Abstract

In this work, we investigate approaches for query recontextualization in the context of conversational search. We use a pipeline setting in which we first reformulate the query and then rank passages according to a backbone model. Our main focus is put on the feature inputs of a T5 query reformulation model and we evaluate different evidence sources such as the history (previous questions and answers) as well as semantic proxy through the doc2query model. We also experiment an end-to-end version of the setting which unfortunately has not been much optimized due to time constraints.

1 Introduction

A key component in conversational search is the construction of contextualized queries, which should incorporate relevant information from previous utterances in the context in order to retrieve documents. Different approaches can be used, ranging from query reformulation before document ranking or directly ranking documents using the conversation context in an end-to-end fashion.

In this work, we aim to test approaches to collect relevant pieces of information from previous utterances through a recontextualization (RC) sequence-to-sequence model. Namely, we use the T5 model \cite{2021} to reformulate queries. From a given query and a conversation context, the trained RC model produces a new query that must contain self-sufficient information allowing a ranking model
(BM25+MonoT5) to retrieve relevant documents. The main objective of the work is to understand what types of input features are necessary for the RC model to perform well. To this end, we only work on features based on either previous queries or the associated provided answers (namely documents). Particularly, our features are based on the raw data (either documents or queries), reformulated raw queries, or queries generated from relevant documents by using the doc2query model [5]. We also develop an end-to-end reformulation/ranking approach, hoping that the outcome can be used to optimize the entire process. However, due to the late development of the approach, we were unable to efficiently train the model and index it on the whole released document collection. We have thus formulated relevance proxy for training data that did not allow us to optimize this last setting at its best. All approaches described here have been submitted to the TREC CAsT 2021 challenge.

In the following sections, we will first briefly present notations. Then, we will describe the different approaches submitted to TREC CAsT 2021, distinguishing pipeline and end-to-end versions. Finally, we present and discuss the results on TREC CAsT 2021 challenge and conclude the paper.

2 Task definition and notation

Let’s consider a conversation \( C \) composed of successive utterances \( i \), alternating questions \( q_i \) and passages returned as answers \( p_i \). Its history is viewed as a sequence of queries/passages:

\[
C = \{q_1, p_1, q_2, p_2, \ldots, q_i, p_i, \ldots\}
\]  

(1)

The objective of the the TREC-CAST 2021 challenge is to retrieve, for each query \( q_i \) in the conversation, the relevant passages \( p_i \) given the previous utterances. To do this task, we develop and submit different types of models: the first group of models relying on a reformulation and ranking pipeline, the second using an end-to-end version for reformulating and ranking conversational queries.

3 Reformulation pipeline

This pipeline relies on learning two main modules as presented in Figure [1]. The first one, reformulating the current query given the context using the pre-trained T5 model[6]. The T5 model is a sequence-to-sequence model encoding and decoding text using transformers. This model has shown great ability for generating text sequences and has already been used for reformulated queries [3]. The second performs re-ranking on the contextualised query using the pre-trained MonoT5 model [4].

---

3.1 Corpora and Training data

For fine-tuning the T5 Reformulation model, we considered two datasets:

• The CANARD corpus\(^2\) is a conversational dataset with reformulated queries. Similarly to the TREC CAS\(\ddot{\mathrm{T}}\) test set, it is possible to obtain an history for a query \(q\) as well as its reformulated version. The training, development, and test sets includes 31,538, 3,418, and 5,571 contextual and reformulated queries respectively.

• CASTuR [1] also includes conversational data but highlights for each query utterance which previous questions could help to express the current information need. For each query, we have manually rewritten the query (by using the context). We thus obtain a new dataset composed of 752 contextual and reformulated queries (by considering only the helpful context). This dataset has been used to augment the CANARD training set.

3.2 Submitted models

We focus on the reformulation module and use a fixed ranking module for all submitted runs. We propose three different models for reformulation, which differ by the type and the data provided to T5 during the inference phase.

\texttt{t5\_monot5:} Given conversation \(C\), this model takes into consideration the previous queries and passages to reformulate query \(q\). The underlying intuition of this model is to evaluate the potential of leveraging the whole conversation for reformulating queries. The input to T5 is the following sequence for query utterance \(q\):

\[
q_1||p_1||q_2||p_2|| \cdots ||q_{i-1}||p_{i-1}||q_i
\]

where \(q_1, \ldots, q_{i-1}\) are previous queries and \(p_1, \ldots, p_{i-1}\) are the passages returned to the user in conversation \(C\).

\texttt{Rewritt5\_monot5:} Similarly to the previous model, the history is given as input to the model, but the raw queries are replaced by the reformulated queries \(q'_1, \ldots, q'_{i-1}\) contained in the ground truth of our training dataset. Our objective is to evaluate the potential of online reformulation, this model being the oracle since it uses the ground truth. The input fed to the T5 model is thus:

\[
q'_1||p_1||q'_2||p_2|| \cdots ||q'_{i-1}||p_{i-1}||q_i
\]

\(^2\)https://sites.google.com/view/qanta/projects/canard
where \( q'_j \) is the reformulated \( j^{th} \) query in the ground truth.

**t5_doc2query:** This model is based on the history of the conversation queries, supplemented by queries generated by a doc2query model (noted \( d2q() \)) on previous relevant passages (\( p_i \)). The doc2query model \( \text{[5]} \) generates queries from a given document which are used to augment the document content for collection indexing. Assuming that passages are longer than queries and might make noise in the semantic understanding, we propose here to replace each passage in the conversation by the top query generated by the doc2query model. Our intuition is that it might enhance the intent context with additional queries. The input to \( T5 \) is a text with a special token delimiting utterance or type of features. For query \( q_i \), the associated input sequence is:

\[
q'_1 || d2q(p_1)|| q'_2 || d2q(p_2) || \ldots || q'_{i-1} || d2q(p_{i-1}) || q_i
\]

where \( d2q(p_j) \) is the first query generated by the doc2query model for paragraph \( p_j \) returned in conversation \( C \) to the user in response to query \( q_j \).

### 4 End-to-end reformulation-based ranking

#### 4.1 Corpus and Training data

We propose here a model learning reformulation and ranking modules in an end-to-end fashion. To learn such a model, one way would be to only rely on a ground truth of relevant passages but we believe that the signal within the back-propagation would be weak for guiding the reformulation module. We thus built a dataset with two types of ground truth: relevant passages and query reformulation pairs. To do so, we rely on two datasets: Canard corpus for reformulation MSMarco for ranking. For each CANARD reformulated query (see Section 3.1), we retrieve passages from MSMarco using a re-ranking pipeline based on BM25\(^3\) and MonoT5\(^4\). We then select the 200 most relevant passages (according to the BM25+MonoT5 re-ranker) for each query. The first 4 passages are assumed to be relevant and the last 4 passages as irrelevant. The final training set contains 123,836 negative/positive pairs (30,959 queries) and validation set, 1152 samples. Due to time constraint, we do not index the corpus, but only test on a re-ranking setting based on the monoT5 baseline.

#### 4.2 Submitted model

Our end-to-end approach is called t5colbert and aims to reformulate and rank by branching two architectures, as illustrated in Figure\( ^2 \).

- For the reformulation, we consider the input of the t5_monot5 model presented in Section 3.2 and the model architecture is the T5 text-to-text encoder-decoder.

\(^3\)Based on https://github.com/castorini/pyserini
\(^4\)https://huggingface.co/castorini/monot5-large-msmarco
For the ranking model, we use the Colbert model [2]. As T5 and Colbert models have different output/input dimension, we design a feed-forward Neural network between them. The network is then trained in a supervised fashion in different steps. We thus have a cross-entropy loss $L_{ref}$ for reformulation and a cross-entropy loss $L_{rank}$ for ranking. The final loss is the sum of both losses.

5 Results

Table 1 reports the results on the test set submitted to TREC CAsT 2021 using the different approaches. Samples of reformulated queries are presented in Table 2.

By first looking at the pipeline models, we can see that the best model is the t5_monot5, which leverages the full conversation with raw queries and raw passages, as provided in the dataset. In contrast, other pipeline models obtain lower results. This suggests that reformulating queries as in t5_doc2query and in Rewritt5_monot5 might induce noise in the conversation. This is surprising since reformulated queries are the ones used in the ground truth. One reason

Table 1: Submitted TREC CAsT 2021 Results ($\times$100)

<table>
<thead>
<tr>
<th>Model</th>
<th>NDCG@3</th>
<th>NDCG@5</th>
<th>NDCG@500</th>
<th>MAP@500</th>
</tr>
</thead>
<tbody>
<tr>
<td>t5_monot5</td>
<td>38.7</td>
<td>39.0</td>
<td>33.6</td>
<td>19.5</td>
</tr>
<tr>
<td>Rewritt5_monot5</td>
<td>36.9</td>
<td>37.2</td>
<td>33.1</td>
<td>18.9</td>
</tr>
<tr>
<td>t5_doc2query</td>
<td>37.7</td>
<td>37.9</td>
<td>33.5</td>
<td>19.7</td>
</tr>
<tr>
<td>E2E (t5colbert)</td>
<td>15.3</td>
<td>15.8</td>
<td>31.4</td>
<td>10.1</td>
</tr>
</tbody>
</table>
Table 2: Examples of reformulations given the original query.

<table>
<thead>
<tr>
<th>Model</th>
<th>Reformulation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Query</td>
<td>I live in a remote area. How can I treat dog swollen ear at home?</td>
</tr>
<tr>
<td>t5_monot5</td>
<td>How can I treat dog swollen ear at home?</td>
</tr>
<tr>
<td>Rewritten5_monot5</td>
<td>How can I treat dog ear hematoma at home?</td>
</tr>
<tr>
<td>t5_doc2query</td>
<td>How can I treat hematoma in dog?</td>
</tr>
<tr>
<td>E2E</td>
<td>I live in a remote area. How can I treat dog swollen ear?</td>
</tr>
<tr>
<td>Query</td>
<td>How can I protect them</td>
</tr>
<tr>
<td>t5_monot5</td>
<td>How can I protect my dog from the coronavirus?</td>
</tr>
<tr>
<td>Rewritten5_monot5</td>
<td>How can I protect Jasmine from the coronavirus?</td>
</tr>
<tr>
<td>t5_doc2query</td>
<td>How can I protect dogs from coronavirus?</td>
</tr>
<tr>
<td>E2E</td>
<td>How can I protect my dog from the coronavirus?</td>
</tr>
<tr>
<td>Query</td>
<td>That’s great. What makes it so efficient?</td>
</tr>
<tr>
<td>t5_monot5</td>
<td>What makes a heat pump so efficient?</td>
</tr>
<tr>
<td>Rewritten5_monot5</td>
<td>What makes a heat pump so efficient?</td>
</tr>
<tr>
<td>t5_doc2query</td>
<td>That’s great. What makes a geothermal heat pump so efficient?</td>
</tr>
<tr>
<td>E2E</td>
<td>What makes geothermal heat pumps so efficient?</td>
</tr>
</tbody>
</table>

might be the redundancy of information between the conversation history and the reformulated queries, which might overload the search intent. Moreover, the t5_doc2query is slightly better than the Rewritten5_monot5. The difference of these models lies in the consideration of relevant passages: raw passages for Rewritten5_monot5 and the generated query issued from the doc2query model for t5_doc2query. This result is surprising since the T5 model has been learned on the full conversation history (sequence of queries and passages) and the t5_doc2query use features of a different distribution at the inferences (sequence of queries). This suggests that although based on a different distribution, passages might be somehow noisy for formulating queries. Combined with the previous statement on query reformulation and the results obtained in Table 1, it seems that replacing passages with queries is not sufficient for balancing the overload of online query reformulation. One additional baseline for future work will be to evaluate the model with raw queries and queries issued from relevant passages (combination of t5_monot5 and t5-doc2query models).

Concerning the end-to-end t5colbert model, it drastically fails on the test set. This can be due to different reasons such as the complexity of training (finding the best hyperparameters) or designing a better training dataset. As explained earlier, we lack time for indexing the whole collection and performing effective tuning.

6 Conclusion

In this report, we describe our work in the TREC CAsT challenge. We proposed different approaches based on feature selection or generation for the conversational search task. To this end, we adopted the reformulation and ranking pipeline which first re-contextualizes the query and then retrieves relevant passages from a collection. We tested different types of input such as generated queries (with a doc2query model), previous relevant documents (passage), previous queries, or previous reformulated queries.
7 Acknowledgements

We would like to thank ANR for supporting this work under the grant ANR JCJC SESAMS (ANR-18- CE23-0001).

References


