Combined Transformers for Conversational Question Answering

Yuanzhi Zhu
yuazhu@ethz.ch
ETH Zürich

Bartosz Dzionek
bdzionek@ethz.ch
ETH Zürich & University of Edinburgh

Abstract

Conversational Question Answering (ConvQA) has been one of the most important question answering tasks. In most of the state-of-the-art models, a passage and the past Q&A session are sent as an input to a pre-trained transformer like BERT [1]. However, only few models are concerned how to combine and use the previous input as an additional information to the current question. In this paper, we proposed new methods for extracting information from the conversational history that can be later used as an input for BERT. Our idea is based on a framework using the combination of BERT and T5 [2]. A T5 model can be used to generate a rewritten question from the conversational history and the current question. We expect the encoder of the T5 model to learn the embedding of the conversational history. This embedding can be added as an extra information to improve the performance of BERT. Our entire code can be found on GitHub[1].

1 Introduction

1.1 Background

Question Answering (QA) has been a challenging task in natural language understanding. QA is a task of automatic answer extraction for questions asked in a natural language [3]. Thus, the key components in QA require the capability of understanding the question and the passage in which the question is generated.

ConvQA techniques form the building blocks of QA dialog systems, and the idea behind it is to let the machine generate an answer to a question based on the provided passage and historical conversation [4].

In ConvQA task, the system is asked a sequence of questions in a conversational manner. As shown in Figure 1, every question after the first one is built on what was asked before [5]. The dependence is often of the form of coreference or ellipsis [6]. ConvQA task can be based on a retriever using a large collection of documents of diversified topics, which make it an open domain conversational question answering task [7]. But the task was also explored in the simplified setting, where the system is directly given a reference text [8].

One potential solution to the ConvQA task is question rewriting (QR). QR is a technique that allows to convert a conversational QA task into non-conversational QA. This is done by generating questions that are semantically equivalent to the original one and contain the whole contextual information [6]. For instance, the question "How old would **she** be?" can be rewritten as "How old would **Jessica** be?" based on the conversation history. ConvQA task is also important as it is the base for many modern chatbots [9] [10].

Jessica went to sit in her rocking chair. Today was her birthday and she was turning 80. Her granddaughter Annie was coming over in the afternoon and Jessica was very excited to see her. Her daughter Melanie and Melanie's husband Josh were coming as well. Jessica had . . .

Q_1: Who had a birthday?
A_1: Jessica
R_1: Jessica went to sit in her rocking chair. Today was her birthday and she was turning 80.

Q_2: How old would she be?
A_2: 80
R_2: she was turning 80

Q_3: Did she plan to have any visitors?
A_3: Yes
R_3: Her granddaughter Annie was coming over

Q_4: How many?
A_4: Three
R_4: Her granddaughter Annie was coming over in the afternoon and Jessica was very excited to see her. Her daughter Melanie and Melanie's husband Josh were coming as well.

Q_5: Who?
A_5: Annie, Melanie and Josh
R_5: Her granddaughter Annie was coming over in the afternoon and Jessica was very excited to see her. Her daughter Melanie and Melanie's husband Josh were coming as well.

Figure 1. A sample of dataset for training (*CoQA*) [5]

1.2 Historical Development

The development of systems allowing a natural langauge conversation with computers can be traced back to *ELIZA* in 1966. It was a simple program based only on keywords and fixed rules [11]. In the next decades, various architectures were developed but they were still based on pattern matching. A notable example is *ALICE* from 2007, working on XML-based AIML files with patterns and templates [12]. The significant progress came around the year 2015 with the

[1] https://github.com/yuanzhi-zhu/CSNLP-Project-ETH

advent of large datasets and deep learning that outperformed traditional rule-based methods [4].

In 2018, the invention of BERT revolutionized many NLP tasks including ConvQA[1]. Most of the state-of-the-art models in the leaderboard of both *CoQA* and *QuAC* are variants of BERT [5] [13]. In [6], researchers adapted different models for QR including *AllenAI Coref* [14], *Transformer++*[15] and others. In [16], the researchers compared and combined several QR methods like *Transformer++*[15] and *QuReTeC* [17].

Except QR, there are also other methods to extract information from the conversational history. In [18], the topic for each round of QA is extracted as additional information to improve the overall model performance.

Vakulenko et al. (2021) combined QR methods of different types (sequence generation or term classification) and found it improved upon individual QR methods and achieved state-of-the-art retrieval performance on *CAsT 2019* [16]. However, they simply append terms from *QuReTeC* to the rewritten question produced by one of the generative models like *Transformer++* [15], which left huge improvement space for modification to better combine different ideas.

In this paper, we combine state-of-the-art QR approaches with answer extraction models and evaluate the performance on ConvQA task.

1.3 Goal

As part of the research project, we build a framework that consists of two modules. One module uses T5 [2] for Question Rewriting, and the second uses BERT [1] for Question Answering. The framework is shown in Figure 2 and described in more detail in the next sections.

2 Baseline Model

2.1 Task Formulation

We first define our task. Given a source passage P and a question Q, the task is to find an answer A that answers the question Q. The pipeline can be formulated as follows:

Input: The input of the model consists of a current question Q_i, given passage P, previous questions Q_{i-1}, ..., Q_{i-k}, and answers A_{i-1}, ..., A_{i-k} for turn i and some fixed window of size k.

Output: The answer A_i identified using a start span index and an end span index generated by the model.

Note that the CoQA dataset contains more generative questions (compared to SQuAD where most of the questions are extractive). In this paper, we only use the start and end index and that limits the final performance.

2.2 Datasets

For question rewriting, we use CANARD [19] and CoQAR [20], which are adapted from QuAC [13] and CoQA [5], respectively.

QuAC dialogs often switch topics while CoQA dialogs include more queries for details. QuAC focuses on information that could plausibly be in context material, and CoQA does not significantly cover unanswerable questions. Our analysis strongly implies that beyond yes/no questions, abstractive behavior is not a significant component in either QuAC or CoQA. As such, QuAC models can be trivially adapted to CoQA [21]. We will have deeper analysis on the predictions of these datasets in the following sections.

Evaluation for CoQA is the same as for SQuAD [22] and is in terms of exact match and F1 score. We analyze the predictions based on these metrics.

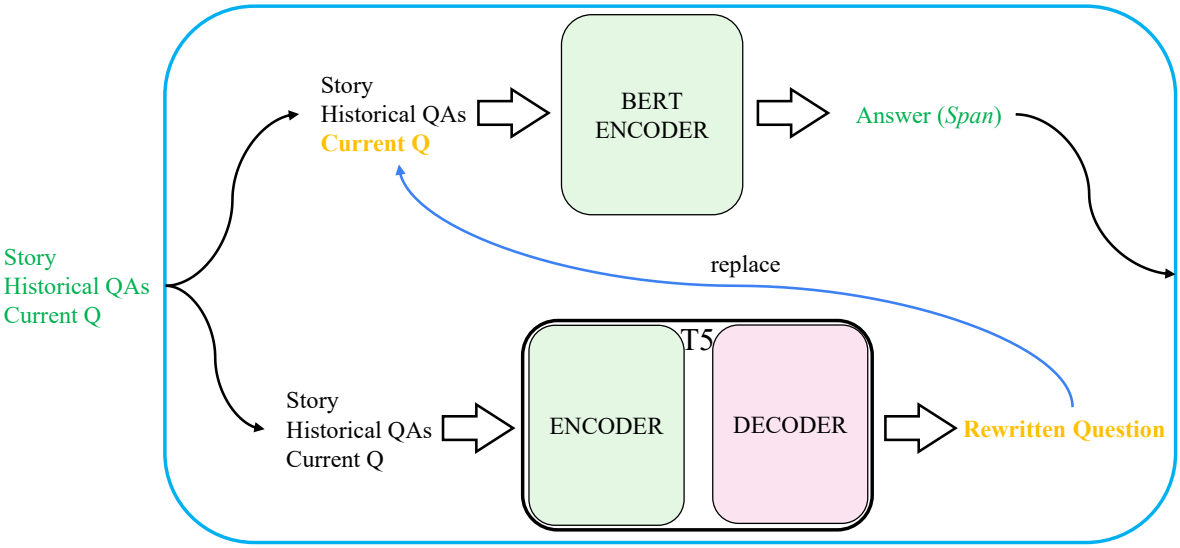

Figure 2. Our Framework

2.3 Fine-Tuning BERT as Baseline

The baseline here is the BERT base uncased model fine-tuned on the CoQA dataset. We show the scores for BERT trained for a different number of epochs in Table 1:

epochs	em	f1
1	65.0	74.9
2	**67.2**	**77.1**
4	66.9	77.0

Table 1. Baseline model for different epochs

From the above table, we use the 2 epochs fine-tuned BERT base model as our baseline for comparison.

The other important training parameters are listed here and we will keep them unchanged unless specified.

BERT model	batch size	max_seq_length	learning rate
base	2	512	1e-5

scheduler	warmup_steps	history_len	weight_decay
linear	2000	2	0.01

Table 2. Parameters for training

3 Model Prediction Analysis

3.1 Distribution of Answers

After fine-tuning BERT, we generated its predictions on the CoQA dev set. For evaluating answers, we used the official script provided by the authors of the dataset. The answers are evaluated by comparing them to human answers. The evaluation metric is exact match score and F1 score of word overlap.

We split the answers of BERT into three categories according to the evaluation metrics:

- *correct* answers are those with exact match with one of human answers,
- *partially* correct answers do not have exact match but have non-zero F1 score,
- *incorrect* answers have zero F1 score.

The dataset is conversational with the maximum turn of 25. It is easy to find that the depth of conversations in the dataset starts to decrease after turn 10 and is rare at depth larger than 20. Hence, we took only the first 20 turns and we normalized them. The result is Figure 3. We see that in the first turn the percent of correct answers is 68.8%, and for turns 2-20 it is on average 58.3% (95% confidence interval = 57.5% – 59.1% [2]). In the first turn, incorrect answers comprise 10.4% of answers. In turns 2-20, they comprise on average 17.3% (95% confidence interval = 16.4% – 18.3% [3]).

[2]Computed with t-test, but we also obtained almost identical results with bootstrapping.

[3]See footnote 1.

Such a difference between the first turn and the rest means BERT underperforms in the multi-turn setting. This allows us to experiment with question rewriting from T5 to decrease the gap between single-turn and multi-turn QA.

Even [18] suggest that with all history, the open retrieval QA model performs better than with only the rewritten question. We argue here that with the correct rewritten question, we can turn multi-turn setting into single-turn and improve the performance.

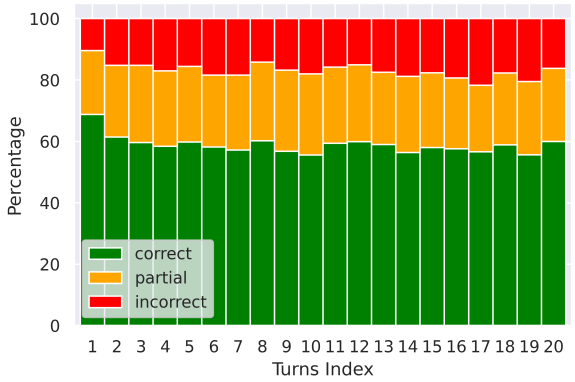

Figure 3. Normalized correctness for turns ≤ 20.

3.2 Failure Case Analysis

As a next step, we looked closer into the answers to find patters connected to incorrect answers. Since we want to get an improvement in the conversational setting, we will only consider examples from turn 2 onward.

We first define the answers with exact match score equal 1 as correct, f1 score equal 0 as incorrect and the rest as partial. We found the ratio of correct/partial/incorrect answers for answers composed of 1 up to 10 words (longer answers comprised very small part) and presented it in Table 3. For answers that are single words it is much easier to get an exact match with the ground truth. As the answers start to be longer, it is less common to get an exact match but more common to have a word overlapping with ground truth answers, thus being partially correct. It is worth noting that ratio of incorrect answers is much bigger for single-word answers (0.22). For comparison, there are 205 single-word answers to a question in the first turn and their incorrect ratio is 0.13. Given that single words have the highest count among the generated answers, they constitute a significant bottleneck hurting performance of the model.

Finally, we checked the correctness with respect to the domain of a question. The public part of CoQA consists of five different domains, every domain appears in approximately similar number of questions. Again, we compared the number of incorrect answers in the first turn against the next

#words	count	correct	partial	incorrect
1	**3518**	**0.70**	0.08	**0.22**
2	1360	0.66	0.21	0.13
3	880	0.56	0.29	0.14
4	511	0.45	0.46	0.09
5	326	0.37	0.51	0.13
6	227	0.34	0.56	0.10
7	138	0.26	0.61	0.13
8	114	0.21	0.66	0.13
9	91	0.26	0.59	0.14
10	46	0.15	**0.74**	0.11

Table 3. Distribution vs length of the answer. (turn ≥ 2)

turns	domain				
	cnn	gutenberg	mctest	race	wikipedia
turn = 1	0.08	0.19	0.10	0.10	0.05
turn ≥ 2	0.15	0.19	0.18	0.20	0.14
abs diff	**0.07**	**0.00**	**0.08**	**0.10**	**0.09**

Table 4. Ratio of incorrect answers.

turns. The result is shown as Table 4. We observed that 4 domains have very similar gap between the first turn and other turns. However, for the *gutenberg* part, we got approximately the same ratio. It might be useful to examine questions in this domain more closely as our goal is to minimize the gap for other four domains.

3.3 Limitations of CoQA

During exploration of the answers, we noted that CoQA under-evaluates some of our answers. That happens when the generated answer is more detailed than ground truth or vice versa. For example:

- model: *"in a barn near a farm house"*, CoQA: *"in a barn"* (EM: 0.0, F1: 0.71)
- model: *"Joe Fontana"*, CoQA: *"Detective Joe Fontana"* (EM: 0.0, F1: 0.8)

Fortunately, this effect is mitigated by a high F1 score. However, final evaluation might be noisy due to the preference of human contributors to shorter or more detailed answers.

The worse problem is that F1 score does not capture sematics of the answer. This can lead perfect answers to be classified as incorrect (e.g. model: *"fifty"*, CoQA: *"50"*, (EM: 0.0, F1: 0.0)), and adds more noise to the final evaluation.

4 Our Framework

Our framework is a combination of BERT [1] and T5 [2]. As you can see in Figure 2, the common input to both models is the story and conversations (including the historical question answers and current question). These inputs are first sent to a T5 small model to generate the rewritten question. Then, we can replace the original question or even the original conversation with the rewritten question to form updated inputs for BERT.

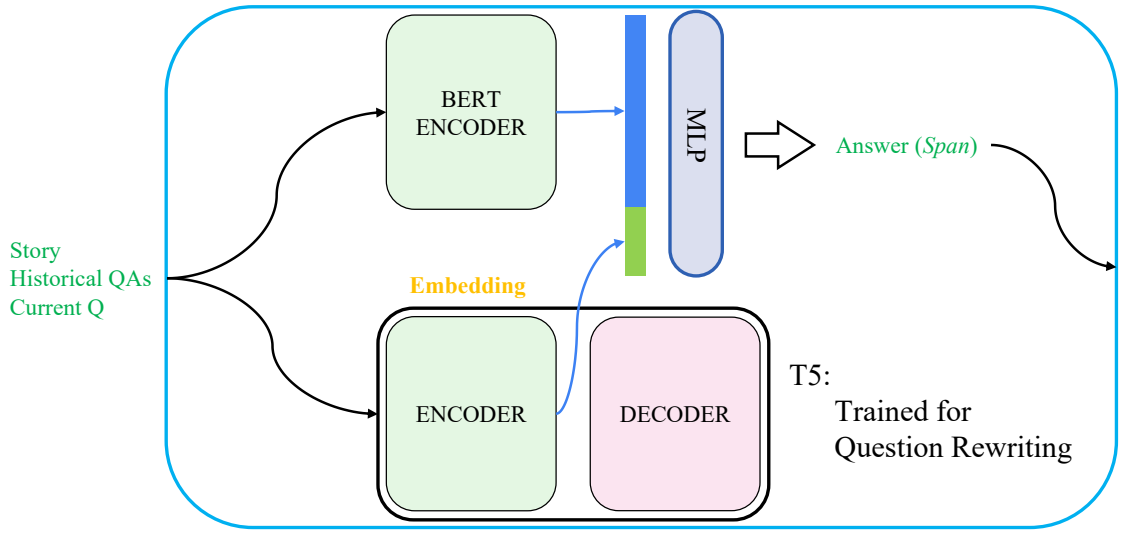

Figure 4. Our Framework (new)

We noted that the T5 encoder and the BERT base have exactly the same architecture. We believed that getting just the encoding of T5 will be better than the rewritten question. This is because the encoding needs to go through the decoder and that can lead to information loss. In [23], the authors also derived the conclusions that for downstream tasks, utilizing only the T5 encoder is better than using the full T5 encoder-decoder model; and encoder-only mean (pooled) embedding is better than encoder-only first (special token embedding [CLS]). As a result, we proposed another way to combine information from these two models, as illustrated in Figure 4. The raw output of BERT and T5 encoder are concatenated into a long vector passed to MLP layers. In our experiment, we will try both the token embedding and the sentence embedding from the T5 encoder.

The final output from BERT is the answer span, and we expect an improvement of more than 5% according to the analysis in section 3.

4.1 T5 Component

We first fine-tuned BERT base on CoQA dataset and T5 small on a mixed dataset of CANARD and CoQAR. The dev set of both CoQAR and CoQA is the same, which makes it easier to build our framework.

The BERT version with the best performance in our case is BERT base trained on CoQA training dataset for 4 epochs. For T5 question rewriting model, we did several experiments because the performance of question rewriting is the key in our framework. The result is shown in Table 5

It is interesting that even without the story, the model can generate a good rewritten question. This is because one can do coreference using only the historical conversation. It is also worth mentioning that the last two experiments show that with the additional training samples from CoQAR, the BLEU score on CANARD is not improved. Indeed, CANARD is adapted from QuAC and CoQAR is adapted from CoQA, and these two datasets have different focus.

According to this result, we select the last model as the component in our framework.

model	story	batch size	hist size	dataset	BLEU CoQA	BLEU CANARD
$T5_{small}$	w/o	16	20	canard	0.3068	0.4665
$T5_{small}$	w/	16	20	canard	0.3304	0.4866
$T5_{base}$	w/	4	20	canard	0.3563	**0.5065**
$T5_{base}$	w/o	4	20	canard	0.3276	0.4953
$T5_{small}$	w/	4	3	canard	0.3261	0.4855
$T5_{small}$	w/	16	3	canard	0.3281	0.4881
$T5_{small}$	w/	16	3	mixed	**0.4039**	0.4756

Table 5. T5 experiments

4.2 BERT Component

Here the BERT part is nothing but the same BERT base model as baseline. However, since the input has changed, we have to repeat the fine-tuning of the model to get desired results. Otherwise, BERT becomes a bottleneck. But what we want is to improve the gap between the first turn and next turns.

Besides, the additional layers may also be fine-tuned to extract better answer from the BERT's raw output.

5 Results

5.1 Results with a rewritten question

At the beginning, we used the fine-tuned T5 model to generate rewritten questions, and use them as new inputs for the fine-tuned BERT model that extracted answers. When we implemented this design, we found that it is rather inconvenient to use the whole pipeline with T5 in real time. Instead, we generated a new dataset with rewritten questions, named *coqa-v1.0-append_with_T5.json* in our GitHub repository, that is fed into the BERT model. The preliminary results are shown below:

Inputs	em	f1
Baseline	67.2	77.1
Replaced with Rewritten Q	61.4	71.5
Concatenated with Rewritten Q	61.8	71.9

Table 6. Preliminary Results without Fine-Tuning

Looking at the result, we observe a much worse score after using the new inputs. To mitigate this issue, we decided to fine-tune the BERT model with the new formed inputs. However, to fine-tune BERT again, we cannot use the T5 model fine-tuned on the mixed dataset. This is because the CoQA train dataset will be seen by the model in advance. So we select the T5 base model fine-tuned only on CANARD dataset. On top of that, we investigated that some of the rewritten questions were not a well-formed questions(3%). In this case, we used the original question as a rewritten question. After the adjustment, we get updated results as in Table 8.

Inputs	em	f1
Baseline	67.2	77.1
Replaced with Rewritten Q (4 epochs)	64.8	74.8
Replaced all (2 epochs)	62.4	71.8
Concatenated w/o history (2 epochs)	63.8	73.5
Concatenated with Rewritten Q (4 epochs)	66.5	76.7
Concatenated with Rewritten Q (2 epochs)	**67.3**	**77.2**

Table 7. Further Results with Fine-Tuning

We also tried replacing all historical conversations and the current question with just the rewritten question from

T5 (replace all); and tried a concatenation of the rewritten question to the current question but without the historical conversation (Concatenated w/o history).

We expected to get a higher score with these two approaches, but it was the opposite. The reason could be that:

- replace: T5 could not generate a well-rewritten questions, which led to an information loss.
- append: the BERT model gets two repeated questions to answer, which introduces more noise.

5.2 Results with T5 embedding

As proposed above, we also tried to concatenate the embedding extracted from the T5 encoder (instead of the decoded rewritten question). There are two types of embeddings in our case: the token embedding with size

$$(batch,\ max\_seq\_length,\ hidden\_size);$$

and the pooled embedding with size

$$(batch,\ hidden\_size).$$

Inputs	em	f1
Baseline	67.2	77.1
embedding (2 epochs)	67.3	77.2
embedding (4 epochs)	**67.4**	**77.2**
pooled embedding (2 epochs)	66.7	76.6
pooled embedding (3 epochs)	66.9	76.8

Table 8. Results with T5 Embedding

These further results are still barely above the baseline, far away from our expectation. The reason behind this could be that the vocabulary of BERT and T5 are different and the way they process the tokens are different (tokenizer). In this study, we have not made it such that the form of these two models are fully the same. This may lead to difficulty for BERT to understand T5's embedding.

6 Conclusion and Future Work

In this work, we did a fair number of experiments to investigate the performance of different BERT settings on Conversational Question Answering.

We first analyzed the distribution of answers for the standard BERT model and found that in general the first turn's F1 score is higher than for the rest turns of conversation. Inspired by this, we planned to inject the rewritten question from the T5 (generative) model to help transform the conversational QA setting into a vanilla QA setting and improve the overall scores by up to 10 percents.

To do this, we proposed two different methods. The first is to combine the rewritten question from T5 with original input to BERT (as in Figure 2), and the other is to get the embedding from a T5 encoder and combine it with embedding of BERT output and send them concatenated to MLP layers (as in Figure 4).

However, through extensive experiments (we omit many more failing ones), we show that we have not gained the expected improvement. The most possible reason behind could be that the power of T5 is limited and the output from T5 is not always helpful.

In the future, we still believe there is a way to improve the conversational QA performance (like joint training of two models) and hope this conversational to one-turn transformation can benefit all conversational QA tasks.

Due to the limited time and shrinking group size, we did all the experiments only on the CoQA dataset and did not explore the dataset as much as we wanted. We would like to encourage others to fine-tune this model and try it also on QuAC and give more comprehensive analysis on both the dataset itself and the predictions given by the model.

Since we also had relatively small computing resources, it might be useful to investigate the impact after training for more epochs. Future work can also look into using different transformers, as well as better evaluation metrics that understand the semantics of the answer. Moreover, it would be beneficial to generate answers in case it is not possible to extract a span that fully answers a particular question.

7 Acknowledgement

We thank prof. Sachan for offering this course and the chance to do this project. Yuanzhi is also grateful for keeping his assignment grade from last year, which made this project possible. We also thank the whole TA team, especially Shehzaad for his insightful remarks and suggestions.

References

[1] 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), pp. 4171–4186, Association for Computational Linguistics, June 2019.

[2] 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," *Journal of Machine Learning Research*, vol. 21, no. 140, pp. 1–67, 2020.

[3] P. Cimiano, C. Unger, and J. McCrae, *Ontology-Based Interpretation of Natural Language*. 2014.

[4] M. Zaib, W. E. Zhang, Q. Z. Sheng, A. Mahmood, and Y. Zhang, "Conversational question answering: A survey," *arXiv preprint arXiv:2106.00874*, 2021.

[5] S. Reddy, D. Chen, and C. D. Manning, "Coqa: A conversational question answering challenge," *Transactions of the Association for Computational Linguistics*, vol. 7, pp. 249–266, 2019.

[6] R. Anantha, S. Vakulenko, Z. Tu, S. Longpre, S. Pulman, and S. Chappidi, "Open-domain question answering goes conversational via question rewriting," *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2021.

[7] D. Chen and W.-t. Yih, "Open-domain question answering," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: Tutorial Abstracts*, (Online), pp. 34–37, Association for Computational Linguistics, July 2020.

[8] C. Qu, L. Yang, M. Qiu, W. B. Croft, Y. Zhang, and M. Iyyer, "Bert with history answer embedding for conversational question answering," in *Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval*, pp. 1133–1136, 2019.

[9] L. Cui, S. Huang, F. Wei, C. Tan, C. Duan, and M. Zhou, "SuperAgent: A customer service chatbot for E-commerce websites," in *Proceedings of ACL 2017, System Demonstrations*, (Vancouver, Canada), pp. 97–102, Association for Computational Linguistics, July 2017.

[10] M. Qiu, F.-L. Li, S. Wang, X. Gao, Y. Chen, W. Zhao, H. Chen, J. Huang, and W. Chu, "AliMe chat: A sequence to sequence and rerank based chatbot engine," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, (Vancouver, Canada), pp. 498–503, Association for Computational Linguistics, July 2017.

[11] J. Weizenbaum, "Eliza—a computer program for the study of natural language communication between man and machine," *Commun. ACM*, vol. 9, p. 36–45, Jan. 1966.

[12] B. Shawar and E. Atwell, "Chatbots: Are they really useful?," *LDV Forum*, vol. 22, pp. 29–49, 01 2007.

[13] E. Choi, H. He, M. Iyyer, M. Yatskar, W. tau Yih, Y. Choi, P. Liang, and L. Zettlemoyer, "Quac : Question answering in context," 2018.

[14] K. Lee, L. He, and L. Zettlemoyer, "Higher-order coreference resolution with coarse-to-fine inference," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, (New Orleans, Louisiana), pp. 687–692, Association for Computational Linguistics, June 2018.

[15] S. Vakulenko, S. Longpre, Z. Tu, and R. Anantha, "Question rewriting for conversational question answering," in *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, WSDM '21, (New York, NY, USA), p. 355–363, Association for Computing Machinery, 2021.

[16] S. Vakulenko, N. Voskarides, Z. Tu, and S. Longpre, "A comparison of question rewriting methods for conversational passage retrieval," 2021.

[17] N. Voskarides, D. Li, P. Ren, E. Kanoulas, and M. de Rijke, *Query Resolution for Conversational Search with Limited Supervision*, p. 921–930. New York, NY, USA: Association for Computing Machinery, 2020.

[18] V. Adlakha, S. Dhuliawala, K. Suleman, H. de Vries, and S. Reddy, "Topiocqa: Open-domain conversational question answering with topic switching," 2021.

[19] A. Elgohary, D. Peskov, and J. Boyd-Graber, "Can you unpack that? learning to rewrite questions-in-context," in *Empirical Methods in Natural Language Processing*, 2019.

[20] G. L. Quentin Brabant and L. Rojas-Barahona, "Coqar question rewriting on coqa," To be published in LREC2022, 2022.

[21] M. Yatskar, "A qualitative comparison of coqa, squad 2.0 and quac," *arXiv preprint arXiv:1809.10735*, 2018.

[22] P. Rajpurkar, R. Jia, and P. Liang, "Know what you don't know: Unanswerable questions for squad," 2018.

[23] J. Ni, G. H. Ábrego, N. Constant, J. Ma, K. B. Hall, D. Cer, and Y. Yang, "Sentence-t5: Scalable sentence encoders from pre-trained text-to-text models," *arXiv preprint arXiv:2108.08877*, 2021.