + k_1 \cdot (1 - b + b \cdot \frac{|D|}{avgdl})} \end{aligned} \tag{2}$$

where $f(q_i, D)$ is defined as the number of times the keyword q_i occurs in D, $|D|$ is the number of words in D, $avgdl$ is the average document length of all the documents. k_i and b are free parameters.

3) Reader Model: The job of the Reader is to identify the correct span of text in the paragraph as the answer to a given query. For the Reader, we further fine-tune the BERT Reader models introduced by [26]. In particular, this reader is based on BERT [12] model with one difference: the final softmax layer over different answer spans is removed. We used BERTserini-base and BERTserini-large with 110M and 345M parameters, respectively. These models were fine-tuned on the SQuAD1.1 QA data set, which makes them suitable for our task.

4) Generative Language Models: For the RT component, we tested with two generative models, namely LLaMA (v1) [27] and Alpaca [28], [29]. For both models, we used the efficient 7B parameters version[4]. LLaMA is a family of models trained on trillions of tokens, and Alpaca is a fine-tuned version of LLaMA with instruction-following data aim to follow instructions.

Since the two models were meant for two different tasks, we created different prompt templates for them. In particular, let $< Q >$ be the question from the user and $< A >$ be the extracted answer from the RC component, then the prompt template for LLaMA is:

```
Transcript of a dialog, where the Student
interacts with an Assistant named AskJAIST.
AskJAIST is helpful, kind, honest, good at
writing, and never fails to answer the
Student's requests immediately and with
precision, using the Student's hints.

User: <Q> (hint: <A>)
AskJAIST:
```

And the prompt template for Alpaca is:

```
Below is an instruction that describes a
task. Write a response that appropriately
completes the request.
User: User asked the question: <Q>. The
answer is: <A>. Generate an appropriate
answer to the question.
AskJAIST:
```

B. Experiments

To conduct the experiments, first, we constructed the data set as described in section III-A1. Then, we trained Reader models on the newly generated data set to get the extractive answers for the questions. Lastly, we used the extractive answers as guidelines for the generative models. In section IV, we present the evaluation methods and main findings of our work.

IV. Results

A. Evaluation methods

The evaluation of our method contains two parts: evaluating the quality of the fine-tuned Reader models and evaluating the output of the generative model.

[4]https://github.com/ggerganov/llama.cpp

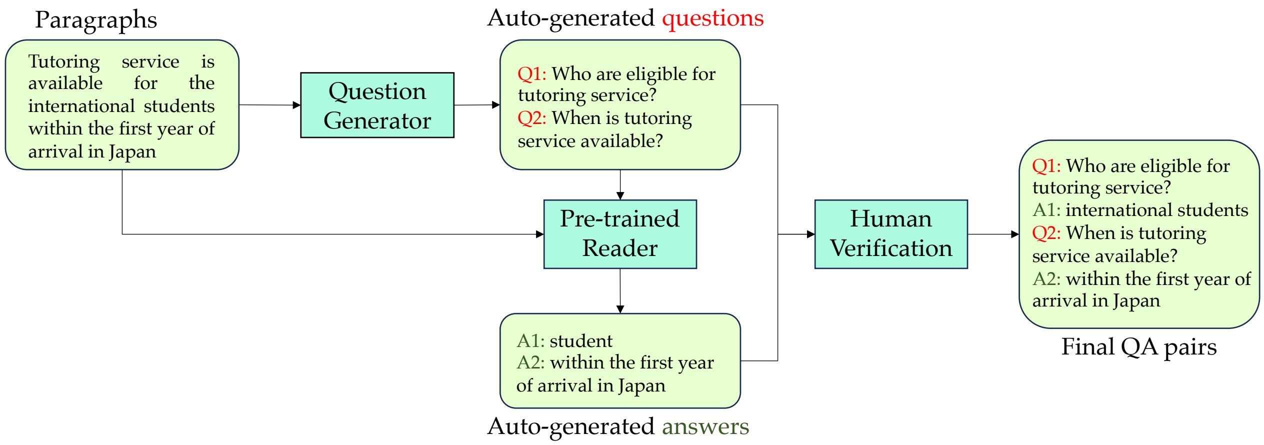

Fig. 2. Data set generation pipeline

TABLE I
EXAMPLE EXTRACTIVE RESPONSES WITH AND WITHOUT THE READER'S OUTPUT

Name	Content
Question Extractive answer	Where can I find a textbook? JAIST library
LLaMA	The textbook you're looking for is available at the Main Library.
LLaMA + prompt	JAIST library has textbooks.
Alpaca	Textbooks are typically available through your school's bookstore or online retailers such as Amazon, Barnes and Noble, Chegg, and others. You may also be able to find used textbooks on websites such as Craigslist or eBay.
Alpaca + prompt	JAIST library has a wide selection of textbooks in various languages, including English and Japanese. You can search for books by keyword or browse through the categories to find the book you need.

TABLE II
HUMAN EVALUATION METRICS FOR THE RETELLING COMPONENT. (1 IS HIGHLY DISAGREE, 5 IS HIGHLY AGREE)

Metric	Question for Evaluator	Scale
Correctness	Do you think this response reflects the question and the answer?	Yes or No
Naturalness	How natural is this response?	1 to 5
Specificity	How specific is this response?	1 to 5
Helpfulness	How helpful is this response?	1 to 5
Preference	Which of the two responses do you prefer more?	R1 (Alpaca) or R2 (LLaMA)

TABLE III
TYPES OF QUESTIONS IN THE DATA SET (QT: QUESTION TYPE, N: NUMBER OF OCCURRENCES)

QT	n	QT	n	QT	n	QT	n
What	729	When	35	How much	10	How often	3
Where	115	How long	33	What kind	0	What time	1
How	87	How many	30	Why	7	How far	1
Who	55	If	11	Which	3	Other types	406

To evaluate the quality of the Reader, we follow previous work (i.e. [1], [6]) and use exact match (EM), F1-score (at the token level), and recall (R - as defined in [6]) as the evaluation metrics on the development set.

To evaluate the quality of the RT component, we selected 20 random samples from the development set, generated the abstractive answers, and finally, five human students were asked to rate the responses according to 5 metrics: Correctness, Naturalness, Specificity, Helpfulness, and Preference. For the Correctness metric, the evaluators were asked to decide whether the abstractive answers correctly reflect the extractive answers to the questions. Table I shows an example of the abstractive responses (the evaluators were asked to evaluate the responses from the "LLaMA + prompt" and the "Alpaca + prompt", our system results are shown in bold) and table II shows the description of the human evaluation metrics.

B. Main Results

1) Data set generation: The first result of this paper is the construction of a new closed-domain QA data set. Applying the pipeline shown in Fig. 2, a total of 1535 question/answer pairs from 385 paragraphs were generated. The paragraphs were collected from *JAIST's Handbook for Students* and some of the faculties' web pages. Using simple prefix string matching, the number of questions belong to each type of question was calculated in Table III. Since each question was generated from a single paragraph (hence the answer for the question is within the paragraph), the generated data set contains single-hop questions.

We then divided the data set into training sets (308 paragraphs with 1198 question/answer pairs) and development sets (77 paragraphs with 337 question/answer pairs) for our

TABLE IV
RESULTS ON DEVELOPMENT SET
(HIGHER IS BETTER)

Model	EM ↑	F1 ↑	R ↑
BERT-base-uncased	0.59	10.44	17.59
BERT-base-uncased + Fine-tune	36.50	53.97	62.26
BERT-base Serini	**56.38**	68.79	69.71
BERT-base Serini + Fine-tune	55.49	**69.38**	**73.89**
BERT-large-uncased	0.0	7.80	17.29
BERT-large-uncased + Fine-tune	48.96	64.33	69.02
BERT-large Serini	53.71	68.76	69.01
BERT-large Serini + Fine-tune	**55.49**	**69.83**	**73.56**

TABLE V
HUMAN EVALUATION OF THE ABSTRACTIVE ANSWERS
(AVERAGE RESULTS OF FIVE EVALUATORS ± STANDARD DEVIATION; CORRECTNESS SCALE: 0 TO 1; OTHER METRIC SCALES: 1 TO 5)

Metric	LLaMA	Alpaca
Correctness	0.97 ± 0.03	0.92 ± 0.15
Naturalness	4.11 ± 0.89	4.15 ± 0.75
Specificity	3.81 ± 0.63	4.43 ± 0.59
Helpfulness	4.06 ± 0.56	4.14 ± 0.72

experiments.

2) Fine-tuned Reader result: Table IV shows the results of the Reader models. BERT-base and BERT-large indicate the BERT version we used. Serini indicates that the version was fine-tuned on the SQuAD QA data set. Fine-tune indicates that we additionally fine-tune the model on our own data set.

3) Results of generative answers: Table V shows the average human evaluation results on the answers generated by LLaMA and Alpaca models. In addition, on average, evaluators prefer the Alpaca's answers over the LLaMA's ones (57% preference for Alpaca).

V. DISCUSSION

The results from Table IV show that the fine-tuned models perform much better than the original in the case of BERT-base-uncased and BERT-large-uncased. This result, however, is still lower than Serini's versions (fine-tuned on the SQuAD data set). Lastly, fine-tuning domain-specific data sets on top of Serini's versions achieve the highest performances. Table IV suggests that it is beneficial to use models trained on open-domain QA for closed-domain QA tasks, and if possible, the models should be additionally fine-tuned on the closed-domain data set.

The example from Table I demonstrated that the generative models give very general responses without incorporating the extractive answer. With the extractive answer, the system is able to direct the models to generate more specific statements while combining its own internal knowledge. Furthermore, the extractive answer serves as an external source since it is linked to the original paragraph, making it possible for the end user to verify the generative models' outputs.

Our work contains some limitations. First, we found that sometimes the auto-generated questions do not make sense.

TABLE VI
EXAMPLE OF THE MODEL GENERATES INCORRECT ANSWER

Name	Content
Question	What is NOT allowed to continue to stay in Japan under the Immigration Control and Refugee Recognition Act?
Extractive answer	If you have a leave of absence
LLaMA + prompt	You are allowed to continue to stay in Japan under the Immigration Control and Refugee Recognition Act if you have a leave of absence.
Alpaca + prompt	If you are on a leave of absence, then you are not allowed to stay in Japan under the Immigration Control and Refugee Recognition Act.

This problem could potentially be overcome by modeling better question generation models. Another problem with the auto-generated questions is they usually contain the words/phrases in the paragraphs, while real-world questions by users usually differ from the paragraphs. The second limitation is that sometimes the language models generate plausible but incorrect answers. As an example, in Table VI, LLaMA generated an incorrect answer, while Alpaca generated a correct answer. Thirdly, the human evaluation method is somewhat subjective, a more objective large-scale version could be explored to better understand the system's outputs. Lastly, generative language models are known to be "unsafe", and this paper has not addressed the safety problems of using them yet.

VI. CONCLUSION AND FUTURE WORK

In this paper, we developed a closed-domain QA system from private organization documents, combining traditional QA and generative large language models. By semi-automatically data generation, we improved the efficiency of generating a closed-domain QA data set and reducing human work. Furthermore, we fine-tuned open-domain QA models on the newly created data set and showed the possibility of producing natural, domain-specific, and factual-grounded answers with generative language models.

Although we get relatively reasonable results from generative language models for our closed-domain QA system, several improvements are feasible for future work. First, the data set collected from this work is still relatively small, a larger data set could be collected to make the system more useful. Second, we could investigate the power of generative models to create useful dialog systems with long context, making them more appealing to end-users. Finally, safeguard methods to prevent the generative models from generating toxic or fake information need to be investigated.

ACKNOWLEDGMENT

We thank Huynh Lab and Okada Lab's members at JAIST for answering the human evaluation survey. This work was also partially supported by the Japan Society for the Promotion of Science (JSPS) KAKENHI (Grant Numbers 22H04860, 22H00536, 23H03506).

References

[1] D. Chen, A. Fisch, J. Weston, and A. Bordes, "Reading Wikipedia to Answer Open-Domain Questions," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Vancouver, Canada: Association for Computational Linguistics, Jul. 2017, pp. 1870–1879.

[2] S. Wang, M. Yu, X. Guo, Z. Wang, T. Klinger, W. Zhang, S. Chang, G. Tesauro, B. Zhou, and J. Jiang, "R3: reinforced ranker-reader for open-domain question answering," in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*. New Orleans, Louisiana, USA: AAAI Press, Feb. 2018, pp. 5981–5988.

[3] K. Lee, M.-W. Chang, and K. Toutanova, "Latent Retrieval for Weakly Supervised Open Domain Question Answering," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, 2019, pp. 6086–6096.

[4] K. Guu, K. Lee, Z. Tung, P. Pasupat, and M.-W. Chang, "REALM: retrieval-augmented language model pre-training," in *Proceedings of the 37th International Conference on Machine Learning*, ser. ICML'20, vol. 119. JMLR.org, Jul. 2020, pp. 3929–3938.

[5] V. Karpukhin, B. Oguz, S. Min, P. Lewis, L. Wu, S. Edunov, D. Chen, and W.-T. Yih, "Dense Passage Retrieval for Open-Domain Question Answering," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online: Association for Computational Linguistics, Nov. 2020, pp. 6769–6781.

[6] W. Yang, Y. Xie, A. Lin, X. Li, L. Tan, K. Xiong, M. Li, and J. Lin, "End-to-End Open-Domain Question Answering with BERTserini," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*. Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 72–77. [Online]. Available: https://aclanthology.org/N19-4013

[7] P. Yang, H. Fang, and J. Lin, "Anserini: Enabling the Use of Lucene for Information Retrieval Research," in *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. Shinjuku Tokyo Japan: ACM, Aug. 2017, pp. 1253–1256. [Online]. Available: https://dl.acm.org/doi/10.1145/3077136.3080721

[8] S. Min, D. Chen, L. Zettlemoyer, and H. Hajishirzi, "Knowledge Guided Text Retrieval and Reading for Open Domain Question Answering," Apr. 2020, arXiv:1911.03868 [cs]. [Online]. Available: http://arxiv.org/abs/1911.03868

[9] N. De Cao, W. Aziz, and I. Titov, "Question Answering by Reasoning Across Documents with Graph Convolutional Networks," 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)*. Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 2306–2317. [Online]. Available: https://aclanthology.org/N19-1240

[10] X. Zhang, A. Bosselut, M. Yasunaga, H. Ren, P. Liang, C. Manning, and J. Leskovec, "GreaseLM: Graph REASoning Enhanced Language Models for Question Answering," *International Conference on Representation Learning (ICLR)*, Jan. 2022.

[11] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is All you Need," in *Advances in Neural Information Processing Systems*, vol. 30. Curran Associates, Inc., 2017.

[12] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "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)*. Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 4171–4186. [Online]. Available: https://aclanthology.org/N19-1423

[13] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "RoBERTa: A Robustly Optimized BERT Pretraining Approach," Jul. 2019, arXiv:1907.11692 [cs]. [Online]. Available: http://arxiv.org/abs/1907.11692

[14] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *The Journal of Machine Learning Research*, vol. 21, no. 1, pp. 140:5485–140:5551, Jan. 2020.

[15] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov, and Q. V. Le, "XLNet: Generalized Autoregressive Pretraining for Language Understanding," in *Advances in Neural Information Processing Systems*, vol. 32. Curran Associates, Inc., 2019.

[16] J. Wei, Y. Tay, R. Bommasani, C. Raffel, B. Zoph, S. Borgeaud, D. Yogatama, M. Bosma, D. Zhou, D. Metzler, E. H. Chi, T. Hashimoto, O. Vinyals, P. Liang, J. Dean, and W. Fedus, "Emergent Abilities of Large Language Models," Oct. 2022, arXiv:2206.07682 [cs]. [Online]. Available: http://arxiv.org/abs/2206.07682

[17] J. Weston, S. Chopra, and A. Bordes, "Memory networks," in *3rd International Conference on Learning Representations (ICLR), Conference Track Proceedings*, 2015.

[18] M. Ghazvininejad, C. Brockett, M.-W. Chang, B. Dolan, J. Gao, W.-t. Yih, and M. Galley, "A Knowledge-Grounded Neural Conversation Model," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, Apr. 2018, number: 1. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/11977

[19] O. Khattab, C. Potts, and M. Zaharia, "Relevance-guided Supervision for OpenQA with ColBERT," *Transactions of the Association for Computational Linguistics*, vol. 9, pp. 929–944, 2021. [Online]. Available: https://aclanthology.org/2021.tacl-1.55

[20] S. Robertson and H. Zaragoza, "The Probabilistic Relevance Framework: BM25 and Beyond," *Foundations and Trends® in Information Retrieval*, vol. 3, no. 4, pp. 333–389, 2009. [Online]. Available: http://www.nowpublishers.com/article/Details/INR-019

[21] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, S. Riedel, and D. Kiela, "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," in *Advances in Neural Information Processing Systems*, vol. 33. Curran Associates, Inc., 2020, pp. 9459–9474.

[22] T. B. Hashimoto, K. Guu, Y. Oren, and P. S. Liang, "A Retrieve-and-Edit Framework for Predicting Structured Outputs," in *Advances in Neural Information Processing Systems*, vol. 31. Curran Associates, Inc., 2018.

[23] J. Gu, Y. Wang, K. Cho, and V. Li, "Search engine guided non-parametric neural machine translation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, May 2017.

[24] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, "SQuAD: 100,000+ Questions for Machine Comprehension of Text," in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Austin, Texas: Association for Computational Linguistics, Nov. 2016, pp. 2383–2392. [Online]. Available: https://aclanthology.org/D16-1264

[25] L. E. Lopez, D. K. Cruz, J. C. B. Cruz, and C. Cheng, "Simplifying Paragraph-Level Question Generation via Transformer Language Models," in *PRICAI 2021: Trends in Artificial Intelligence*, ser. Lecture Notes in Computer Science, D. N. Pham, T. Theeramunkong, G. Governatori, and F. Liu, Eds. Cham: Springer International Publishing, 2021, pp. 323–334.

[26] P. Yang, H. Fang, and J. Lin, "Anserini: Reproducible Ranking Baselines Using Lucene," *Journal of Data and Information Quality*, vol. 10, no. 4, pp. 1–20, Dec. 2018. [Online]. Available: https://dl.acm.org/doi/10.1145/3239571

[27] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, A. Rodriguez, A. Joulin, E. Grave, and G. Lample, "LLaMA: Open and Efficient Foundation Language Models," Feb. 2023, arXiv:2302.13971 [cs]. [Online]. Available: http://arxiv.org/abs/2302.13971

[28] R. Taori, I. Gulrajani, T. Zhang, Y. Dubois, X. Li, C. Guestrin, P. Liang, and Tatsunori B. Hashimoto, "Stanford alpaca: An instruction-following LLaMA model," 2023. [Online]. Available: https://github.com/tatsu-lab/stanford˙alpaca

[29] Y. Wang, Y. Kordi, S. Mishra, A. Liu, N. A. Smith, D. Khashabi, and H. Hajishirzi, "Self-Instruct: Aligning Language Models with Self-Generated Instructions," in *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Toronto, Canada: Association for Computational Linguistics, Jul. 2023, pp. 13 484–13 508.