UNIVERSITY OF CAMBRIDGE AT TREC CAST 2022

Adian Liusie
Cambridge University
al826@cam.ac.uk

Mengjie Qian
Cambridge University
mq227@cam.ac.uk

Xiang Li
Cambridge University
xl514@cam.ac.uk

Mark Gales
Cambridge University
mjfg@cam.ac.uk

ABSTRACT

Team heatwave (of the University of Cambridge) submitted 3 automatic runs to the TREC 2022 Conversational Assistance Track. This paper discusses our approach to the challenge of conversational informational retrieval. We first describe our four stage approach of query reformulation, BM25 retrieval, passage reranking, and response extraction. Our experiments then show that our multi-query approach, which uses the raw concatenated conversational context for BM25 and the rewritten query for reranking, shows considerable performance improvement over a single-query approach, where our best performing system achieves a NDCG@3 of 0.440 in the 2022 CAsT challenge.

1 INTRODUCTION

The Conversational Assistance Track (CaST) is an annual challenge that encourages progress in information retrieval for conversations. The aim is to develop an automatic system that can retrieve passages relevant to a request in conversational form. Given a current query utterance, conversational history, and a large document archive, the automatic system must retrieve a set of useful passages that meets the information needs of the user. The 2022 challenge [1] differs to the 2021 challenge (Dalton et al., 2022) by assessing the quality of generated responses, having multiple conversations per topic and by using a larger document archive.

This paper focuses on our approach to the 2022 CAsT challenge. We investigated several different aspects of conversational retrieval including: 1) using different queries at retrieval and reranking 2) looking at the impact of k between retrieval and reranking and 3) investigating whether various systems are complimentary when combined. Our best performing system used the concatenated contextual history as queries for BM25 retrieval and rewritten queries for a DuoT5 reranker, and achieved a NDCG@20 of 0.389 and NDCG@3 of 0.440 on the 2022 challenge data.

2 METHODOLOGY

Our approach follows the three stage pipeline commonly used by previous successful teams (Lin et al., 2020) consisting of: (1) query rewriting, (2) document retrieval, and (3) reranking. This year, the output is expected to be a short response, and so a final generation/extraction head is also used to generate the returned response. This section briefly describes the purpose of each component and considered approaches.

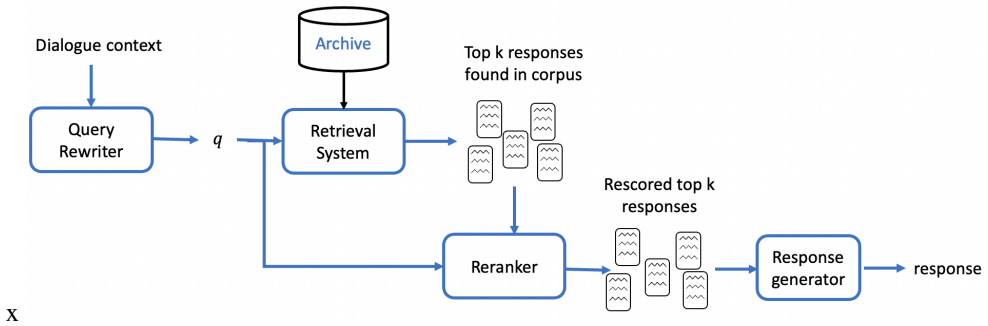


Figure 1: System Pipeline

[1] https://www.treccast.ai/

2.1 QUERY REWRITING

The objective of query rewriting is to create a query q_k that encapsulates all previous contextual information. That is, all ambiguities and anaphoras present in the current utterance u_k are resolved to create the query q_k, such that q_k contains all information required to identify the best next response of the system. Let ctx-n-m refer to concatenating the previous n user utterances and m system responses (as well as the current user utterance) in chronological order:

$$\text{ctx-2-2} \quad \Rightarrow \quad u_{k-2} \oplus r_{k-2} \oplus u_{k-1} \oplus r_{k-1} \oplus u_k \tag{1}$$

$$\text{ctx-2-1} \quad \Rightarrow \quad u_{k-2} \oplus u_{k-1} \oplus r_{k-1} \oplus u_k \tag{2}$$

We use T5 systems (Raffel et al., 2020) to generate a rewritten query q_k from ctx-4-2. We also look at using the raw conversational context ctx-5-3 and ctx-3-1 as a baseline for retrieval.

2.2 PASSAGE RETRIEVAL

The second stage retrieves a batch of passages that are relevant to the current query. For a given query, every passage in the archive is scored and the top-k passages are passed on in the pipeline. If $S_1(d, q)$ is a scoring function that measures the relevance of passage d for query q, and $\mathcal{R}$ is the entire document archive, then the returned corpus $\mathcal{R}_{top-k}$ is the top-k most relevant passages as predicted by the retrieval approach. Our

$$\mathcal{R}_{top-k} = \{r \mid S_1(r, q_i) > S_1(r_k, q_i) \quad \forall r_k \in \mathcal{R} \cap \bar{\mathcal{R}}_{top-k}\} \tag{3}$$

In this stage, millions of documents per query must be scored, and so the scoring function has to be very efficient. We therefore consider the popular approach of BM25 (Robertson et al., 2009) – an unsupervised lexical method. We initially also considered a sparse retrieval approach of ANCE (Xiong et al., 2020), however initial results were worse and this investigation was discontinued.

2.3 PASSAGE RE-RANKING

Re-ranking, as well, scores passages by their relevance to the current query. However, instead of scoring all the passages in the archive, only the top-k passages (returned by the previous retrieval stage) are rescored. Since the number of passages to score is massively smaller, a more powerful and accurate scoring function can be used.

We consider three deep learning systems: SentenceBERT (Reimers et al., 2019), MonoT5 (Nogueira et al., 2020) and DuoT5 (Pradeep et al., 2021). SentenceBERT encodes each query and response into vectors, where the relevance score is the inner product of the two. MonoT5 is a T5 based system, with the query and response are concatenated as the input and the decoder predicts the relevance score. DuoT5 consists of two reranking stages, where first MonoT5 is used to select the top n documents, then pairwise ranking is used to re-rank the top n documents.

We also investigate using different queries at retrieval and reranking. Instead of using the same query rewriting system to generate a single query used for both retrieval and reranking, we consider having different queries q_1 and q_2 for the different stages, as shown in Figure 2

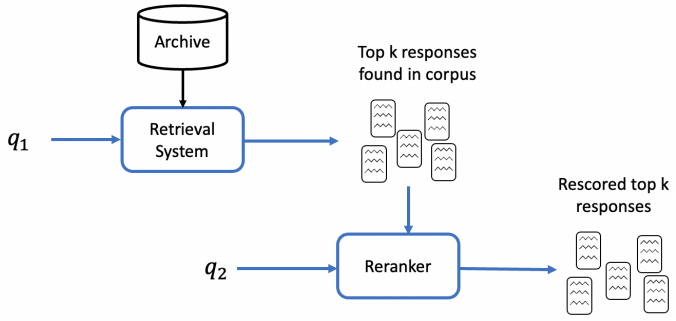


Figure 2: Set up for using different queries for retrieval and reranking

2.4 Response Generation

New to CAsT 2022, a response different to the raw passage can be returned. This can be a generated response, which may be based on a single or multiple retrieved passages. We use a very simple answer extraction approach (Clark et al., 2020) to generate our response. The context for the question answering system is the predicted relevant passage, and the query is used as the question. The answer span is then returned as the response.

3 Experimental Setup

3.1 CAsT datasets

Archive
The archive to retrieve from is a mix of highly informative content such as news and Wikipedia articles. The archive is made up of three corpora: MSMARCO, KILT and Washington Post. For 2021, there are 9 million documents present in the archive, while 2022 had 17 million documents. The difference in the 2021 and 2022 archive is that 2021 uses MSMARCO v1 while 2022 uses MSMARCO v2.

Development
Two CaST datasets were used for development: The final 2021 challenge data, **EVAL-21**, and the 2022 challenge data along with the canonical answers, **DEV-22**. EVAL-21 had a full list of relevant documents and relevance scores for each conversational query. To compare our systems with last year systems, we follow the same set up as in final evaluation, which used binary relevance scores thresholded at 2. For the 2022 challenge data with the canonical documents (which was available before submission), 1-3 relevant documents were provisionally marked for each query, which is much sparser than the relevance list used in final evaluation.

Although the three stage pipeline (of figure 1) was used for both the EVAL-21 and DEV-22, there were differences in the two set ups. For EVAL-21 BM25 was done at the document level, and the re-ranker then operated at the passage level, while DEV-22 split documents into passages before BM25. Initial development initially focused on performance on EVAL-21, while later development focused on DEV-22.

Evaluation
Final evaluation, done after submission, was done on the 2022 challenge data, EVAL-22. EVAL-22 had a full list of relevant passages and relevance scores for each query. EVAL-22 and DEV-22 use the same set of queries, however DEV-22 only had the canonical documents, and so only had sparse query labels.

3.2 Systems Configuration

Query Rewriting
The query rewriter is trained on one of two different query reformulation datasets, either canard (Elgohary et al., 2019) or qrecc (Anantha et al., 2020). We investigated fine-tuning both T5-base (Raffel et al., 2020) and T5-large (Raffel et al., 2020) systems, where the T5-large systems were used since they showed better downstream performance. As a baseline, we also consider using ctx-5-3 and ctx-3-1 directly as the 'rewritten query'.

Retrieval
We use the beir framework (Thakur et al., 2021) implementation of BM25 for retrieval, and use the default parameters of ($k = 0.9$ and $b = 0.4$).

Reranker
MSMARCO (Nguyen et al., 2016) was used as the retrieval training data for all the different rerankers. Our rerankers were downloaded from the huggingface hub (SBERT [2], monot5 [3], duoT5 [4]), where the rerankers were all trained at the passage level.

Reponse Generator
Reponses were generated using a question answer extraction system, trained on (Rajpurkar et al., 2016). The

[2] https://huggingface.co/sentence-transformers/msmarco-distilbert-dot-v5
[3] https://huggingface.co/castorini/monot5-base-msmarco
[4] https://huggingface.co/castorini/duot5-3b-msmarco

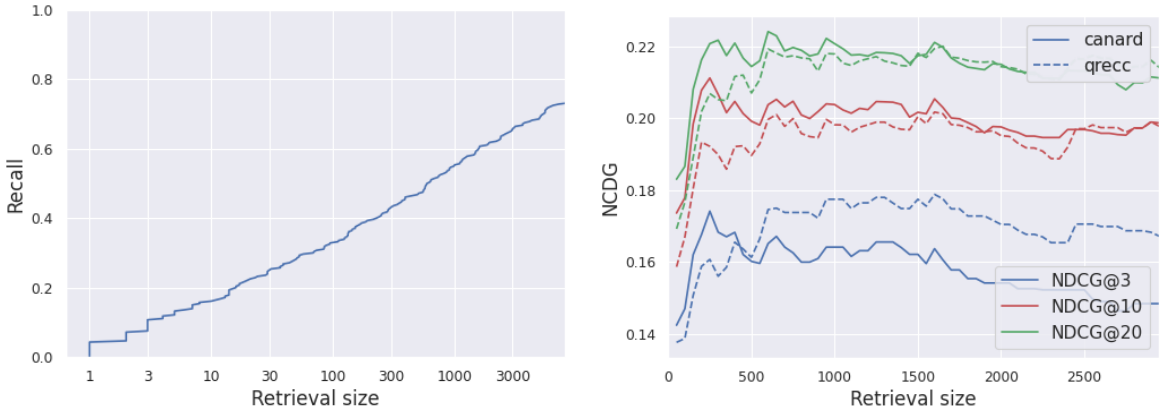
query was then used as the question, the most relevant document as the context, and returned the extracted answer span was the response.

4 Experimental Results

Query	EVAL-21 R@1k	EVAL-21 R@500	EVAL-21 NDCG@3	DEV-22 R@1k	DEV-22 R@500	DEV-22 NDCG@3
T5-canard	0.672	0.596	0.324	0.458	0.387	0.068
T5-qrecc	-	-	-	0.460	0.397	0.060
ctx_3_1	0.709	0.637	0.296	0.572	0.488	**0.135**
ctx_5_3	**0.734**	**0.648**	0.248	**0.583**	**0.495**	0.094
automatic	0.650	0.581	**0.326**	0.485	0.419	0.072

Table 1: Retrieval results when using BM25 for different queries

Best retrieval query: We first investigate the rewriting system for retrieval. The query rewriter generates a query, which is used by BM25 to return the highest scoring 1000 passages. The objective after retrieval is to preserve as many relevant documents in the top-1000 as possible, irrespective of ranking since the next stage will re-score all top 1000 documents. Therefore emphasis is on recall@1000 (R@1k). Table 1 shows retrieval performance for the different query rewriting systems, where the baseline of raw concatenated conversational context clearly showed best performance for both EVAL-21 and DEV-22.



(a) recall at different retrieval sizes, k

(b) Performance sweep for different k

Figure 3: Retrieval Top-k results on DEV-22

Retrieval top-k: We next look into whether returning the top 1000 passages at retrieval is a sensible decision. If more passages are returned at retrieval, the number of relevant passages that the re-ranker scores will be larger, which may benefit down stream performance. Figure 3a shows how recall scales roughly with O(log(k)). Figure 3b shows the downstream MonoT5 performance on DEV-22, using ctx-5-3 queries at retrieval and T5 rewritten queries for the re-ranker, and sweeping over k. 1000 therefore seemed to be a sensible choice for the size of retrieval list, with performance not improving for larger k. We therefore stick with top-1000 documents at retrieval for all future results.

DuoT5 top k	NDCG@20	NDCG@10	NDCG@3
20	0.199	0.220	0.162
30	0.231	0.222	0.169
50	0.230	0.218	0.170

Table 2: Reranking results when varying the DuoT5 top k on DEV-22

DuoT5 top-k: DuoT5 does pairwise comparisons between passages, and therefore scales with $O(N^2)$ instead of the $O(N)$ of MonoT5. One has to therefore use much shorter top-k lists for DuoT5. DuoT5 is applied after MonoT5 scoring, where Table 2 shows reranking performance for the top 20, 30 or 50 passages. Using top-20 was significantly worse than using the top-30 or top-50, while the top-30 and top-50 were comparable with top-30 being marginally better. As top-50 is also considerably more computationally expensive, we therefore use the top-30 in all DuoT5 systems considered.

			EVAL-21			DEV-22		
query 1	query 2	reranker	MAP	NDCG@20	NDCG@3	MAP	NDCG@20	NDCG@3
↑	↑	-	0.168	0.261	0.324	0.076	0.098	0.068
T5-canard	T5-canard	MonoT5	0.288	0.407	0.450	0.152	0.190	0.140
↓	↓	sbert	0.204	0.309	0.349	-	-	-
↑	↑	-	0.165	0.248	0.248	0.094	0.130	0.094
ctx-5-3	T5-canard	MonoT5	0.310	0.430	**0.489**	0.175	0.221	0.164
↓	↓	+ DuoT5	-	-	-	**0.184**	**0.231**	0.169
↑	↑	-	-	-	-	0.094	0.130	0.094
ctx-5-3	T5-qrecc	MonoT5	-	-	-	0.179	0.218	**0.177**
↓	↓	+ DuoT5	-	-	-	0.166	0.211	0.161

Table 3: Reranking results

Reranker results: Table 3 shows the performance of the different rerankers for different retrieval outputs. The concatenated contextual query seems to be optimal for the retrieval, with notable performance improvement over using only a single rewritten query. At the reranker, both the T5-canard and T5-qrecc rewriting systems performed similarly for DEV-22, as did the monoT5 and duoT5 systems. Canard + duoT5 had the best all round performance, while qrecc + monoT5 had the best NDCG@3 performance, and hence these two were submitted.

			NDCG@20		NDCG@3		MAP	
query 1	query 2	reranker	DEV-22	EVAL-22	DEV-22	EVAL-22	DEV-22	EVAL-22
ctx-5-3	T5-canard	monoT5	0.221	0.236	0.164	0.274	0.175	0.136
		+ duoT5	0.231	**0.389**	0.169	**0.440**	0.184	**0.200**
ctx-5-3	T5-qrecc	monoT5	0.218	0.261	0.177	0.320	0.179	0.139
		+ duoT5	0.211	0.364	0.161	0.423	0.166	0.189

Table 4: Reranking results for DEV-22 and EVAL-22.

Reranker Results on EVAL-22: Although pre-submission analysis suggested that both monoT5 and duoT5 had comparable performance, on EVAL-22 (Table 4) DuoT5 performed significantly better. The only difference between DEV-22 and EVAL-22 is that EVAL-22 had very sparse results, and it is hypothesized that this sparsity meant that the passages marked relevant in DEV-22 were passages with high relevance. Mono-T5 may therefore be able to identify clearly relevant passages, however DuoT5 may have a better understanding when it comes to documents which are relevant but of lower score.

	query 1	query 2	reranker	NDCG@20		NDCG@3		MAP	
				DEV-22	EVAL-22	DEV-22	EVAL-22	DEV-22	EVAL-22
run1	ctx-5-3	T5-qrecc	MonoT5	0.218	0.261	0.177	0.320	0.179	0.139
run2	ctx-5-3	T5-canard	+ DuoT5	0.231	0.389	0.169	0.440	0.184	0.200
run3	ctx-5-3	-	combine	0.228	0.384	0.167	0.439	0.183	0.197
run4	gold	gold	MonoT5	0.306	0.365	0.238	0.405	0.246	0.217

Table 5: submitted systems final performance

Final Performance of Submitted Systems: Table 5 shows the performance of our submitted systems. We additionally submitted a combined system which averaged the normalised scores of the MonoT5 and DuoT5 systems, even though analysis done on DEV22 did not show clear benefit in system combination (which we looked at multiple different levels). Overall we found that DuoT5 which used multiple queries was our best performing system, and achieved a MAP of 0.200, NDCG@20 of 0.389 and NDCG@3 of 0.440.

5 Conclusions

In this notebook, we follow a four stage pipeline that returns a relevant response for conversational queries. Our experimental results show that using different queries at retrieval and re-ranking can be largely advantageous, and that the concatenated conversational context serves as a good query for BM25 retrieval. We further find that DuoT5 outperforms monoT5 in the 2022 CAsT challenge data. Future work could investigate this performance difference, and to consider efficient modifications on duoT5 to instead operate with $O(N)$ and therefore rescore more documents.

Acknowledgements

This paper reports on research partially supported by Cambridge University Press & Assessment, a department of The Chancellor, Masters, and Scholars of the University of Cambridge. It is also partly supported by EPSRC Project EP/V006223/1 (Multimodal Video Search by Examples).

References

Raviteja Anantha, Svitlana Vakulenko, Zhucheng Tu, Shayne Longpre, Stephen Pulman, and Srinivas Chappidi. Open-domain question answering goes conversational via question rewriting. *arXiv preprint arXiv:2010.04898*, 2020.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. Electra: Pre-training text encoders as discriminators rather than generators. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=r1xMH1BtvB.

Jeffrey Dalton, Chenyan Xiong, and Jamie Callan. TREC CAsT 2021: The Conversational Assistance Track Overview. In *TREC*, 2022.

Ahmed Elgohary, Denis Peskov, and Jordan Boyd-Graber. Can You Unpack That? Learning to Rewrite Questions-in-Context. In *Empirical Methods in Natural Language Processing*, 2019. URL http://umiacs.umd.edu/~jbg//docs/2019_emnlp_sequentialqa.pdf.

Sheng-Chieh Lin, Jheng-Hong Yang, and Jimmy Lin. Trec 2020 notebook: Cast track. In *TREC*, 2020.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. Ms marco: A human generated machine reading comprehension dataset. In *CoCo@ NIPs*, 2016.

Rodrigo Nogueira, Zhiying Jiang, Ronak Pradeep, and Jimmy Lin. Document Ranking with a Pretrained Sequence-to-Sequence Model. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 708–718, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.63. URL https://aclanthology.org/2020.findings-emnlp.63.

Ronak Pradeep, Rodrigo Nogueira, and Jimmy Lin. The expando-mono-duo design pattern for text ranking with pretrained sequence-to-sequence models. *arXiv preprint arXiv:2101.05667*, 2021.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020. URL http://jmlr.org/papers/v21/20-074.html.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 2383–2392, Austin, Texas, November 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1264. URL https://aclanthology.org/D16-1264.

Nils Reimers, Iryna Gurevych, Nils Reimers, Iryna Gurevych, Nandan Thakur, Nils Reimers, Johannes Daxenberger, Iryna Gurevych, Nils Reimers, Iryna Gurevych, et al. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, pp. 671–688. Association for Computational Linguistics, 2019.

Stephen Robertson, Hugo Zaragoza, et al. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389, 2009.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. Beir: A heterogeneous benchmark for zero-shot evaluation of information retrieval models. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul N Bennett, Junaid Ahmed, and Arnold Overwijk. Approximate Nearest Neighbor Negative Contrastive Learning for Dense Text Retrieval. In *International Conference on Learning Representations*, 2020.