IELAB for TREC Conversational Assistance Track (CAsT) 2020

Sebastian, Cross[1], Hang Li[1], Arvin Zhuang[1], Ahmed, Mourad[1], Bevan, Koopman[2], and Guido, Zuccon[1]

[1]The University of Queensland, Brisbane, QLD, Australia
[2]CSIRO, Brisbane, QLD, Australia

November 11, 2020

Abstract

This paper describes the work done by the IELAB for the TREC Conversational Assistance Track (CAsT) 2020. IELAB investigated two methods to improve both the retrieval and re-ranking stages of a conversational IR system. The first method used an Adapted Query (AQ), which extracted context from the first utterance only to expand all subsequent queries for a conversational session. The second method utilized a query likelihood model (QLM) which used a pre-trained deep language model to estimate the likelihood that a query could be generated by a document. Specifically, we used the text-to-text transfer transformer (T5) as a scoring functions for re-ranking passages.

1 Introduction

The aim of TREC Conversational Assistance Track (CAsT) is to advance research on conversational search systems. For TREC CAsT of 2020, the aim is to focus on candidate information ranking in the context of the conversation. This introduces two challenges: the first is extracting relevant context or information from the conversation. This is done to improve the effectiveness of the current query; the second is a traditional information retrieval task, which is retrieving and ranking documents that are relevant to the given query.

The IELAB team participated in TREC CAsT 2020 by submitting two systems that addressed these two challenges. The first system used an Adapted Query (AQ) approach to expand the query based on the contextual history of the conversation. Previous submissions in TREC CAst 2019 [1] included the entire contextual history of a conversation. However, our approach assumes that the main context of a conversation is stated during the first utterance. Table 1 shows an example conversation from the TREC CAst 2020 dataset. The focus of this conversation is the "garage door opener" which is stated within the first utterance. This term is never explicitly mentioned through the rest of the conversation although being explicitly referred to in each subsequent utterance. This leaves many questions open with no context or focus.

The second system used a query likelihood model for re-ranking the results. Previous success with the T5 model within the context of natural language processing was the leading motivation for using T5 in one of our systems.

Turn	Raw utterance	Manually rewritten utterance
1	How do you know when your garage door opener is going bad?	How do you know when your garage door opener is going bad?
2	Now it stopped working. Why?	Now my garage door opener stopped working. Why?
3	How much does it cost for someone to fix it?	How much does it cost for someone to repair a garage door opener?
4	How about replacing it instead?	How much does it cost to replace a garage door opener?
5	How do I choose a new one?	How do I choose a new garage door opener?
6	What does a smart one do?	What does a smart garage door opener do?
7	What's important for me to know about their safety?	What's important for me to know about the safety of smart garage door openers?
8	How could they be hacked?	How could smart garage door openers be hacked?

Table 1: Conversation example of raw and manually rewritten utterances from the 2020 data set

2 Methodology

In TREC CAst 2019, most of the submissions considered the entire context of a conversation and deployed off-the-shelf coreference models [1]. However, the large gap in performance between the raw utterance and manually resolved utterance systems demonstrated the struggle of the coreference models with TREC CAst topics. Our AQ approach, therefore, aims to partially resolve this issue by considering the first utterance in the conversation. Considering re-ranking, we build atop of TREC CAst 2019 submissions by using neural models to improve the re-ranking. However, we employed T5 instead of BERT. While other methods were investigated, the IELAB team submitted two systems based on the AQ and T5-QLM methods to the CAsT task for 2020.

2.1 Data Collections

TREC provided a set of 25 conversational scenarios. Each of these contained a set of conversational search utterances, which were provided in three forms: raw utterances, automatically rewritten utterances and manually rewritten utterances. The AQ system used the raw utterances, while the T5-QLM system used the manually rewritten utterances.

Indexing. We considered the two data collections provided by TREC CAsT, namely, MS-MARCO [3] and TREC CAR. They were both converted to JSON documents with the following format "id": "", "contents": "". Id represented the id of the document, and which collection it originated from. The contents represented the main text from each of the documents. Then we created two indexes: MS-MARCO and TREC CAR, which is used by AQ; and MS-MARCO only, which is used by T5-QLM. The file size of the MS-MARCO & TREC CAR index is 25GB, while the file size of MS-MARCO index is 5GB. The data collections were indexed using Apache Lucene [1]

2.2 Conversational Search Systems

The IELAB team submitted two systems. All submissions employed Pyserini[2] (a python wrapper for Anserini [5]), with BM25 and RM3, as a first stage retrieval model. It is also considered as a baseline. Pyserini was independently fine tuned to suit the task.

AQ System. AQ seeks to improve the BM25+RM3 baseline by using query expansion. It involves selecting the first query (Q1) from the conversation, and using this query to extend every subsequent query (Qn). The adaptation is achieved by appending Q1 to Qn. The new extended query is then run through the fine tuned BM25+RM3 index to retrieve the candidate responses.

T5-QLM System The QLM-T5, uses the text-to-text transfer transformer language model (T5)[4] instead of the maximum likelihood estimation in a query likelihood model (QLM). A QLM is used to calculate the probability $P(Q|D)$ of generating a query Q based on a document D. The T5 model uses an encoder-decoder architecture. The encoder's input is tokens from the document $d_0, d_1 ... d_n \in D$. The decoder's input is tokens from the target query $q_0, q_1 ... q_{|Q|} \in Q$ along with a pad token <PAD>. This pad token is used to signify the start sequence for the decoder. For each time step t, the decoder generates the probability of using the next token from the target query $P_{T5}(q_{t+1})$. We exploit this QLM-T5 to re-rank documents, by estimating the likelihood that a query could be generated by a document. Specifically, we compute the query (log) likelihood for Q given the document D as

$$log(P_{QLM-T5}(Q, D)) = log(P_{T5}(< PAD >)) + \sum_{i=0}^{|Q|-1} log(P_{T5}(q_i)) \quad (1)$$

Other Explorations We developed two other systems that utilized BERT translation models. One system to reproduce previous models that used BERT[2] for re-ranking. The other system utilized BERT for query expansion in a pseudo relevance feedback manner. Results for both systems were not reported because of a number of errors within the development process.

3 Results and Analysis

Each system was developed using the 2019 CAsT dataset, and evaluated using the 2019 CAsT Qrels. The 2019 CAsT task used three evaluation metrics, nDCG@3, MRR and MAP. Results shown in Table 2

[1] https://lucene.apache.org/

[2] https://github.com/castorini/pyserini

Run Methods	NDCG@3	MRR	MAP
BM25+RM3	.1716	.3495	.1469
BM25+RM3+AQ	.2614	.4935	.1925

Table 2: Experimental Run Results from 2019 Topics Dataset

Run Methods	NDCG@3	nDCG@5	nDCG@1000	MAP@1000
Raw Median	0.2795	0.2735	0.3749	0.1801
BM25_AQ	.0.1330	0.1282	0.2463	0.1024

Table 3: Raw Utterance Median result and AQ results

Run Methods	NDCG@3	nDCG@5	nDCG@1000	MAP@1000
Manual Median	0.4137	0.3977	0.4886	0.2633
BM25T5_QLM	.0.4103	0.4113	0.4472	0.2712

Table 4: Manual Utterance Median result and T5_QLM results

demonstrated that AQ substantially outperformed the baseline of BM25+RM3 over all metrics.

The results of our submitted runs closely resemble the expectations of our test runs. As seen in table 3 the AQ method results were nDCG@1000 = 0.2463 and the MAP@1000 = 0.1024. While our system under performed compared to the median raw utterance score, it should be noted that this system was not judge. The T5_QLM method will not be compared with the AQ method as this was performed over the manual utterance. The results for this method were nDCG@1000 = 0.4103 and the MAP@1000 = 0.2712.

4 Conclusion

The IELAB submission for TREC Conversational Assistance (CAsT) Track 2020 focused on two submissions the AQ and QLM_T5. These two submission were chosen as they attempt to solve the two main challenges outlined in the TREC CAsT task. Future work will be focused on improving the retrieval results by extending the current submitted runs. For instance, investigation into improving the method of context extraction has begun, specifically extracting context from the syntactical structure of the conversation.

References

[1] Jeffrey Dalton, Chenyan Xiong, and Jamie Callan. Cast 2019: The conversational assistance track overview. In *Proceedings of the Twenty-Eighth Text REtrieval Conference, TREC*, pages 13–15, 2019.

[2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[3] Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. Ms marco: A human-generated machine reading comprehension dataset. 2016.

[4] 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. *arXiv preprint arXiv:1910.10683*, 2019.

[5] Peilin Yang, Hui Fang, and Jimmy Lin. Anserini: Reproducible ranking baselines using lucene. *Journal of Data and Information Quality (JDIQ)*, 10(4):1–20, 2018.