Exploiting Simulated User Feedback for Conversational Search: Ranking, Rewriting, and Beyond

Paul Owoicho*
University of Glasgow
Glasgow, Scotland, UK
p.owoicho.1@research.gla.ac.uk

Ivan Sekulić*
Università della Svizzera italiana
Lugano, Switzerland
ivan.sekulic@usi.ch

Mohammad Aliannejadi
University of Amsterdam
Amsterdam, The Netherlands
m.aliannejadi@uva.nl

Jeffrey Dalton
University of Glasgow
Glasgow, Scotland, UK
jeff.dalton@glasgow.ac.uk

Fabio Crestani
Università della Svizzera italiana
Lugano, Switzerland
fabio.crestani@usi.ch

ABSTRACT

This research aims to explore various methods for assessing user feedback in mixed-initiative conversational search (CS) systems. While CS systems enjoy profuse advancements across multiple aspects, recent research fails to successfully incorporate feedback from the users. One of the main reasons for that is the lack of system–user conversational interaction data. To this end, we propose a user simulator-based framework for multi-turn interactions with a variety of mixed-initiative CS systems. Specifically, we develop a user simulator, dubbed ConvSim, that, once initialized with an information need description, is capable of providing feedback to the system’s responses, as well as answering potential clarifying questions. Our experiments on a wide variety of state-of-the-art passage retrieval and neural re-ranking models show that effective utilization of user feedback can lead to 16% retrieval performance increase in terms of nDCG@3. Moreover, we observe consistent improvements as the number of feedback rounds increases (35% relative improvement in terms of nDCG@3 after three rounds). This points to a research gap in the development of specific feedback processing modules and opens a potential for significant advancements in CS. To support further research in the topic, we release over 30 000 transcripts of system-simulator interactions based on well-established CS datasets.

CCS CONCEPTS
- Information systems → Users and interactive retrieval.

KEYWORDS
user simulation, conversational information seeking, mixed-initiative

ACM Reference Format:

1 INTRODUCTION

The primary goal of a conversational search (CS) system is to satisfy the user’s information need. However, there are several challenges that arise when it comes to CS, as opposed to traditional ad-hoc search. An important tool for addressing these challenges is the use of mixed-initiative techniques. Under the mixed-initiative paradigm, the conversational search system can proactively initiate prompts, such as suggestions, warnings, or questions, at any point in the conversation. In recent years, mixed-initiative conversational search has received significant attention from the information retrieval (IR) research community, leading to advancements in various aspects of this field, including conversational passage retrieval [18, 38], query rewriting in context [54], intent prediction in conversations [39], and asking clarifying questions [5].

Despite the abundance of research on various components of mixed-initiative search systems, little has been done to study the impact of user feedback. Users can provide explicit feedback on the quality of system’s responses, as well as answer potential questions prompted by the system. Such feedback is beneficial to mixed-initiative CS systems and can provide valuable information on user’s needs. Moreover, feedback can have a great effect on how conversation is shaped by, e.g., giving the system the chance to recover from an initial failed attempt [65]. Despite its significance, lack of research in this area can be attributed to the difficulty of collecting appropriate data containing user feedback.

Furthermore, evaluation of CS systems is arduous [29, 36]. Typically, it requires the actual users to interact with the system, presenting their information needs, answering potential questions, and providing feedback. Such studies are expensive and time consuming, often requiring a large number of experiments to properly evaluate specific approaches. That is even more the case with mixed initiatives, as the number of possible conversations is essentially limitless [10]. An attempt to address this issue is to compile offline collections aimed at specific challenges in conversational search [4, 18, 38]. Existing data collections are mainly built on online human–human conversations [38], synthetic human–computer interactions [18], and multiple rounds of crowdsourcing [4]. No existing
data collections, however, feature explicit user feedback extensively in a conversation, thus limiting research in this area. Moreover, such corpus-based evaluation paradigms usually remain limited to single-turn interactions and do not take into account the interactive nature of CS, not to mention being limited to non-generative models.

To address the vicious circle composed of the lack of research on feedback utilization and the lack of appropriate data, we develop a comprehensive experimental framework based on simulated user–system interactions, as shown in Figure 1. The framework allows us to evaluate multiple state-of-the-art mixed-initiative CS systems, addressing several challenges, including contextual query resolution, asking clarifying questions, and incorporating user feedback.

Existing work [2] aims to study the effect of different mixed-initiative strategies on retrieval, however, their findings are limited to a single data collection, and lexical-based retrieval techniques. More recently, work on user simulators for conversational systems aims to address these limitations, however, it remains limited to pre-defined or templated interactions [46, 63] or focus only on one aspect of the search system, e.g., answering clarifying questions [48]. To address these limitations, we propose a user simulator called ConvSim, capable of multi-turn interactions with mixed-initiative CS systems. Given a textual description of the information need, ConvSim answers prompted clarifying questions and provides both positive and negative feedback, as necessary. Recent advancements in large language models (LLMs), e.g., GPT-3 [13], PALM [15], open the possibilities of addressing such nuanced tasks. Thus, we base core functionalities of the proposed simulator on LLMs. Finally, the ConvSim addresses the limitation of pre-built corpora, as the simulator’s behavior adapts to the system’s response.

Our experimental evaluation shows that ConvSim can reliably be used for interacting with mixed-initiative conversational systems. Specifically, we demonstrate that responses generated by the simulator are natural, in line with defined information needs, and, unlike previous work [48], coherent across multiple conversational turns. The proposed simulator interacts with CS systems entirely in natural language, without the need to access the system’s source code or inner mechanisms. Furthermore, the experimental framework, centered around ConvSim, allows for seamless curation of synthetic data on top of existing static IR benchmarks, as the simulator–system interactions can extend over multiple conversational turns.

We stress the fact that research questions around feedback utilization in CS can hardly be answered by existing or pre-built collections. On the other hand, while the questions around leveraging user feedback could be answered through comprehensive user studies, such studies are time-consuming, expensive, and largely limited in the number of experiments we would be able to conduct.

We find significant improvements in retrieval performance of methods utilizing feedback compared to non-feedback methods, even with only a single turn of feedback. Well-established methods, such as RM3, adapted to handle explicit feedback, demonstrate relative improvement of 11% and 9% in terms of recall and nDCG@3. Further, we identify a shortcoming of standard T5 query rewriter [28] in the task of processing feedback. To address this, we propose a novel adaptation of the T5 method and achieve 10% and 16% improvements in terms of recall and nDCG@3, respectively. Similarly, incorporating answers to clarifying questions yields improvements both in recall (18%) and nDCG@3 (12%). We also find that multiple rounds of simulator–system interactions result in further improvements in retrieval effectiveness (35% relative improvement in terms of nDCG@3 after three rounds). Moreover, we observe that existing methods react poorly to certain types of feedback (e.g., positive feedback “Thanks”), leading to a decrease in performance. This points to a research gap in development of specific feedback processing modules and opens a potential for significant advancements in CS.

Our main contributions are:

• New insights into mixed-initiative CS system design, with a focus on processing users’ feedback, including explicit feedback and their answers to clarifying questions.
A user simulator, capable of multi-turn interactions with mixed-initiative search systems. We release transcripts, code and guidelines\footnote{https://github.com/grill-lab/ConvSim} to foster further research.

2 RELATED WORK

2.1 Mixed-initiative conversational search

In recent years, conversational search has attracted significant attention both from the IR and natural language processing (NLP) communities\cite{6}. To this end, Radlinski and Craswell\cite{40} propose a theoretical framework of conversational search, identifying key properties of such systems and focusing on natural and efficient information access through conversations. While some of the challenges remain similar to traditional ad hoc search, significant new ones arise in the conversational paradigm. These are surveyed in the recent manuscript of Zamani et al.\cite{62}. They include conversational query rewriting\cite{54,57}, conversational retrieval\cite{18,58} and user intent prediction\cite{39}.

One key element of conversational search is mixed-initiative, which is the interaction pattern where both the system have rich forms of interaction. Under the mixed-initiative paradigm, conversational search systems can at any point of conversation take initiative and prompt the user with various questions or suggestions. Mixed-initiative has a long history in dialogue systems with Walker and Whittaker\cite{56} identifying it as an integral part of conversations and Horvitz\cite{22} identifying key principles of mixed-initiative interactions. One of the most prominent uses of mixed-initiative is asking clarifying questions with a goal of elucidating the underlying user’s information need\cite{5,12,51,60}. The benefits of prompting the user with clarifying questions is found by multiple studies, including improving retrieval performance in conversational search\cite{3,24,44,61,64}. Clarifying questions are generally either selected from a pre-defined pool of questions\cite{3,5,41} or generated\cite{42,47,59}. While decent success has been demonstrated by various question selection methods\cite{3}, such approaches remain limited to pre-defined conversational trajectories and are not fit for a realistic search scenario. Therefore, generating a clarifying question poses itself as a natural improvement over the selection task, mitigating the need to collect all of the potential questions beforehand. Various question generation methods exist, centered around either template-based questions or LLM-based generation. In this work, we also study clarifying questions and use simulation methods to answer them. While there are benefits to clarifying questions, there is also cost to the user for these interactions\cite{7,8}. In this work we focus on their effectiveness in a simulation environment and don’t study user costs directly.

2.2 Evaluation and user simulation

Deriu et al.\cite{19} state that the evaluation method in context of conversational systems should be automated, repeatable, correlated to human judgments, able to differentiate between different conversational systems, and explainable. However, evaluating all of these elements in conversational systems is challenging. While various unsupervised and user-based evaluation methods exist\cite{19} there are key trade-offs. Liu et al.\cite{30} conduct a thorough empirical analysis of unsupervised metrics for conversational system evaluation and conclude that they correlate very weakly with human judgments, emphasizing that reliable automated metrics would accelerate research in conversational systems. Thus,\cite{19} identify user studies as a more reliable method for evaluating conversational systems, stressing the fact that such evaluation is both cost- and time-intensive.

Conversational search has similar evaluation challenges, further complicated by the retrieval of relevant documents from a large collection\cite{36}. While traditional Cranfield paradigm fits well for evaluation of ad hoc search systems, it is not easily transferable to conversational search\cite{20,29}. One of the specific challenges is that the complexity of multi-turn queries and the overall context is ignored by traditional metrics, and requires a more holistic approach\cite{21,23}.

Balog\cite{10} makes the case that simulation is an important emerging research frontier for conversational search evaluation. Pääkkönen et al.\cite{34} assess the validity of the use of simulated users in interactive IR and find it justified under a common interaction model. While user simulators are a well-established idea in IR\cite{14,31}, including applications such as simulating user satisfaction for the evaluation of task-oriented dialogue systems\cite{52} and recommender systems\cite{1,63}, their utilization in mixed-initiative conversational search is limited.

To address this, Salle et al.\cite{46} design a simulator that selects an answer to potential clarifying questions posed by the system. However, their approach is limited to pre-defined clarifying questions and pre-defined answers, making its usability restricted to a closed collection of such questions and answers. Sekulić et al.\cite{48} address that issue and design USI, a simulator capable of generating answers to clarifying questions posed by the system. Nonetheless, their approach is limited to single-turn interactions and does not take into account conversational context. Moreover, USI only addresses clarifying questions that are direct and about a single facet of the query. In this work we propose ConvSim, a simulator capable of multi-turn interactions with mixed-initiative conversational search systems. ConvSim addresses the challenges of previous work, while also further extending simulator capabilities by being able to provide positive and negative feedback to system’s responses.

3 BACKGROUND AND PROBLEM DEFINITION

In this section, we formally define the main task definition of mixed-initiative conversational search systems. We then link these to the requirements of user simulation.

Formally, a search session consists of multiple turns of the user’s utterances $u$ and the system’s utterances $s$, forming conversational history $H = [u^1, s^1, \ldots, u^{t-1}, s^{t-1}]$, with $u^t$ and $s^t$ corresponding to the user’s and system’s utterance at conversational turn $t$, respectively. One key factor is that we differentiate between discourse types of user utterances $u$, namely queries $q_u$, answers $a_u$ to clarifying questions posed by the system, and explicit feedback $f_e$ to the system’s responses. Similarly, the system’s utterance $s$ can either be a response $s_r$ aimed at satisfying the user’s information need $IN$ or a clarifying question $q_s$ aimed at elucidating the user’s information need. One of the inputs to various modules of mixed-initiative systems can as well be the ranked list of results $R = [r_1, r_2, \ldots, r_N]$. 
retrieved in response to $u_i$, where $N$ is the maximum number of results considered.

### 3.1 Mixed initiatives

A conversational search system should be able to effectively conduct contextual query understanding, document retrieval, and response generation. Moreover, under the mixed-initiative paradigm, the CS system can at any point take initiative and prompt the user with various suggestions or clarifying questions [40].

#### 3.1.1 Clarification

When necessary, e.g., in case of a user’s query being ambiguous, the CS system can ask a clarifying question, or questions, to elucidate the user’s underlying information need. Thus, the first challenge of a mixed-initiative search system is to assess the need for clarification [4]. Specifically, given the current user’s utterance $u^t$, the task is to predict whether asking a clarifying question is required, or whether the system should issue a response aimed at answering the user’s question. Thus one of the modules of the search system needs to model a function $\text{clarification\_need} = \mathcal{f}(u^t|H, D)$, where $\text{clarification\_need} \in \{0, 1\}$, indicating whether not to ask or to ask a clarifying question.

As mentioned, asking clarifying questions methods can be broadly categorized into question selection and question generation [3, 5] methods. In the first approach, given the current user utterance, $u^t$, and a conversational history $H$, the task is to select an appropriate clarifying question from a predefined pool of questions $CQ = \{cq_1, cq_2, \ldots, cq_n\}$. Formally, we model $s_{cq} = \phi(u^t|H, R, CQ)$ where $\phi$ is our question selection model. As discussed in Section 2, question generation poses itself as a necessary step in CS, going beyond selection from pre-defined corpora. Formally, the task of the question generation module is to model $\psi$ in $s_{cq} = \psi(u^t|H, R)$. In this work, we implement several state-of-the-art question selection and generation models and evaluate their performance. Moreover, we test the robustness of feedback processing modules depending on the type of clarifying question.

#### 3.1.2 Processing user feedback

A CS system needs to be able to process feedback given by the user during the conversation including both answers to clarifying questions and explicit feedback to the system’s response. Therefore, the system, in both cases, needs to update its internal state by refining its representation of the user’s information need. Formally, we define updates to the system’s interpretation of the user’s information need, as query reformulation: $u'_t = \gamma(u^t|H)$, where $\gamma$ is the query rewriting model. We note that, depending on the design choices of mixed-initiative systems, different forms of feedback, i.e., answers to clarifying questions and explicit feedback to the system’s responses, can be modeled differently — e.g., $u'_t = \gamma_1(u^t|H)$ and $u'_t = \gamma_2(u^t|H)$. Furthermore, we point out that similar methods might be used to model contextual query reformulation, which aims at resolving current user utterance in the context of conversational history: $u'_t = \gamma_3(u^t|H)$.

### 3.2 User simulation

A user simulator aims to mimic key user’s roles in MI interactions. Although Balog [10] defines several desired properties of a realistic user simulator, we focus on the simulator’s ability to capture and communicate aspects of the information need. The simulator should coherently answer any posed clarifying questions, or provide positive/negative feedback to the system’s responses. In other words, the requirements of a user simulator are complementary to the ones of mixed-initiative CS systems. Inspired by Zhang and Balog [63], we base our user interaction model on the general QREFA model for the conversational information-seeking process [53].

Formally, the user simulator needs to be able to carry out multi-turn interactions with the search system and generate a variety of different utterances: (i) $u_q$ — seek information through querying; (ii) $u_a$ — answer clarifying questions; and (iii) $u_f$ — provide feedback to systems’ responses. All of the utterances generated by the simulator need to be in line with the underlying information need $IN$. First, a simulator needs to represent its information need by constructing a query utterance $u_q = h(IN)$. Moreover, when prompted with a clarifying question utterance $s_{cq}$, the user simulator should be able to provide an answer $u_a = \theta_1(s_{cq}|H, IN)$, where $\theta_1$ denotes answer generation model. Similarly, when given a response $s_{r}$ to its query, it needs to generate feedback $u_f = \theta_2(s_{r}|H, IN)$, where $\theta_2$ is the response generation function. Figure 1 shows a components of the simulator, where $\theta_1$ and $\theta_2$ are utilized at appropriate stages.

Asking too many clarifying questions or providing unsatisfactory responses might impair user’s satisfaction with the search system [65]. Thus, a simulator should encapsulate similar behaviors. Following Salle et al. [46], we introduce the notion of patience $p \in \mathbb{R}^+$ — a parameter that indicates how many turns of feedback a simulated user willing to provide. Simulator decreases its patience $p$ after each turn in which it has to provide feedback, terminating the conversation once $p = 0$. A conversation is stopped by the simulator either when $IN$ is satisfied or when patience runs out.

#### 3.2.1 Naturalness and usefulness of generated answers

In order for simulator’s behavior to be similar to real users [10], both answers $s_a$ and feedback $s_f$ need to be relevant, in coherent natural language, and consistent with information need $IN$. Following Sekulić et al. [48], we assess naturalness and usefulness of the generated answers to clarifying questions. Naturalness refers to the utterance being in fluent natural language and likely generated by humans [35, 45]. We ground our definition of usefulness in previous work assessing clarifying questions [44] and their answers [48]. Specifically, it captures whether answers and feedback generated by the simulator are consistent with the provided information need, and can be related to adequacy [50] and informativeness [16]. Moreover, by extending the evaluation to the multi-turn setting, we are also evaluating simulator’s context awareness.

#### 3.2.2 Feedback

Explicit feedback $u_f$, generated in response to the systems’ responses, needs to be reliable and accurate. To this end, at each turn $u'_q$, the system returns response $s_{r+1}$ and the simulator generates feedback $u^{r+1}_f$. Moreover, the utterance $u^{r+1}_f$ is externally annotated as positive or negative feedback. Our aim is to measure correlation of retrieval performance at turn $u'_q$ and type of feedback $u^{r+1}_f$ (positive or negative). Finally, we assess potential differences, as measured by retrieval metrics, between turns that received positive vs negative feedback. Positive feedback should be generated in cases where performance is high, while negative feedback should be given when performance is low.
4 METHODOLOGY

4.1 Proposed simulator framework

We propose ConvSim, a Conversational search Simulator, capable of multi-turn interactions with the search system in a conversational manner. We design ConvSim to satisfy the requirements defined in Section 3.2. As such, the simulator needs to encapsulate different behaviors across utterances of various discourse types, including querying $u_q$, as well as providing feedback $u_f$.

We conduct our simulator experiments within the framework of a conversational pipeline that encapsulates the commonly used components in a mixed-initiative conversational search pipeline: query rewriting, passage retrieval, passage reranking, clarifying question selection and generation, and response generation. The framework is depicted in Figure 1. It enables seamless multi-turn exchange of user simulator utterances $u$ and system’s utterances $s$, detailed in Section 3. The framework includes a suggested logical exchange of the utterances, i.e., when the system produces a response $s_r$, the simulator is tasked to provide feedback $u_f$. Likewise, when posed with a clarifying question $c_q$, the simulator needs to provide an answer $u_q$. Such interactions continue as long as simulator patience $p > 0$ and $IN$ is not satisfied. Moreover, we design this framework to be flexible, allowing us to easily configure and (re)arrange the steps per our experimental needs. At the heart of this framework is a conversational turn representation that holds all relevant properties about a turn, such as a user query, system response, conversational context, and retrieved documents. We refer to our codebase for the implementation details of this experimental framework.

Specifically, we initialize ConvSim with an information need description $IN$, specific to each turn. This ensures the responses generated by ConvSim are consistent with the user information need and guide the conversation towards the relevant information.

We model feedback generation functions $θ_1$ and $θ_2$ detailed in Section 3.2 using LLMs. Given the focus of our experiments, we implement each of the simulator’s possible actions (clarifying question answering for $θ_1$, feedback generation for $θ_2$) as steps in the conversational pipeline framework described below.

4.1.1 Implementation details. We build ConvSim on top of OpenAI’s Text-Davinci-003 [13] model using few-shot prompting. We use OpenAI’s completions API endpoint with the following parameter settings based on the author’s guidelines [13] and initial empirical exploration:

- **max_tokens**: 50. This prevents the model from generating overly long responses but is also sufficient enough for the model to generate clarifying questions in addition to negative feedback or to expand a bit on its answers to clarifying questions.
- **temperature**: 0.5. This is a halfway point between a very conservative and risky model. While we want creative outputs, we also want the responses to be on topic.
- **frequency_penalty**: 0.2. This discourages the model from generating previously generated tokens (i.e., repeating itself).
- **presence_penalty**: 0.5. This encourages the model to introduce new topics. In the same way as the temperature parameter, this enables fairly novel responses that are always on topic.

For a given turn $t$, we prompt the model with a task description (i.e., whether to generate an answer to a clarifying question or feedback to system’s response), a description of the information need $IN$, sample transcripts between a user and a system with the desired behavior, and a transcript of the conversational history $H$ between the user and system up to turn $t$. The exact prompts used can be found in our codebase. We do not explicitly implement the information seeking model $u_q = h(IN)$. Instead, we take the initial query $u_q$ directly from the dataset to ensure fair comparisons between non-feedback and feedback utilizing methods described above.

4.2 Evaluation Data

We primarily use the TREC CAsT [33] benchmark, designed for the development and evaluation of conversational search systems. CAsT is composed of a series of fixed conversations, each with a pre-determined trajectory and containing a series of topical user utterances and canonical responses. We focus on year 4 because it is the only dataset that includes mixed-initiative interactions.

Because each turn in CAsT does not have an $IN$ description, we augment it by adding turn-level information need descriptions. Specifically, two expert annotators independently study each CAsT utterance in the conversation context and describe the full information need in a sentence. We decide on the length of the information, following the typical topic description in the TREC Web track topic list [17]. We instruct the annotators to take into account various sources of information such as the canonical responses and the rewritten queries. The final goal is to generate a self-contained description for each user utterance in CAsT. One could argue that the human rewritten utterances would be sufficient for this aim. In our preliminary analysis, we discover that the re-written utterances miss various contextual information that makes them dependent on the overall conversation context. We compare the generated information need descriptions by the two annotators. In case of minor differences, we select either of them. However, in cases where the difference is major there is discussion until agreement.

4.3 Mixed-initiative systems

4.3.1 Compared methods. We focus our investigations on the effects and ways of using simulated user feedback and answers to clarifying questions for downstream retrieval. In order to analyze the effects of feedback processing modules, we compare their performances against the following non-feedback baselines which do not use any initiative or simulation:

- **Organizer-auto** is a competitive baseline used in the TREC CAsT shared task over the past two years. First, it reformulates the user query with a generative T5 query rewriter fine-tuned on the CANARD dataset [2]. As context, the rewriter takes in all previous turn queries and system responses as input: $u_q^C = γ_3(u_q^C|H)$. No special considerations are made for cases where the input token length exceeds the model’s limit (i.e., 512 tokens). Next, it uses Pyserini’s 3 BM25 implementation ($k_1=4.66, b=0.82$) to retrieve the top 1 000 documents from the collection and re-ranks it’s constituent passages with a point-wise T5 passage ranker (MonoT5) [32] trained

---

[4] https://huggingface.co/castorini/t5-base-canard
on MSMARCO [9]. Finally, a BART model summarizes the top 3 passages to output a system response. We run organizer-manual on the CAsT benchmark using the manually reformulated queries at each turn for every conversation in the dataset. As these manual rewrites are context-free, this baseline represents an upper bound for retrieval performance without initiative or simulated responses using CAsT’s bag-of-words retrieval and neural ranking methods. We refer the reader to CAsT’21 and CAsT’22 overview papers for more on the implementation details of these baselines.

For incorporating user feedback, we compare against additional baselines built on top of the organizer-auto baseline. Formally, we model the following method with the function \( u' = \gamma(u'|H) \), described in Section 3.1.2, aimed at updating the system’s understanding of the user’s information need:

- **organizer-auto+RM3** uses the user feedback \( u_f \) after the BART response generation step. Using the RM3 algorithm [26], we expand the reformulated query \( u' \) with up to 10 terms from the feedback utterance \( u_f \). This expanded query is fed through the BM25 and MonoT5 steps, followed by BART response generation. For our experiments, we interpret the number of feedback rounds as a proxy for user patience, detailed in Section 3.2, i.e., the more rounds of feedback a user is willing to give, the more patient they are.

- **organizer-auto+Rocchio** follows the same setup as organizer-auto+RM3 but uses the Rocchio algorithm [43] for processing explicit feedback: \( u'_q = u_q + \text{Rocchio}(u_f) \). To assess if feedback utilization works on other systems, we also evaluate three of the strongest automatic submissions to CAsT’22, including 

- splade_15mm_ens, 
- ais_sparsebool, and 
- UWCCano22.

We obtain the run files of these systems from the CAsT’22 organizers.

### 4.3 Utilizing feedback

We implement query rewriting and passage ranking methods to utilize feedback by adapting state-of-the-art systems as follows:

**Passage Ranking.** We modify the query input of the MonoT5 re-ranker by adding feedback text to it, while keeping the passage input as is. Specifically, we format the input to MonoT5 as follows:

- Query \([u_q, u_f] \) [u_r]

**Query rewriting.** We use the baseline T5 query rewriter (T5-CQR) to reformulate the feedback utterance based on conversation context (including the user’s raw query). We observe that this makes the rewriter prone to ‘over-rewriting’, especially in the case of positive feedback. For example, ‘Thanks!’ may be rewritten to ‘What types of essential oils should I consider for a scented lotion?’, essentially repeating the user’s query, even after a positive feedback from the user. Given the lack of discourse-aware query rewriters, we examine the effects of mitigating this by also implementing an improved version of the rewriter that only reformulates negative feedback (Discourse-CQR). In both cases, as with the baseline system, the input text is automatically truncated where it exceeds the model’s limit of 512 tokens.

Additionally, we process the answers to clarifying questions following Aliannejadi et al. [5]. Specifically, we append the answer and the asked clarifying question to the initial query: \( u'_q = u'_q + u'_c \). The reformulated utterance is then fed through our baseline pipeline organizer-auto, without the first step of query rewriting.

#### 4.3.3 Asking clarifying questions

We implement several established approaches to asking clarifying questions. While we acknowledge that not all utterances require clarification, as indicated by the \( \text{clarification \_need} \) variable described in Section 3.1.1, we do not explicitly model it. The clarifying question is thus either not asked at all (\( \text{clarification \_need} = 0 \)) or asked at each turn (\( \text{clarification \_need} = 1 \)), depending on the experiment. We focus on both question selection and question generation, implementing the following baselines.

**Question selection.** As detailed in Section 3.1.1, the aim of this group of models is to select an appropriate clarifying question \( s'_{cq} \) given the user’s current utterance \( u'_q \). Therefore, we opt for two ranking-based methods. First, a BM25-based method, termed SelectCQ-BM25, which indexes the clarifying question pool \( CQ \) and performs retrieval with reformulated user utterance \( u'_q \), specifically: \( s'_{cq} = \arg \max_i (BM25(cq|u'_q)) \), \( cq \in CQ \). A similar approach has been taken in previous works [3, 5]. Second, a semantic matching-based method, termed SelectCQ-MPNet, utilizing MPNet [49] to predict a score for each question \( cq \) from the pool: \( s'_{cq} = \arg \max_i (MPNet(cq, u'_q)) \), \( cq \in CQ \). A similar approach has been adapted for CAsT’22 [25]. In both cases, the clarifying question with the highest score is selected, as indicated by the \( \text{arg max} \) function.

**Question generation.** We implement entity- and template-based clarifying question generation method, dubbed GenerateQC-Entity. Template-based question generation has been widely utilized in the research community due to its simplicity and effectiveness [47, 59, 63]. With entities being central to the topic of a document, we opt to utilize SWAT [37] to extract salient entities to generate clarifying questions. Specifically, we extract entities above a certain threshold (\( \rho > 0.35 \), as recommended by the authors) from the top \( n \) results in the ranked list. We then sort the entities by their salience score in descending order, resulting in a list of entities \( E = \{e_1, e_2, \ldots, e_M\} \). Finally, the question is constructed by inserting up to \( m \) entities (\( m \) is set to 3) to the question template “Are you interested in \( e_1 \), \( e_2 \) or \( e_3 \)?” Note that we alter the template according to the number of entities, in case \( E \) contains less than 3 entities.
4.4 Evaluation

4.4.1 Mixed-initiative search systems. We use the official measures and methodology from the CAsT benchmark for comparison. We report macro-averaged retrieval effectiveness of all systems at the turn level. We report NDCG@3 to focus on precision at the top ranks as well as standard IR evaluation measures (MAP, MRR, NDCG) to a depth of 1000 and at a relevance threshold of 2 for binary measures. Statistical significance is reported under the two-tailed $t$-test with the null hypothesis of equal performance. We reject the null hypothesis in favor of the alternative with $p$-value < 0.05. We design the experimental framework with the goal of assessing the impact of various CS system components on retrieval performance. Specifically, we evaluate the base pipeline, described in Section 4.1 for passage retrieval with and without CS system components.

4.4.2 Naturalness and usefulness of generated answers. We evaluate ConvSim in terms of naturalness and usefulness, as described in Section 3.2.1. To this end, we compare our method to the current state-of-the-art simulator for answering clarifying questions, USi [48], as well as human-generated responses. Following [48], we conduct a crowdsourcing-based evaluation on the ClariQ dataset. Specifically, two crowd workers annotate a pair of answers, where one is generated by ConvSim, and the other by USi or humans. We instruct them to evaluate the answers in terms of naturalness and usefulness. In this pairwise setting, we count a win for a method if both crowd workers vote that the method’s answer is more natural (or useful), while if the two crowd workers do not agree, we count it a tie. For multi-turn evaluation, we utilize a multi-turn extension of the ClariQ dataset with human-generated multi-turn conversations. We follow Li et al. [27] and present full conversations for comparisons. We report statistical significance under the trinomial test [11], an alternative to the binomial and Sign tests that takes into account ties. The null hypothesis of equal performance is rejected in favor of the alternative with $p$-value < 0.05. We present the results for both single- and multi-turn assessments.

We use the Amazon Mechanical Turk platform for our crowdsourcing-based experiments. We take several steps to ensure high-quality annotations: (i) we select workers based in the United States, in order to mitigate potential language barriers; (ii) the selected workers have above 95% lifetime approval rate and at least 5000 approved HITs; (iii) we reject workers with wrong annotations on manually constructed test set; (iv) we provide fair compensation of $0.25 per HIT, which with an average completion time of about 30 seconds, more than 300% of the minimum wage in the U.S.

4.4.3 Feedback. We evaluate the feedback generation capabilities of ConvSim as described in Section 3.2.2. To this end, we generate responses for each turn in the CAsT22 dataset with the Organizer-auto method, described in Section 4.3.1. Next, we utilize ConvSim to give feedback to the generated responses and manually annotate whether the generated feedback is positive or negative. We consider feedback positive if it is along the lines of “Thank you, that was helpful,” and negative if similar to “That’s not what I asked for.” We consider it as negative feedback if it includes a more detailed sub-question aimed at eliciting the missing component (e.g., “Thanks, but what is its impact on climate change in developing countries?”), since the information need is not entirely satisfied. We compare the system’s responses to the canonical responses present in CAsT to assess whether the information need is satisfied or not.

5 RESULTS

In this section we present the empirical evaluation with three core research questions:

RQ1 How can we leverage user feedback and what is its effect on core components of a conversational search pipeline including: explicit relevance feedback processing, ranking and generating clarifying questions, and core ranking?

RQ2 How does the ConvSim model compare with existing approaches for multi-turn simulation in terms of naturalness and usefulness?

RQ3 What is the effect of multiple rounds of simulated feedback when used in ranking?

5.1 Mixed-initiative systems

Tables 1 and 2 list the retrieval results for query reformulation and passage ranking, respectively. Generally, the results demonstrate improvements of feedback-aware methods over the baselines. Below, we discuss the findings in detail.

5.1.1 Query rewriting with feedback. Compared to the baseline system, the addition of the QuReTeC results in a 39% decrease in nDCG@3. This is surprising, considering QuReTeC’s strong performance on previous editions of the CAsT benchmark. Likewise, Rocchio also leads to a decrease in performance, with nDCG@3 going down by 0.151 points (41%). In contrast, the addition of RMI3 improves performance compared to the baseline, significantly outperforming it in terms of Recall, MAP, nDCG, and nDCG@3. Moreover, the results show the Discourse-CQR method to outperform the baseline across all metrics, demonstrating the strongest performance among the implemented methods.

Expectedly, high-quality query rewriting/refORMulation with feedback enables systems to retrieve more relevant passages in the initial retrieval stage at each turn, as evidenced by the increase in recall for the RMI3 and Discourse-CQR methods over the baseline. Not all reformulation methods are effective in all cases, however. Consider a turn where a user provides the following negative feedback without clarification: “That’s not what I asked for. Can you please answer my question?” Term expansion methods based on explicit feedback alone, such as RMI3 and Rocchio, completely fail, given the lack of relevant terms in the feedback utterance. On the other hand, methods that rely on explicit feedback and conversational history stand a better chance, as they have access to more relevant context to arrive at a better expression of the under-specified query.

We note that, without fine-tuning, T5-CQR performs competitively as a feedback rewriter, but still underperforms RMI3 due to the ‘over-rewriting’ issues discussed in Section 4.3.1. When we account for this with the Discourse-CQR method, we observe boosts across all metrics. This suggests that naively using current models and systems to exploit explicit feedback through query rewriting are failure-prone. As a result, future ‘feedback-aware’ conversational query rewriters need to take the feedback type into consideration, in order to be effective.
Table 1: Retrieval performance of methods for query reformulation using explicit feedback. Sign † indicates a significant difference compared to the organizer-auto baseline.

<table>
<thead>
<tr>
<th>Method</th>
<th>R</th>
<th>MAP</th>
<th>MRR</th>
<th>nDCG</th>
<th>nDCG@3</th>
</tr>
</thead>
<tbody>
<tr>
<td>organizer-auto</td>
<td>0.348</td>
<td>0.155</td>
<td>0.533</td>
<td>0.311</td>
<td>0.365</td>
</tr>
<tr>
<td>+ QuReTiC</td>
<td>0.192</td>
<td>0.088</td>
<td>0.310</td>
<td>0.180</td>
<td>0.223</td>
</tr>
<tr>
<td>+ Rocchio</td>
<td>0.195</td>
<td>0.086</td>
<td>0.316</td>
<td>0.174</td>
<td>0.214</td>
</tr>
<tr>
<td>+ T5-CQR</td>
<td>0.340</td>
<td>0.131</td>
<td>0.500</td>
<td>0.288</td>
<td>0.329</td>
</tr>
<tr>
<td>+ RM3</td>
<td>0.384†</td>
<td>0.167†</td>
<td>0.565</td>
<td>0.343†</td>
<td>0.398†</td>
</tr>
<tr>
<td>+ Discourse-CQR</td>
<td>0.384†</td>
<td>0.174†</td>
<td>0.620†</td>
<td>0.348†</td>
<td>0.423†</td>
</tr>
</tbody>
</table>

5.1.2 Passage ranking with feedback. Across the board, we note that passage ranking with feedback leads to additional performance gains when used in a multi-step reranking setup. Specifically, the use of FeedbackMonoT5 on top of selected participant submissions to TREC CAsT’22 leads to boosts in nDCG@3, nDCG, and MRR scores at various reranking thresholds. Although we only report the results of ranking the top 100 passages in Table 2, we observe similar trends when reranking at depth 10 and 50, and expect that these observations continue beyond the depth of 100. We further note that the magnitude of the improvement explicit feedback brings for retrieval varies between these participant systems, indicating that the effectiveness of explicit feedback may depend on the underlying characteristics of each system.

We note that the addition of FeedbackMonoT5 leads to an average 6% gain in nDCG@3. These results are consistent for the MRR metric too as FeedbackMonoT5 provides an average 7% gain. Showing that explicit feedback can be useful in improving the overall retrieval. This is not just due to the quality of the MonoT5 passage ranker but is a result of the additional context from explicit feedback.

We delve deeper into the queries where the delta in nDCG@3 before and after feedback ranking is at least 0.5 points in the splade_15mm_ens run. We observe that passage ranking with feedback hurts performance in cases of positive feedback (“Thanks,”) and negative feedback without clarification (“Can you please answer my question?”), whereas negative feedback with clarification boosts performance (“That’s interesting, but what makes the beef so special?”). Feedback that introduces more explicit context is more useful. As with query rewriting, this phenomenon suggests that ranking models should be feedback aware.

5.1.3 Clarification and answer processing. Table 3 shows performance of three clarifying question construction methods, described in Section 4.3.3. We observe an overall increase in effectiveness across all methods, with SelectCQ-BM25 and SelectCQ-MPNet significantly outperforming the baseline across several metrics. Most gains in performance are in recall, as the original query is expanded by the answer and clarifying question providing additional information to the initial retriever. GenerateCQ-Entity does not perform as well as selection-based methods. We attribute this finding to potentially off-topic clarifying questions, as the entities extracted were not necessarily geared towards elucidating user’s need. ConvSim might have responded along the lines of “I don’t know,” or “No thanks,” thus not helping elucidate the underlying information need.

Table 2: Retrieval performance of passage ranking using explicit feedback on top of selected CAsT participant systems. This reranking step only reranks the first 100 passages from each system.

<table>
<thead>
<tr>
<th>System</th>
<th>MAP</th>
<th>MRR</th>
<th>nDCG</th>
<th>nDCG@3</th>
</tr>
</thead>
<tbody>
<tr>
<td>organizer-auto</td>
<td>0.155</td>
<td>0.533</td>
<td>0.311</td>
<td>0.365</td>
</tr>
<tr>
<td>+ MonoT5</td>
<td>0.093</td>
<td>0.315</td>
<td>0.257</td>
<td>0.189</td>
</tr>
<tr>
<td>+ FeedbackMonoT5</td>
<td>0.152</td>
<td>0.560</td>
<td>0.313</td>
<td>0.387</td>
</tr>
<tr>
<td>splade_15mm_ens</td>
<td>0.217</td>
<td>0.585</td>
<td>0.479</td>
<td>0.411</td>
</tr>
<tr>
<td>+ MonoT5</td>
<td>0.221</td>
<td>0.614</td>
<td>0.484</td>
<td>0.417</td>
</tr>
<tr>
<td>+ FeedbackMonoT5</td>
<td>0.226</td>
<td>0.632</td>
<td>0.489</td>
<td>0.442</td>
</tr>
<tr>
<td>uiis_sparseboat</td>
<td>0.187</td>
<td>0.559</td>
<td>0.407</td>
<td>0.383</td>
</tr>
<tr>
<td>+ MonoT5</td>
<td>0.177</td>
<td>0.581</td>
<td>0.399</td>
<td>0.381</td>
</tr>
<tr>
<td>+ FeedbackMonoT5</td>
<td>0.184</td>
<td>0.611</td>
<td>0.408</td>
<td>0.415</td>
</tr>
<tr>
<td>UWCcano22</td>
<td>0.213</td>
<td>0.617</td>
<td>0.441</td>
<td>0.438</td>
</tr>
<tr>
<td>+ MonoT5</td>
<td>0.217</td>
<td>0.612</td>
<td>0.443</td>
<td>0.427</td>
</tr>
<tr>
<td>+ FeedbackMonoT5</td>
<td>0.217</td>
<td>0.659</td>
<td>0.454</td>
<td>0.454</td>
</tr>
</tbody>
</table>

Table 3: Performance after asking a clarifying question constructed by various methods, compared to the baseline.

<table>
<thead>
<tr>
<th>Method</th>
<th>R</th>
<th>MAP</th>
<th>MRR</th>
<th>nDCG</th>
<th>nDCG@3</th>
</tr>
</thead>
<tbody>
<tr>
<td>organizer-auto</td>
<td>0.348</td>
<td>0.154</td>
<td>0.532</td>
<td>0.311</td>
<td>0.365</td>
</tr>
<tr>
<td>+ SelectCQ-BM25</td>
<td>0.433†</td>
<td>0.166</td>
<td>0.625</td>
<td>0.364†</td>
<td>0.411</td>
</tr>
<tr>
<td>+ SelectCQ-MPNet</td>
<td>0.431†</td>
<td>0.173†</td>
<td>0.631</td>
<td>0.362</td>
<td>0.409</td>
</tr>
<tr>
<td>+ GenerateCQ-Entity</td>
<td>0.409</td>
<td>0.162</td>
<td>0.577</td>
<td>0.348</td>
<td>0.398</td>
</tr>
</tbody>
</table>

5.2 User simulator

5.2.1 Single- and multi-turn clarifying question answering. Table 4 presents the results in comparison to USi [48] and human-generated answers to clarifying questions in single- and multi-turn scenarios. We make several observations from the results. First, ConvSim significantly outperforms USi both in terms of naturalness and usefulness in both single- and multi-turn settings. Second, the difference between the performance of ConvSim and USi is especially evident in the multi-turn setting, which is one of USi’s potential limitations indicated by the authors [48]. The difference is even greater in multi-turn usefulness assessments, which can be attributed to USi’s hallucinations, and thus not staying on topic. Finally, ConvSim in most cases does not significantly outperform human-generated answers, except in single-turn usefulness. Although further analysis is required, we suspect the difference to have come from ConvSim’s precision in answering clarifying questions, while crowd workers sometimes answer them reluctantly and concisely, with no notion of grammar and punctuality (e.g., “no”). The results indicate that ConvSim can be used to answer clarifying questions both in single- and multi-turn settings, outperforming state-of-the-art methods both in terms of naturalness and usefulness.

5.2.2 Generated feedback evaluation. Table 5 shows the performances of Organizer-auto model on CAsT’22 queries broken down by whether feedback given to the system’s response is positive or negative, as described in Section 3.2.2. Results show significant differences between responses with positive and negative feedback. Feedback on the system’s responses generated by ConvSim is useful, as the responses receiving negative feedback correspond to the poor retrieval effectiveness. On the contrary, when the system’s response
Table 4: Results of crowdsourcing study assessing naturalness and usefulness of generated answers to clarifying questions in single- and multi-turn scenarios. Each value indicates the percentage of pairwise comparisons won by the specific model as well as ties. Sign † indicates a significant difference.

<table>
<thead>
<tr>
<th>Model</th>
<th>Naturalness</th>
<th>Usefulness</th>
<th>Ties</th>
<th>ConvSim</th>
<th>USi</th>
<th>Ties</th>
<th>ConvSim</th>
<th>Human</th>
<th>Ties</th>
</tr>
</thead>
<tbody>
<tr>
<td>organizer-auto</td>
<td>37%</td>
<td>44%</td>
<td>36%†</td>
<td>25%</td>
<td>3%</td>
<td>20%†</td>
<td>44%</td>
<td>47%</td>
<td>58%</td>
</tr>
<tr>
<td>splade_t5mm_ens</td>
<td>35%</td>
<td>37%</td>
<td>36%†</td>
<td>25%</td>
<td>3%</td>
<td>20%†</td>
<td>44%</td>
<td>47%</td>
<td>58%</td>
</tr>
<tr>
<td>organizer-manual</td>
<td>45%</td>
<td>58%</td>
<td>0%</td>
<td>6%</td>
<td>5%</td>
<td>0%</td>
<td>6%</td>
<td>5%</td>
<td>0%</td>
</tr>
</tbody>
</table>

Table 5: Performance on turns where feedback is negative vs. turns where feedback is positive. The “Perc.” column indicates the percentage of such turns in the CAsT’22 dataset. All the differences are significant.

<table>
<thead>
<tr>
<th>Feedback</th>
<th>Perc.</th>
<th>R</th>
<th>MAP</th>
<th>MRR</th>
<th>nDCG</th>
<th>nDCG@3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Negative</td>
<td>49%</td>
<td>0.073</td>
<td>0.039</td>
<td>0.399</td>
<td>0.091</td>
<td>0.161</td>
</tr>
<tr>
<td>Positive</td>
<td>51%</td>
<td>0.185</td>
<td>0.128</td>
<td>0.739</td>
<td>0.239</td>
<td>0.449</td>
</tr>
</tbody>
</table>

Table 6: Passage ranking using explicit feedback on top of select CAsT participant runs. Runs are evaluated on a subset of queries annotated to require initiative.

<table>
<thead>
<tr>
<th>System</th>
<th>MAP</th>
<th>MRR</th>
<th>nDCG</th>
<th>nDCG@3</th>
</tr>
</thead>
<tbody>
<tr>
<td>organizer-auto</td>
<td>0.091</td>
<td>0.567</td>
<td>0.251</td>
<td>0.392</td>
</tr>
<tr>
<td>+ FeedbackMonoT5</td>
<td>0.101</td>
<td>0.589</td>
<td>0.256</td>
<td>0.404</td>
</tr>
<tr>
<td>splade_t5mm_ens</td>
<td>0.168</td>
<td>0.597</td>
<td>0.444</td>
<td>0.424</td>
</tr>
<tr>
<td>+ FeedbackMonoT5</td>
<td>0.195</td>
<td>0.659</td>
<td>0.463</td>
<td>0.492</td>
</tr>
<tr>
<td>uis_sparseboat</td>
<td>0.137</td>
<td>0.739</td>
<td>0.381</td>
<td>0.445</td>
</tr>
<tr>
<td>+ FeedbackMonoT5</td>
<td>0.133</td>
<td>0.733</td>
<td>0.378</td>
<td>0.490</td>
</tr>
<tr>
<td>UWCcano22</td>
<td>0.145</td>
<td>0.661</td>
<td>0.388</td>
<td>0.386</td>
</tr>
<tr>
<td>+ FeedbackMonoT5</td>
<td>0.150</td>
<td>0.610</td>
<td>0.395</td>
<td>0.393</td>
</tr>
</tbody>
</table>

satisfies the given information need, as demonstrated by higher retrieval performance, the simulator’s feedback is positive. ConvSim is not aware of the system’s retrieval effectiveness and provides feedback solely on the generated response and IN description.

6 DISCUSSION AND ANALYSIS

Does feedback help where it matters? Section 5 shows that systems that leverage feedback outperform systems that do not use it. We investigate a subset of 24 queries that require initiative as annotated by organizers [33]. These turns require additional user input and are typically open-ended or a branching point. Systems that exploit user input should perform better on these queries than systems that do not. Table 6 shows results of feedback passage ranking method on top of the participant runs introduced in table 6. Using feedback ranking FeedbackMonoT5 leads to non-significant improvements across most metrics for all runs with an average increase of 7.75% in nDCG@3 with other metrics being similar.

Effect of iterative feedback. We investigate the potential for multiple rounds of feedback in a simulated environment. We run the organizer-auto-Discourse-CQR system with FeedbackMonoT5 passage ranker for 10 rounds of feedback. For efficiency we only apply re-ranking to the first 100 passages retrieved. Figure 2 shows consistent improvements in terms nDCG@3 over the organizer-auto system with + FeedbackMonoT5 (round 0) baseline, with slight dips and plateaus between rounds 3 to 5 and rounds 6 to 8. At rounds 6 and above both MRR and nDCG@3 of this system exceed those of the organizer-manual system. Recall and MAP at round 8 come within 0.004 and 0.003 points of the manual run, respectively, further highlighting the utility of explicit feedback. Prompting the user for up to 8 or more rounds of feedback is not realistic and motivates the need for more effective feedback models that can learn from fewer rounds of feedback.

Combining clarification and explicit feedback. We analyze the effectiveness of FeedbackMonoT5 for processing answers to questions selected with SelectCQ-BM25. The results suggest an improvement over the organizer-auto baseline (nDCG@3 = 0.392; +47% relative improvement), suggesting that FeedbackMonoT5 can be used for processing answers to clarifying questions. We experiment with a round of clarification and a round of feedback and observe significant boost in Recall (0.448; +29% vs the baseline), but a relatively low improvement in terms of nDCG@3 (0.389; +6%). We hypothesize that both rounds of feedback result in well-defined information need, thus boosting the Recall, but query reformulation methods (i.e., FeedbackMonoT5) fail to resolve the complex context, leading to poor re-ranking performance.

7 CONCLUSIONS

We study the effectiveness of mixed-initiative conversational search models in combination with simulated user feedback. Specifically, we compare and extend proven models with an aim of incorporating user feedback, including answers to clarifying questions and explicit feedback on system’s responses. We propose a new user simulator, ConvSim, capable of multi-turn interaction, leveraging LLMs. The results show utilizing feedback consistently improves retrieval across the majority of the methods, resulting in +16% improvement in nDCG@3 after a single turn of feedback. Moreover, we show that several rounds of feedback result in even greater boost (+35% after three rounds). This promises potential for advancements in CS and calls for further work on feedback processing methods.

8 ACKNOWLEDGMENTS

This work is supported by the Engineering and Physical Sciences Research Council (EPSRC) grant EP/V025708/1 and a 2019 Google Research Award.
REFERENCES


