Making Information Seeking Easier: An Improved Pipeline for Conversational Search

Vaibhav Kumar *
Language Technologies Institute
Carnegie Mellon University
vaibhav2@cs.cmu.edu

Jamie Callan
Language Technologies Institute
Carnegie Mellon University
callan@cs.cmu.edu

Abstract

This paper presents a highly effective pipeline for passage retrieval in a conversational search setting. The pipeline comprises of two components: Conversational Term Selection (CTS) and Multi-View Reranking (MVR). CTS is responsible for performing the first-stage of passage retrieval. Given an input question, it uses a BERT-based classifier (trained with weak supervision) to de-contextualize the input by selecting relevant terms from the dialog history. Using the question and the selected terms, it issues a query to a search engine to perform the first-stage of passage retrieval. On the other hand, MVR is responsible for contextualized passage reranking. It first constructs multiple views of the information need embedded within an input question. The views are based on the dialog history and the top documents obtained in the first-stage of retrieval. It then uses each view to rerank passages using BERT (fine-tuned for passage ranking). Finally, MVR performs a fusion over the rankings produced by the individual views. Experiments show that the above combination improves first-state retrieval as well as the overall accuracy in a reranking pipeline. On the key metric of NDCG@3, the proposed combination achieves a relative performance improvement of 14.8% over the state-of-the-art baseline and is also able to surpass the Oracle.

1 Introduction

The abilities of current conversational assistants (Alexa, Cortana etc.) to perform open-domain *conversational information seeking* (CIS) functions are limited (Dalton et al., 2019). Thus, to encourage and support research on conversational information seeking, the TREC Conversational Assistance Track (CAsT) (Dalton et al., 2019) defined a model of conversational information seeking in which the conversation is a sequence of related passage ranking tasks, some of which require knowing the conversational history.

Title: goat breeds
Description: Interested in buying goats that implies interest in different breeds of goats and their use (milk, meat, and fur).

Turn	Utterance (Question)
1	What are the main breeds of goat?
2	Tell me about boer goats.
3	What breed is good for meat?
4	Are angora goats good for it?
5	What about boer goats?
6	What are pygmies used for?
7	What is the best for fiber production?
8	How long do Angora goats live?
9	Can you milk them?
10	How many can you have per acre?
11	Are they profitable?

Table 1: An example of a training topic in CAsT.

For example, the question "Can you milk them?" in Table 1 is not by itself sufficient to support effective retrieval; the conversational context is required. More formally, given a series of natural language utterances/questions $U = \{u_1, u_2, u_3 \dots u_n\}$ based on a conversational topic T, the task is to retrieve relevant passages P_i for each utterance u_i by conditioning on the utterances/questions occurring prior to it, i.e $\{u_1, u_2, \dots u_{i-1}\}$. Note that, each utterance in the conversation is essentially a question by itself.

CAsT questions pose a variety of problems for a conversational information seeking system. To begin with, the evolution of the conversation is accompanied by introduction of pronouns, which creates an under-specified (or missing) context within the posed questions. Depending on the question, the context markers might be explicit (pronouns) or

* The author is now an Applied Scientist at Amazon Alexa AI. Alternatively, he can be contacted at vaibhav4595@gmail.com.

implicit (ellipsis). For example, in Table 1, turn 4 contains the pronoun 'it', which explicitly refers to the term 'meat' in turn 3. On the other hand, turn 5 does not contain any explicit pronoun marker, but implicitly questions whether 'boer goats are good for meat' by grounding itself in turns 3 and 4.

One can think of coreference resolution as a special case of context resolution. However, off-the-shelf coreference models struggle with conversational questions (Dalton et al., 2019). Contextualized questions lead to an ineffective representation of the desired information need, causing a poor retrieval of informative passages.

Recent (and relatively successful) attempts to conversational search have focused on rewriting the conversational questions into de-contextualized questions that contain all the necessary information. These de-contextualised questions are then used for retrieval. For instance, one of the best performing systems submitted to TREC CAsT was the ATeam's query rewriter which used a pre-trained GPT-2 model (Radford et al., 2019) to rewrite questions. More recently, Yu et al. (2020) fine-tuned GPT-2 using a large amount of ad-hoc search sessions for rewriting questions.

However, the performance of the above methods on passage ranking has a ways to go compared to non-automatic methods where ground truth query reformulations are used (Dalton et al., 2019). Both automatic and non-automatic methods use standard BERT (fine-tuned on passage ranking) for reranking passages. Thus, even if automatic query reformulations are perfect, their overall passage retrieval performance will have an upper bound which will be equal to what the ground truth reformulations can achieve. Also, the current automatic methods do not aim to adapt the reranker to the conversational setting.

Similar to the idea behind pseudo-relevance feedback, this paper starts by motivating that going beyond the ground truth question reformulations by incorporating additional terms from the dialog history and the top-retrieved passages (retrieved during the first-round of retrieval), which might not be present in the ground truth reformulations, can help in improving passage retrieval. For example: turn 6 in Table 1 is self-sufficient i.e there is no need to reformulate it. However, adding the term 'goat' to the question can help in improving the retrieval performance. At the same time, this paper also aims to adapt the typical ad-hoc reranker to the conversational setting by a simple means of data fusion.

Adding to the above challenges, the TREC CAsT dataset also has a limited number of training examples which might hinder the effective training of models. Navigating through all the above presented issues, this paper presents a ranking pipeline aimed at improving the performance of passage retrieval in a conversational setting. The entire pipeline consists of two major components: **Conversational Term Selection** (CTS) and **Multi-View Reranking** (MVR).

CTS is designed to handle the first-round retrieval of passages. Given an input question, CTS utilises BERT (Devlin et al., 2018) in conjunction with a linear classifier to perform a binary classification over terms provided by the dialog history. This results in a set of conversational terms which are concatenated with the input question and queried to a search engine in order to retrieve passages. As mentioned earlier, the limited amount of training data provided in the CAsT dataset hinders an effective training of the classifier used in CTS. To this end, the CTS classifier is trained using weak supervision by utilising dialogs from a task-oriented dialog dataset (Quan et al., 2019).

On the other hand, MVR is designed for reranking. It reranks the passages obtained through CTS. By a simple means of data fusion it adapts an ad-hoc reranker to the conversational setting. It first begins by constructing three different views of the information need embedded within an input question. Each view is a query in its own sense and aims at extracting different types of contextual information. The first view is based on the reformulation of the input question. Using a similar mechanism as pseudo-relevance feedback, the second and the third view use the dialog history and the passages retrieved during the first-round of retrieval in order to expand the input question. Later, MVR individually uses each view to rerank passages using BERT (which is fine-tuned for passage ranking). Finally, it performs a fusion over the rankings produced by the individual views.

The experimental results show that the entire pipeline is highly effective for passage retrieval i.e it improves the first-stage retrieval of passages as well the overall accuracy in a reranking pipeline. On the key metric of NDCG@3, the proposed pipeline achieves a relative performance improvement of 14.8% over the state-of-the-art baseline.

It also performs 3% relatively better than the Oracle which uses ground truth query reformulations for ranking of passages. To the best of our knowledge, no automatic system had been able to beat the Oracle until now.

2 Related Work

Previous research provides guidance about the requirements of conversational search systems. For example, Radlinski and Craswell (2017) described desirable key features for conversational information retrieval systems. Trippas et al. (2018) identified commonly-used interactions and informed conversational search system design by studying the conversations of real users. Thomas et al. (2017) released the Microsoft Information-Seeking Conversation (MISC) dataset, which mimics conversational assistants such as Cortana.

Prior to the CAsT dataset, researchers often utilised dialog response reranking tasks (Zhou et al., 2016; Wu et al., 2016), conversational question-answering (Choi et al., 2018; Reddy et al., 2019) and voice based recommendation (Zhang et al., 2018) as a 'proxy' for a conversational search setting. For example, Kenter and de Rijke (2017) presented an end-to-end trainable Attentive Memory Network for reading comprehension. Yang et al. (2018) proposed a method for dialog response ranking that incorporates external knowledge into deep neural models with pseudo-relevance feedback. Aliannejadi et al. (2019) formulated the task of asking clarifying questions in open-domain information-seeking conversational systems.

The introduction of the CAsT dataset (Dalton et al., 2020) has brought in a new range of systems which focus on conversational information seeking. The ATeam's run (Dalton et al., 2019) of TREC CAsT 2019 utilises GPT-2 (Radford et al., 2019) to translate questions augmented with previous turns of the conversation into stand-alone questions that are afterwards used to retrieve relevant passages. Their question rewriting approach is based on a transfer learning paradigm. On the other hand, to overcome the problem of limited data, Yang et al. (2019) propose two ad-hoc approaches based on historical question expansion and historical answer expansion in combination with BERT (Devlin et al., 2018) for ranking passages. More recently, Yu et al. (2020) utilise rule-based and self-supervised methods to generate weak supervision data using large amounts of ad hoc search sessions which in turn are used to fine-tune GPT-2 in order to rewrite conversational queries. The rewritten queries are then used for ranking passages.

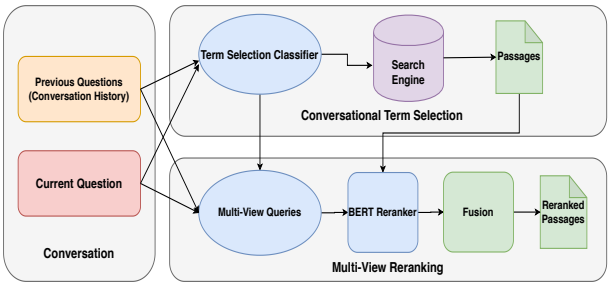

Figure 1: An overview of the proposed pipeline.

3 Proposed Approach

Figure 1 presents an overview of the approach. The pipeline consists of i) Conversational Term Selection (CTS), and ii) Multi-View Reranking. The CTS component uses a classifier to select relevant contextual terms from dialog history. The classifier's predictions are then used to convert the given question into a query before submitting it to the search engine. The top R passages obtained with this method are passed to the Multi-View Reranking component which begins by projecting the input question into Multi-View Queries. The first view is based on question reformulation, the second view utilizes the CTS predictions, whereas the third view uses the top retrieved passages obtained using CTS. Each of these views are then individually used for reranking. The process of reranking is performed using a BERT-based reranker. Finally, the rerankings produced using the individual views are combined using fusion.

CTS and MVR are described in details below.

3.1 Conversational Term Selection (CTS)

Figure 2 provides an overview. CTS is designed for first-stage passage retrieval. First, given a conversational topic T, an utterance question u_t produced during turn t, the set of questions $T_{t-1} = \{u_1, u_2 \dots u_{t-1}\}$ produced in the turns prior to t where each u_i comprises of individual terms $\{u_{i,1}, u_{i,2}, u_{i,3} \dots\}$ ($u_{i,j}$ represents the j^{th} term of the i^{th} utterance), the CTS classifier classifies each term u_{ij} present within the questions of T_{t-1} as 0 or 1.

Thus, term selection becomes a binary classification problem where each term occurring in previous turns should either be selected or removed. Each selected term acts a relevant contextual term which can help in improving retrieval. This process can

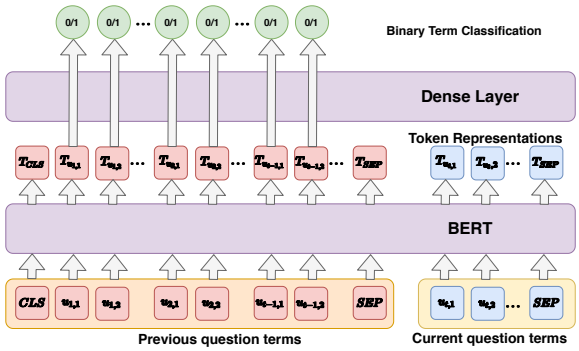

Figure 2: The Conversational Term Selection (CTS) classifier.

also be thought of as a query expansion technique, albeit different from pseudo relevance feedback. Instead of conditioning on the retrieved documents for finding appropriate expansion terms, the previous turns of a conversation are used for doing so.

Later, the selected terms along with the input question are queried to a search engine for retrieving passages. Unlike MVR, CTS does not project an input question into multiple views at the time of retrieval. This would be unnecessary as the first-round of retrieval only focuses on retrieving relevant passages. It does not focus on bringing the highly relevant passages at the top. The job of bringing the most highly relevant passages at the top of the ranked list is that of the reranker. Nonetheless, it would still be interesting to see how a multi view initial ranking would affect the final reranking. This is left for future work.

3.1.1 Training Data Creation

For each question within a conversation topic T, the CAsT dataset also provides a ground truth reformulated version.

For each $u_t \in T$, a ground truth reformulated question r_t is also provided. These reformulated questions can be leveraged to create data for training the CTS classifier. First, for each question u_t, a set of conversational terms CT_t is created that help resolve the context of u_t. The set CT_t consists of terms present in r_t but not in u_t i.e., $CT_t = \{r_{tj} | \forall r_{tj} \notin u_t\}$. Next, the terms present in questions ranging from $u_1 \dots u_{t-1}$ are marked as 0 or 1 depending on whether they were a part of the set CT_t or not. This process helps in forming the required dataset.

3.1.2 Training with Weak Supervision

To overcome the limitations caused by the small size of CAsT trainig data and to achieve better generalization capabilities, the CTS classifier is trained using weak supervision. This is done by additionally training the classifier with examples from a task-oriented dialog dataset.

Quan et al. (2019) manually constructed a dataset on the basis of the public dataset CamRest676 (Wen et al., 2016), which is meant for training task oriented dialog systems. This dataset is particularly suitable for training the CTS classifier because i) the utterances within a conversation consists of ellipsis and coreferences which can help in providing better signals, and ii) each utterance is accompanied by its ground truth reformulation, thereby making it slightly straightforward to manipulate the dataset in order to come up with examples suitable for training the CTS classifier. This can be done by simply using the process of data creation as described above (Section 3.1.1).

Note that this might lead to the creation of imprecise examples as the CamRest676 dataset does not provide any information about how much should one look back further within the dialog history in order to resolve the context of the input utterance. Due to this reason, training on the created data leads to a weakly-supervised classifier.

3.1.3 BERT with Linear Classifier

CTS classifier uses BERT in conjunction with a linear layer to select conversational terms. Given the question in the current turn u_t, and the previous questions $u_1 \dots u_{t-1}$, BERT is used to generate the token representations of individual terms within the questions. Next, the token representations of the terms within $u_1 \dots u_{t-1}$ are individually passed as inputs to the linear layer in order to decide whether to select the individual terms or not (as in Figure 2).

3.1.4 First-Stage Passage Retrieval

After the CTS classifier selects the necessary conversational terms, the selected terms are concatenated with the current question (input question) to define a query that can be used for passage retrieval. Passage retrieval is performed by the Indri search engine (Strohman et al., 2005) with the query wrapped around a 'combine' operator. Passages are indexed without the removal of stopwords. Stemming of the passages is done using the Krovetz stemmer.

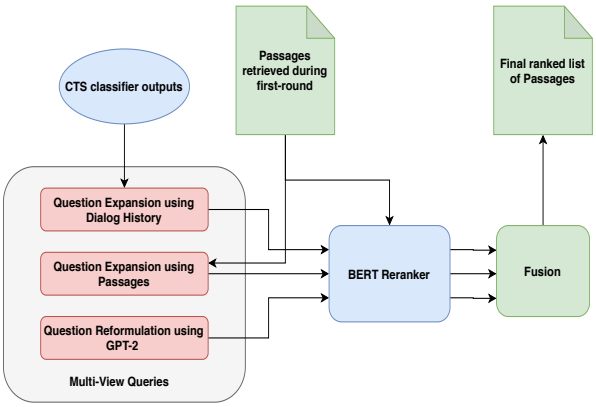

Figure 3: An overview of MVR architecture.

3.2 Multi-View Reranking (MVR)

An overview of the architecture can be seen from Fig. 3. The input question is first converted into Multi-View Queries. Each view is produced using a different source and serves a different kind of purpose. The first view is constructed using the terms present in the dialog history. The second view is constructed using the terms present in the retrieved passages. The third view is the reformulation of the input question. Each of these views are individually used for reranking passages. Finally, MVR performs a fusion over the ranked list produced by the individual views.

3.2.1 Multi-View Queries

As mentioned earlier, MVR constructs three different views of the queries. Each view looks at a different source of information and tries to represent the information need embedded within the input question in a different manner. The views are described below:

1. **Question Expansion using Dialog History**: The outputs of the CTS classifier obtained via the CTS component is concatenated with the input question.

2. **Question Expansion using Passages**: Given the input question and a few of the R passages produced during the first-round of retrieval, the CTS classifier first classifies the terms present in each of the selected passages. The positively classified terms are then concatenated with the input question.

3. **Query Reformulation using GPT-2**: This view adopts the method presented by Yu et al. (2020). The input question is reformulated to a de-contextualized question using GPT-2 (Radford et al., 2019). For this task, GPT-2 is fine-tuned using weakly supervised data obtained from large amounts of ad hoc search sessions aimed at mimicking conversational style questions.

Note that all three views attemp to present the same information need in a different manner, albeit with different granularities. Query expansion with passages attempts to de-contextualize the input question using the retrieved passages. Query Reformulation using GPT-2 attempts to produce a well-formed natural language reformulation of the input question. Whereas, Question Expansion using Dialog History is a type of pseudo-relevance feedback mechanism which aims at selecting terms from the dialog history in order to keep the focus of the input question on topic.

There is a slight difference between Query Reformulation view and the Query Expansion view. Query reformulation only aims to reformulate the question by handling ellipsis or co-references. However, query expansion aims to extend beyond that by selecting additional terms which can help in keeping the focus of the question on topic and at the same provide more informational terms.

3.2.2 BERT Reranker

Each view is individually used to rerank the passages produced during the first-round of retrieval using a BERT-based reranker. Here, the BERT-base model is fine-tuned for the task of ad hoc passage ranking using the MS-MARCO passage ranking dataset. Following (Nogueira and Cho, 2019), BERT-base is fine-tuned on 2% of the training data.

Note that the reranker used here is the same as the one used by Yu et al. (2020) and ATeam (Dalton et al., 2019). However, MVR aims to extend the capabilities of the reranker and adapts it to the conversational setting by exposing it with multiple forms of the input question.

3.2.3 Fusion

This step within MVR is extremely straightforward and aims to merge the rankings produced by the individual views. This is done by a simple aggregation of the scores produced for a passage by each of the individual views.

4 Experimental Methodology

Dataset: The CAsT dataset (Dalton et al., 2020) consists of 30 training topics (9 questions per topic,

269 in total), and 50 test topics (9.6 questions per topic, 478 in total). However, relevance judgments are available only for 20 test topics (173 questions). Therefore, evaluation is performed only over the 20 judged topics. The passages in CAsT dataset are borrowed from MSMARCO and TREC Complex Answer Retrieval Track. On the other hand, the annotated CamRest676 dataset, which is used for weak supervision, consists of 676 dialogs with coreferences and ellipsis annotations (Quan et al., 2019).

Parameter Settings: The CTS classifier uses BERT-base-uncased model and is fine-tuned for 5 epochs. It uses Adam (Kingma and Ba, 2014) as the optimiser with a learning rate of 5×10^{-5}. While training, the maximum length of the context is clipped to 100, and the length of the input question is clipped to 30. On the other hand, MVR uses a BERT-base-uncased model fine-tuned on 2% of the MS-MARCO Passage Ranking dataset. During training, the maximum length of the query is clipped to 64, whereas that of the passage is clipped to 256. The first-round of retrieval by CTS leads to a total of 1000 passages per input question. During the reranking phase, only top R of the initial passages are reranked by MVR.

Evaluation Metrics: The performance of the CTS classifier is measured using Precision (Prec), Recall and F1. The passage retrieval performance is measured using Normalized Discounted Cumulative Gain at a ranking depth of 3 (NDCG@3) which is the main metric prescribed by TREC CAsT. The results are also evaluated using Mean Reciprocal Rank (MRR).

5 Experiments and Results

This section is divided into two halves. The first half evaluates the performance of CTS. The second half evaluates the performance of MVR i.e the result of using the entire pipeline.

5.1 Efficacy of CTS

Experiments over CTS aim to answer the following questions:

- **Q1**: How well does the CTS classifier perform?
- **Q2**: To what extent does incorporating weak supervision help improve the performance of the classifier?

Prev. turns Used	Prec	Recall	F1
1	0.462	0.453	0.457
2	0.481	0.338	0.397
3	0.493	0.304	0.377
4	0.566	0.266	0.363
5	0.567	0.282	0.377

Table 2: Accuracy of the CTS Classifier when trained on CAsT topics with varying amounts of history.

Supervision Type	Prec	Recall	F1
Add Only	0.88	0.327	0.476
Add + 1 previous	0.617	0.744	0.674
Add + 2 previous	0.706	0.621	0.661
Add + 3 previous	0.695	0.680	0.687
Add + 4 previous	0.724	0.684	0.703
Add + 5 previous	0.709	0.691	0.705
Add + All previous	0.698	0.758	**0.727**

Table 3: Accuracy of CTS Classifier with trained using Weak Supervision.

- **Q3**: What is CTS's first-round retrieval performance?

5.1.1 Q1: Performance of CTS Classifier

Table 2 shows the performance of the classifier when trained on CAsT training data. It also reflects the effects of training the classifier with different amounts of dialog history. The CAsT training set is split into training and validation in a ratio of 4:1. In the entire setup, the classifier is trained with restricted amount of dialog history and tested with the entire dialog history made available to it. This setup helps understand its generalization capabilities.

It is clear that the precision of the classifier increases with an increase in the amount of dialog history. However, the trend for recall is the exact opposite. The F1 scores remain quite low for all the cases. These trends clearly depict the data scarcity issue which has been mentioned in Section 1 and 3.1.2. The classifier's generalization capabilities are hindered by the low number of training examples used in fine-tuning.

5.1.2 Q2: Effect of Training with Weak-Supervision

Table 3 shows the performance of the classifier when trained with weak supervision. In the table, '**Add Only**' refers to the model trained only on the modified examples obtained from the additional dialog dataset. Whereas, '**Add +** k **previous**' refers to the model trained by combining examples from

the additional dialog dataset and examples from the CAsT training set (with the dialog history clipped to k previous turns).

On comparing the statistics in Table 2 and Table 3, it is evident that the precision of the classifier improves greatly when trained on 'Add Only'. However, there is no improvement in its recall. On the other hand, it seems that the increase in the amount of k in 'Add + k previous' (with the exception of $k = 1$) leads to an increase in the classifier's recall. This trend is in contrast with Table 2 where the recall decreases with increasing number of turns. A possible reason could be the fact that presence of weakly supervised examples forces better grounding of the coreference terms within the dialog.

The best F1 score is obtained with 'Add + All previous'. This provides almost a 60% improvement over the best result in Table 2. Thus, it is clear that weakly supervised data helps in improving performance.

It might be possible that the CTS classifier ends up selecting a few noisy terms. This might lead to low scores for some of the relevant passages during the first-round of retrieval. However, MVR, by utilising three different types of information should be able to boost the scores for those relevant passages, thereby bringing them at the top of the ranked list.

5.1.3 Q3: Passage Retrieval Performance

The performance of the proposed method is compared to that of four baselines. **Base1** uses the original questions without any modifications for retrieval. **Base2** appends the nouns, verbs and adjectives from the preceding turns to the current question before retrieval. **AllenCoref** (Lee et al., 2017) performs co-reference resolution to re-write the input question before performing passage retrieval. Finally, **Spacy N-Coref** uses Spacy's neural coreference model to do the same as AllenCoref.

The results are shown in Table 4. The results of CTS are based on the model trained on 'Add + All Previous'. The poor performance of Base1 depicts the need for finding appropriate contextual terms for effective query creation. On the other hand, the poor performance of AllenCoref and NeuralCoref show that co-reference models were unable to resolve the questions effectively, thereby confirming that off-the-shelf co-reference methods struggle with conversational style questions. Their results might also hint that co-reference alone is not enough for retrieval. Base2, which simply chooses the nouns, verbs and adjectives, performs better than co-reference models. However, selecting all the nouns, verbs and adjectives might end up adding noise (of undesirable proportions) to the created query and could cause a drift in its topic. By alleviating this issue precisely, CTS seems to outperform the other methods.

Method	NDCG@3	MRR
Base1	0.153	0.317
Base2	0.271	0.538
AllenCoref	0.206	0.404
Spacy N-Coref	0.191	0.398
CTS	**0.294**	**0.558**

Table 4: Retrieval Performance of CTS

Method	NDCG@3	MRR
Pgbert	0.413	0.665
h2oloo_RUN2	0.434	0.714
CFDA_CLIP_RUN7	0.436	0.715
GPT-2 Rewrite	0.492	0.780
Oracle	0.545	0.842
MVR	**0.565**	0.833

Table 5: Reranking Performance of MVR

Here the retrieval performance of CTS is not compared with the state-of-the-art baselines. This is because the baselines only report their final results which are obtained after the reranking phase. It is also unclear how the baselines conduct their first-round of retrieval. However, the final results of this paper make a fair comparison with the final results of the state-of-the-art.

5.2 Efficacy of MVR

This part aims to measure the effectiveness of the entire pipeline by measuring the final reranking performance of MVR. Experiments over MVR aim to answer the following questions:

1. **Q1**: What is MVR's reranking performance?
2. **Q2**: What is the effect of adding different views?

5.2.1 Passage Reranking

The results can be seen from Table 5. As is evident, the performance of MVR is compared against several baselines. Pgbert, h2oloo_RUN2 and CFDA_CLIP_RUN7 are the top three automatic runs submitted to TREC CAsT. **Pgbert** uses GPT-2 for query rewriting and later reranks the passages using BERT. Both **h2oloo_RUN2** and

CFDA_CLIP_RUN7 use a heuristic-based method for query expansion. Later, they use the title of the conversation and the expanded query for reranking of passages using BERT. Note: in a real scenario a user may not necessarily provide a title to the conversation before starting one. Thus, h2oloo_RUN2 and CFDA_CLIP_RUN7 simply utilise additional information which may not be readily available. MVR does not make use of any such 'given' additional information. **GPT-2 Rewrite** (Yu et al., 2020) uses a fine-tuned GPT-2 for question reformulation and reranks passages using BERT. Finally, the Oracle uses ground truth question reformulations for reranking via BERT.

Out of the 1000 passages retrieved by CTS, MVR reranks the top R of them. The Query Expansion using Passage view is constructed using the top K passages, out of the 1000 total retrieved during the first-stage of retrieval. Both R and K are tuned and set as 500 and 50 respectively.

As is evident from Table 5, the MVR is able to outperform all the automatic baselines by quite a substantial margin. By using a sophisticated mechanism for conversational term selection and without using any additional information like the title, MVR is able to perform better than h2oloo_RUN2 and CFDA_CLIP_RUN7, both of which utilise title and are based on a heuristic method for query expansion. This clearly depicts that the expansion terms selected by CTS helps MVR to produce an effective ranked list of passages.

Both Pgbert and GPT-2 rewrite use GPT-2 question reformulation. The reformulations act as the sole information for reranking passages. By accumulating three different types of information (one of which includes reformulation), MVR is able to perform better than question reformulation mechanims. MVR is able to outperform GPT-2 rewrite, which is also state-of-the-art by almost 14.8%.

On the key metric of NDCG@3, it can be said that MVR is better than the Oracle by a slight margin. Although the NDCG@3 of MVR is greater than the Oracle, its MRR is slightly lower. A reason for this could be the fact that the Oracle retrieves more relevant passages than the MVR but the MVR better ranks highly relevant passages.

One must also note that the rerankers used by all the baselines have the same configuration i.e all the rerankers are fine-tuned on the passage ranking corpus of MS-MARCO. Therefore, it would not be futile to say that the power of MVR lies within its Multi-View Queries.

Selected View(s)	NDCG@3	MRR
Passages	0.306	0.571
Dialog History	0.509	0.765
Passages + Dialog History	0.514	0.796
All Views (Full MVR)	0.564	0.833

Table 6: Reranking performance when using different views in MVR.

5.2.2 Performance of Adding Views

The results of using different views is presented in Table 6. It is clear that using the expansion using passage view does not a have good performance by itself. One of the reasons for this could be the fact that the questions asked in the CAsT conversations do not refer to any entities within the answer of the previous passages i.e the questions in CAsT can be resolved using dialog history alone. Therefore, expansion using passages by itself is not very efficient. However, it does help when combined with other views. This is because expansion using passages provides extra credit to the more highly relevant passages.

On the other hand, expansion using the dialog history view is able to perform better than the best baseline as its NDCG is higher than that of than GPT-2 rewrite (refer Table 5). It is important to note that the GPT-2 rewrite is equivalent to the question reformulation view of MVR.

Fusion of the expansion using passages view and expansion using history view provides a further improvement over expansion using history view alone. Finally, by combining the all three views together, MVR is able to provide the best result.

6 Conclusion

This paper presents a simple yet highly effective pipeline for conversational search. The pipeline consists of two components: CTS and MVR. CTS aids in first-round of passage retrieval by selecting important contextual terms from the dialog history. MVR reranks the passages obtained by CTS by expressing the information need embedded within a question in multiple forms. The combination is able to surpass the state-of-the-art and at the same time perform slightly better than the Oracle. To the best of our knowledge, no automatic system has been able to do so.

References

Mohammad Aliannejadi, Hamed Zamani, Fabio Crestani, and W Bruce Croft. 2019. Asking clarifying questions in open-domain information-seeking conversations. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 475–484.

Eunsol Choi, He He, Mohit Iyyer, Mark Yatskar, Wentau Yih, Yejin Choi, Percy Liang, and Luke Zettlemoyer. 2018. Quac: Question answering in context. *arXiv preprint arXiv:1808.07036*.

Jeff Dalton, Chenyan Xiong, Vaibhav Kumar, and Jamie Callan. 2020. Cast-19: A dataset for conversational information seeking. In *Proceedings of the 43rd annual International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Jeffrey Dalton, Chenyan Xiong, and Jamie Callan. 2019. Cast 2019: The conversational assistance track overview. In *The Twenty-Eighth Text Retrieval Conference Proceedings*.

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

Tom Kenter and Maarten de Rijke. 2017. Attentive memory networks: Efficient machine reading for conversational search. *arXiv preprint arXiv:1712.07229*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. 2017. End-to-end neural coreference resolution. *ArXiv*, abs/1707.07045.

Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage re-ranking with bert. *arXiv preprint arXiv:1901.04085*.

Jun Quan, Deyi Xiong, Bonnie Webber, and Changjian Hu. 2019. Gecor: An end-to-end generative ellipsis and co-reference resolution model for task-oriented dialogue. *arXiv preprint arXiv:1909.12086*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9.

Filip Radlinski and Nick Craswell. 2017. A theoretical framework for conversational search. In *Proceedings of the 2017 conference on conference human information interaction and retrieval*, pages 117–126.

Siva Reddy, Danqi Chen, and Christopher D Manning. 2019. Coqa: A conversational question answering challenge. *Transactions of the Association for Computational Linguistics*, 7:249–266.

Trevor Strohman, Donald Metzler, Howard Turtle, and W Bruce Croft. 2005. Indri: A language model-based search engine for complex queries. In *Proceedings of the international conference on intelligent analysis*, volume 2, pages 2–6. Citeseer.

Paul Thomas, Daniel McDuff, Mary Czerwinski, and Nick Craswell. 2017. Misc: A data set of information-seeking conversations. In *SIGIR 1st International Workshop on Conversational Approaches to Information Retrieval (CAIR'17)*, volume 5.

Johanne R Trippas, Damiano Spina, Lawrence Cavedon, Hideo Joho, and Mark Sanderson. 2018. Informing the design of spoken conversational search: Perspective paper. In *Proceedings of the 2018 Conference on Human Information Interaction & Retrieval*, pages 32–41.

Tsung-Hsien Wen, David Vandyke, Nikola Mrksic, Milica Gasic, Lina M Rojas-Barahona, Pei-Hao Su, Stefan Ultes, and Steve Young. 2016. A network-based end-to-end trainable task-oriented dialogue system. *arXiv preprint arXiv:1604.04562*.

Yu Wu, Wei Wu, Chen Xing, Ming Zhou, and Zhoujun Li. 2016. Sequential matching network: A new architecture for multi-turn response selection in retrieval-based chatbots. *arXiv preprint arXiv:1612.01627*.

Jheng-Hong Yang, Sheng-Chieh Lin, Jimmy Lin, Ming-Feng Tsai, and Chuan-Ju Wang. 2019. Query and answer expansion from conversation history. In *The Twenty-Eighth Text Retrieval Conference Proceedings*.

Liu Yang, Minghui Qiu, Chen Qu, Jiafeng Guo, Yongfeng Zhang, W Bruce Croft, Jun Huang, and Haiqing Chen. 2018. Response ranking with deep matching networks and external knowledge in information-seeking conversation systems. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 245–254.

Shi Yu, Jiahua Liu, Jingqin Yang, Chenyan Xiong, Paul Bennett, Jianfeng Gao, and Zhiyuan Liu. 2020. Few-shot generative conversational query rewriting. In *Proceedings of the 43rd annual International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Yongfeng Zhang, Xu Chen, Qingyao Ai, Liu Yang, and W Bruce Croft. 2018. Towards conversational search and recommendation: System ask, user respond. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pages 177–186.

Xiangyang Zhou, Daxiang Dong, Hua Wu, Shiqi Zhao, Dianhai Yu, Hao Tian, Xuan Liu, and Rui Yan. 2016. Multi-view response selection for human-computer

conversation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 372–381.