TREC 2020 Notebook: CAsT Track

Sheng-Chieh Lin, Jheng-Hong Yang, Jimmy Lin
David R. Cheriton School of Computer Science
University of Waterloo

Abstract

This notebook describes our participation (h2oloo) in TREC CAsT 2020. We first illustrate our multi-stage pipeline for conversational search: sequence-to-sequence query reformulation followed by an *ad hoc* text ranking pipeline; then, detail our proposed method for canonical response entry. Empirically, we show that our method effectively reformulates conversational queries considering both historical user utterances and system responses, yielding final ranking result 0.363 and 0.494 in terms of MAP and NDCG@3 respectively, which is our best submission to CAsT 2020.

1 Introduction

Table 1: CAsT2020 examples.

Turn (i)	Conversation utterances (u_i) and system responses (r_i)
u_1	What are some interesting facts about bees?
r_1	Fun facts about bees: 1 Honeybees are the only insect that produces food eaten by humans ... 5 **Honey** never spoils.
u_2	Why doesn't it spoil?
r_2	Honey doesn't spoil like other foods and even if it has turned cloudy, it's still safe to eat ...
...	
u_1	Which is the biggest commercial plane?
r_1	The airliner that holds the current record of highest passenger capacity is the **Airbus A380** ...
u_2	What are its operational costs?
r_2	The Airbus A380, the largest passenger jet, costs between $26,000 and $29,000 per hour...
...	

Recently, conversational search grabs the attention of researchers due to its potential applications (e.g., smart speakers). Last year, TREC Conversational Assistant Track (CAsT 2019) [3] took a step toward conversational IR by building a conversational passage retrieval dataset for practitioners. However, the conversational queries of CAsT 2019 is made with an innate assumption that users' utterances only depend on their previous utterances. The assumption limits generalization capabilities of the models built upon the dataset since in real applications, users may also give utterances based on system response (see the examples in Table 1). This year, the organizers of CAsT 2020 takes the scenario into consideration and newly constructed a more comprehensive dataset.

In this paper, we focus on our participation in canonical response entry using T5 [9] as our query reformulation (QR) model. Although many works [7, 10, 12] have demonstrated the effectiveness of pretrained sequence-to-sequence models on the task of query reformulation. However, all of them are based on CAsT 2019 dataset and do not take system responses into account. Thus, in this work, we first highlight the challenges of using sequence-to-sequence models for QR in canonical response entry. Then, we propose our method and make a comparison with other possible solutions. We empirically demonstrate that our proposed method effectively reformulates queries when taking system response into consideration.

2 Methodology

In this section, we first describe our multi-stage pipeline for conversational search (CS), including the modules for query reformulation, passage retrieval and passage re-ranking. Secondly, we will describe our approach to canonical response entry, which is the new task in CAsT 2020 dataset where users' utterance could depend on both historical user utterances and system responses as shown in Table 1.

2.1 Problem setting

Given a sequence of conversational utterances $u^s = (u_1, \cdots, u_i, u_{i+1}, \cdots)$ and the corresponding system response $r^s = (r_1, \cdots, r_i, r_{i+1}, \cdots)$ for a topic-oriented session $s \in S$, where S is the set of all dialogue sessions and u_i (or r_i) stands for the i-th utterance (or system response) ($i \in \mathbb{N}^+$) in the session. For each turn i, the goal of this task is to find a set of relevant passages $\mathcal{P}_i$, for each turn's user utterance u_i that satisfies the information needs with the context in previous turns $ctx_{<i} = u_1 \oplus r_1 \oplus \cdots, u_{i-1} \oplus r_{i-1}$, where $\oplus$ denotes the operation of text concatenation. For an IR system, let $P(R = 1 \mid q, p)$ denote the probability conditioned on a query-passage pair (q, p), where $R = 1$ denotes that a passage $p \in \mathcal{C}$ is relevant to a query q (otherwise, $R = 0$).

2.2 Multi-stage Pipeline for Conversational Search

Following [7], we factorize the probability of retrieving a relevant passage $p \in \mathcal{P}_i$ for each turn i. For the problem setting of CAsT 2020, we replace the information set for query reformulation model by $\{u_i, ctx_{<i}\}$ that comprises the utterances and responses.

$$P(R = 1 \mid \{u_i, ctx_{<i}\}, p) = P(R = 1 \mid q_i, p)\, P\left(q_i \mid \{u_i, ctx_{<i}\}\right). \tag{1}$$

With this formulation, CS can be approximated by separately maximizing the probabilities of (a) a relevance prediction model $P(R = 1 \mid q_i, p)$ and (b) a query reformulation model $P(q_i \mid \{u_i, ctx_{<i}\})$. Thus, the goal of a query reformulation model is to reformulate a raw conversational user utterance u_i in each turn i into a clear and informative query q_i for the relevance prediction model [13].

Query reformulation. Following the previous works [7], we adopt text-to-text-transfer transformer (T5) [9] as our query reformulation model. Specifically, we adopt pretrained T5 model checkpoints from [9] and fine-tune them on CANARD dataset [2], which is a conversational query rewriting dataset. In CANARD, for each conversation turn, we concatenated historical queries and answers as source texts and use human annotated queries as target texts. Using the paired source and target texts of all conversation turns, the query reformulation models are trained by the standard sequence-to-sequence scheme: cross-entropy loss and teacher forcing. Then, we directly transfer the fine-tuned weights for inference in CAsT dataset:

$$\hat{q_i} = \text{Seq2Seq}\left(ctx_{<i} \oplus u_i, \hat{\theta}\right) \tag{2}$$

where $\hat{\theta}$ denotes the fine-tuned weights from CANARD dataset.

Passage retrieval. Our passage retrieval model facilitates first-stage candidate elicitation that takes reformulated queries to search for relevant passages in the passage collection. We use the tightly-coupled teacher distillation proposed by Lin et al. [6] to incorporate dense representations of dual encoders and sparse representations from BM25. Both our teacher model, ColBERT [5], and student model, dual encoders with BERT-base, are trained on the MS MARCO passage ranking dataset [1]. The dense representation indexing and searching are facilitated by Faiss [4] in which we use flat index and inner product as our metric for searching. As for our sparse representation, we use Anserini [11] to calculate BM25 matching scores. Finally, we use the hybrid scheme proposed in [6] to fuse the similarity scores of dot products from dense representations and BM25 matching scores from sparse representations.

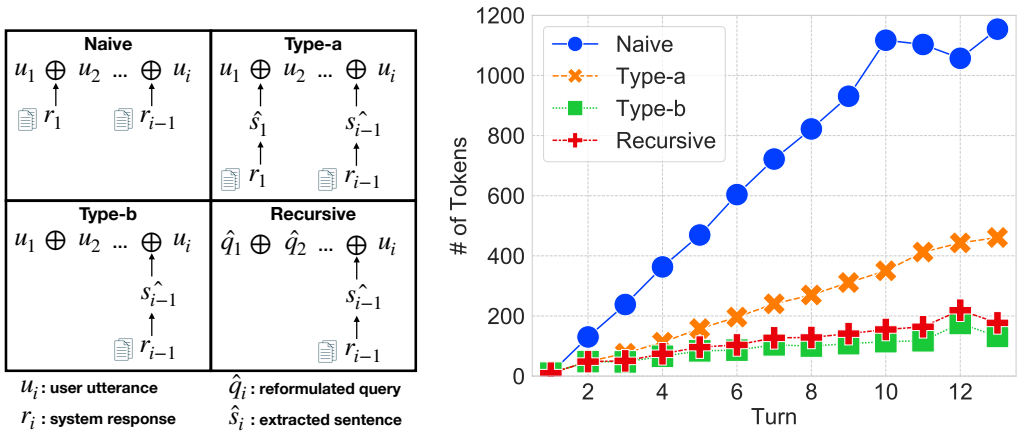
Passage re-ranking. In our multi-stage pipeline, we use T5 as our text re-ranking model. Initiated from the checkpoints in [9], we fine-tune the T5 model for paired (query, passage) text relevance ranking. We adopt the training scheme proposed by [8] to leverage the implicit knowledge in pretrained tokens via recasting the passage ranking task under text-to-text framework. To be more specific, we use "true" and "false" tokens as our relevance target tokens and calculate the relevance

ranking according to the value of the "true" logit, which is softmax normalized among the pair of tokens. Our re-ranking model training is also based on the MS MARCO passage ranking dataset. During inference, we take reformulated queries and concatenate them with top-1000 relevant passages returned by our passage retrieval model.

2.3 Canonical Response Entry

To include the information from system response, we can naively concatenate all the historical user utterances and system responses as the context for query reformulation (see Naive in Figure 1); however, this approach causes problems such as:

1. Long processing time: system responses are passages which normaly contains 100-150 words in average. Thus, including all system responses to the context lead to long input texts.
2. Performance degradation: the whole passages from system responses may include unrelated context, and the noisy context raises difficulties when conducting query reformulation.



(a) Query reformulation inference types (b) Average # of tokens input to T5 by turn depth

Figure 1: The comparison of different query reformulation methods

Information extraction from system response. To address the problems, we propose to first extract a representative sentence from system responses, that is most related to the dialogue, and then append the extracted sentence to the context (see Type-a in Figure 1). Formally speaking, given a system response r_i consisting a number of $n(r_i)$ sentences, the response is represented as a tuple $\left(s_i^1, \ldots, s_i^{n(r_i)}\right)$, where s_i^k denotes the k-th sentence in r_i and we seek to reduce the long context by replacing $ctx_{<i}$ with $ctx^a_{<i} = u_1 \oplus \hat{s}_1 \oplus \cdots, u_{i-1} \oplus \hat{s}_{i-1}$, where $\hat{s}_{i-1}$ denotes the sentence extracted from the $(i-1)$-th response r_{i-1}.

From Table 1, out first observation is that $\hat{s}_{i-1}$ is likely to have textual overlaps with the next utterance u_i or the answer of its previous utterance u_{i-1}. Looking at the first example, the utterance u_2 continues the topic in the last sentence of its previous response r_1, while from the second conversation, the pronounce in u_2 refers to Airbus A380 in the first sentence of r_1, which is the answer for u_1. According to the observation, we formulate the function of extracting sentence from r_{i-1} as finding the sentence most related to u_i (or u_{i-1}):

$$\hat{s}_{i-1} = \begin{cases} \underset{1\le x\le n(r_{i-1})}{\text{argmax}} \ \text{Sim}(s^x_{i-1}, u_i) & \text{if} \ \underset{1\le x\le n(r_{i-1})}{\max} \ \text{Sim}(s^x_{i-1}, u_{i-1}) \neq 0; \\ \underset{1\le x\le n(r_{i-1})}{\text{argmax}} \ \text{Sim}(s^x_{i-1}, u_{i-1}) & \text{if} \ \underset{1\le x\le n(r_{i-1})}{\max} \ \text{Sim}(s^x_{i-1}, u_i) \neq 0; \\ \emptyset, & \text{otherwise.} \end{cases} \tag{3}$$

where $\text{Sim}(\cdot,\cdot)$ is the similarity measurement of two texts. For simplicity, we use the number of keyword matching as the similarity function.[1] If there is no keyword matching between any sentence

[1]For each input text, we define keywords as the word with noun, verb or adjective POS tags.

Table 2: Experimental results

Cond.	Query reformulation		Retrieval (dense+sparse)			Re-ranking (T5-3B)		BLEU	Run
	Model(T5)	Inference	R@1000	MAP	NDCG@3	MAP	NDCG@3		
Manual	-	-	0.840	0.324	0.463	0.459	0.613	100.00	-
1	base	Query-only	0.668	0.225	0.343	0.330	0.452	63.75	Run4
2	base	Type-b	0.661	0.216	0.337	-	-	63.12	-
3	base	Recursive	0.684	0.220	0.328	-	-	62.18	-
4	large	Query-only	0.696	0.238	0.360	-	-	64.33	-
5	large	Type-a	0.708	0.239	0.364	-	-	64.43	-
6	large	Type-b	0.697	0.238	0.358	0.345	0.480	64.64	-
7	large	Recursive	**0.724**	**0.250**	**0.367**	**0.363**	**0.494**	**65.23**	Run2

in r_{i-1} and u_i (or u_{i-1}), we do not include any sentence from r_{i-1}. Observing Figure 1(b) (Naive vs Type-a), replacing the whole passages with their representative sentences significantly reduces the number of tokens for query reformulation to an acceptable level.

Recursive inference. While aforementioned method has already reduced the length of input texts for query reformulation, in this work, we further seek to reduce the input length without losing of context information. Intuitively, at turn i, the most important response for reformulating u_i is the previous response r_{i-1} (or $\hat{s}_{i-1}$). Thus, we can remove the other responses from the context, i.e., $ctx^b_{<i} = u_1 \oplus u_2 \oplus \cdots u_{i-1} \oplus \hat{s}_{i-1}$ (see Type-b in Figure 1). However, this may sacrifice some context information from the removing responses. To address this issue, we propose to replace the raw utterances in the context $ctx^b_{<i}$ with the reformulated ones (see Recursive in Figure 1):

$$ctx^{recur}_{<i} = \hat{q}_1 \oplus \hat{q}_2 \oplus \cdots \hat{q}_{i-1} \oplus \hat{s}_{i-1} \tag{4}$$

where $\hat{q}_i$ is the reformulated query at turn i in equation 2. Our insight is that at each turn i, query reformulation can bring the context information from response into $\hat{q}_i$; thus, ideally, $ctx^{recur}_{<i}$ maintains sufficient context information from both historical utterances and responses without the the concatenation of all historical system responses. Equation 4 shows that at each turn, the context $ctx^{recur}_{<i}$ for query reformulation depends on the output of previous turns. This is the reason why we call it recursive inference.

3 Experiments

Settings. In our experiments, we use T5-base and T5-large fine-tuned on the CANARD dataset and test their query reformulation (QR) performance under different inference settings: Query-only, Type-a, Type-b and Recursive. While inference, we use greedy search (beam size 1) for simplicity. We evaluate model performance in two perspectives: (1) Query reformulation performance: we compare models' reformulated queries with manual reformulated queries and quantify the performance using BLEU scores using manual annotated queries provided by CAsT 2020 as golden queries. (2) Downstream passage ranking: we feed the reformulated queries to our multi-stage pipeline for passage retrieval and test the overall (R@1000, MAP) and top-k (NDCG@3) ranking performance.

Results. Observing Table 2, T5-large shows better performance than T5-base in terms of BLEU and the downstream passage retrieval task. Among all, our proposed recursive inference using T5-large (condition 7) yields the best overall and top-k ranking performance, which is our best run submitted to CAsT 2020. Another observation is that ranking effectiveness seems to have positive correlation with reformulation metrics. It is worth noting that T5-base and T5-large show different trends among inference types. First, query-only inference yields better QR performance when using T5-base while inference with system response outperforms Query-only inference when using T5-large. This is possibly because in addition to context information, concatenating system response also introduces unrelated information and T5-base does not have sufficient capability to rewrite queries under the complex scenario. Finally, the comparison of ours and manual QR methods, large performance gap can be seen, indicating that there is still room for improvement.

Case study. Figure 2 compares the reformulated queries with the two inference types: Query-only, Recursive. Observing turns 2, 3 and 5, Recursive inference shows better ranking result than query-only inference since recursive method captures the context, Airbus A380, from system

Turn	Raw	Manual	T5-base Query-only	T5-large Query-only	T5-base Recursive	T5-large Recursive
1	which is the biggest commercial plane ?					
NDCG@3	-	1	1	1	1	1
2	What are its operational costs?	What are the operational costs of the **Airbus A380**?	what are the operational costs of the biggest commercial plane ?	which is the biggest commercial plane ?	what are the operational costs of the **Airbus A380** ?	
NDCG@3	-	0.5307	0	0	**0.5307**	**0.5307**
3	How does its fuel consumption compare to its competitors? compare to its competitors?	How does the **Airbus A380** fuel consumption compare to its competitors?	how does the biggest commercial plane 's fuel consumption compare to its competitors ?	how does Which is the biggest commercial plane 's fuel consumption compare to its	how does the fuel consumption of the **Airbus A380** compare to its competitors ?	
NDCG@3	-	0.0782	0	0	**0.0782**	**0.0782**
4	How do the freighter versions compare to each other?	How do the freighter versions of the **Airbus A380** and **Boeing 747** compare to each other?	how do the freighter versions of the biggest commercial plane compare to each other?	how do the freighter versions of which is the biggest commercial plane compare to each other ?	how do the freighter versions of the **airbus a380** compare to each other ?	
NDCG@3	-	0	0	**0.3612**	0	0
5	Why did the A380 stop being produced?	Why did the **Airbus A380** stop being produced?	Why did the A380 stop being produced?		why did the **airbus a380** stop being produced ?	
NDCG@3	-	0.4058	0.1353	0.1353	**0.4058**	**0.4058**
6	What was Boeing's response to compete with it?	What was Boeing's response to compete with the **Airbus A380**?	what was Boeing 's response to compete with the a380 ?		what was Boeing 's response to compete with Airbus ?	what was Boeing 's response to compete with the **Airbus A380** ?
NDCG@3	-	0.1246	**0.3605**	**0.3605**	0	0.1246

Figure 2: Case study (Session 90). Due to space limitation, we omit the last two turns (turns 7 and 8). For simplicity, we compare QR methods' ranking performance from our retrieval (dense+sparse) module.

response and keeps it in the reformulated queries. However, at turn 4, recursive inference fails to capture another context, Boeing 747, from response and reformulates queries incorrectly, which even get worse performance than the query-only counterpart (using T5-large). Furthermore, from turn 6, we observe that T5-large shows better QR capability than T5-base under the scenario of recursive inference. That is, recursive inference using T5-base loses the keywords, A380, and this downgrades its ranking performance.

Discussion. From our numerical results and case study, we demonstrate that recursive inference can capture the context from system response and that T5-base does not have sufficient capability under such scenario. However, we admit that it is challenging to quantify the measurement since we do not know exactly which user utterances refer to the context from system response. In addition to the ranking results, another interesting aspect is to compare model performance on the user utterances referring to the context from system responses and historical utterances separately.

4 Conclusion

In this notebook, we introduce our multi-stage conversational search pipeline, including query reformulation, passage retrieval and passage re-ranking modules. In addition, we highlight the main challenges of using sequence-to-sequence models for QR in canonical response entry and how we address this problem. Our experimental results show that our proposed method effectively captures the context from system response without concatenating the whole response (passages) into the input texts for QR.

References

[1] P. Bajaj, D. Campos, N. Craswell, L. Deng, J. Gao, X. Liu, R. Majumder, A. McNamara, B. Mitra, T. Nguyen, et al. MS MARCO: A human generated machine reading comprehension dataset. *arXiv:1611.09268*, 2016.

[2] A. Elgohary, D. Peskov, and J. Boyd-Graber. Can you unpack that? learning to rewrite questions-in-context. In *Proc. EMNLP*, pages 5917–5923, 2019.

[3] D. Jeffrey, C. Xiong, and J. Callan. CAsT 2019: The conversational assistance track overview. In *Proc. TREC*, 2019.

[4] J. Johnson, M. Douze, and H. Jégou. Billion-scale similarity search with GPUs. *arXiv:1702.08734*, 2017.

[5] O. Khattab and M. Zaharia. ColBERT: Efficient and effective passage search via contextualized late interaction over BERT. In *Proc. SIGIR*, page 39–48, 2020.

[6] S.-C. Lin, J.-H. Yang, and J. Lin. Distilling dense representations for ranking using tightly-coupled teachers. 2020.

[7] S.-C. Lin, J.-H. Yang, R. Nogueira, M.-F. Tsai, C.-J. Wang, and J. Lin. Query reformulation using query history for passage retrieval in conversational search. 2020.

[8] R. Nogueira, Z. Jiang, and J. Lin. Document ranking with a pretrained sequence-to-sequence model, 2020.

[9] 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*, 21:1–67, 2020.

[10] S. Vakulenko, S. Longpre, Z. Tu, and R. Anantha. Question rewriting for conversational question answering. *arXiv:2004.14652*, 2020.

[11] P. Yang, H. Fang, and J. Lin. Anserini: Reproducible ranking baselines using Lucene. *ACM J. Data. Inf. Qual.*, 10(4):Article 16, 2018.

[12] S. Yu, J. Liu, J. Yang, C. Xiong, P. Bennett, J. Gao, and Z. Liu. Few-shot generative conversational query rewriting. In *Proc. SIGIR*, pages 1933–1936, 2020.

[13] S. Zou, G. Tao, J. Wang, W. Zhang, and D. Zhang. On the equilibrium of query reformulation and document retrieval. In *Proc. SIGIR*, pages 43–50, 2018.