Retrieval-Augmented Conversational Recommendation with
Prompt-based Semi-Structured Natural Language State Tracking

University of Toronto
Toronto, Ontario, Canada

1 INTRODUCTION

An effective conversational recommendation (ConvRec) system needs to understand rich and diverse natural language (NL) expressions of user preferences and intents, which are often communicated in a subtle or indirect manner (e.g., “I’m watching my weight”). Such complex language requests pose significant challenges to retrieving relevant items, especially if only relying on often incomplete or out-of-date metadata. Fortunately, many recommendation domains have an abundance of rich NL item reviews that cover standard metadata categories as well as complex opinions and narratives that might match a user’s interests (e.g., “classy joint for a date”). However, only recently has the advent of large language models (LLMs) provided us with the ability to unlock the commonsense reasoning connections between rich user preference utterances and complex language in user-generated content such as NL reviews. Further, LLMs enable novel paradigms for semi-structured dialogue state tracking, complex intent and preference understanding, and generating recommendations, explanations, and question answers. Motivated by these possibilities, we introduce a novel demonstration technology RA-Rec, a Retrieval-Augmented, LLM-driven dialogue state tracking system for ConvRec, showcased on restaurant recommendation using a video,1 well-documented open source GitHub repository,2 and interactive Google Colab notebook.3

CCS CONCEPTS
• Information systems → Recommender systems; Personalization; Language models.

KEYWORDS
Conversational Recommendation, LLM, Dialogue State Tracking

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

ACM Reference Format:

1 These authors contributed equally to this research.

1 https://www.youtube.com/watch?v=WUY56U2L6TU
2 https://github.com/D3Mlab/llm-convrec
3 https://apoj.short.gy/dm-llm-convrec-demo
Aiming to unlock the expressive NL data available in reviews, Abdol-lah Pour et al. [1] recently extended Neural IR [22] to an approach they call Reviewed Item Retrieval (RIR), where the key challenge lies in fusing low-level information from multiple reviews to a higher item level [27]. They demonstrate it is more effective to first score individual reviews against a query and then aggregate these scores to an item level (late fusion), instead of summarizing reviews at an item level before query-scoring (early fusion), since late fusion retains critical nuanced review information lost by early fusion.

In late fusion RIR, given a query and a set of reviews, a neural encoder maps each review and the query to respective embeddings. A similarity function, such as the dot product, then computes a query-review similarity score. For each item, scores from the reviews with the highest query-review similarities are then averaged (fused) to give a query-item similarity score, and the top-scoring items are returned. As we will discuss in the next section, our RA-Rec system adapts late fusion RIR to ConvRec by generating queries from a NL dialogue state and using review-based retrieval-augmented generation for recommendation and QA, as illustrated in Figure 2.
We employ a prompt-driven approach for intent classification and state updating, with the latter relying on a JSON format NL state that can be configured with domain-specific keys while capturing nuance through LLM-generated NL values. We then use this NL state to facilitate personalized, retrieval-augmented recommendation and QA utilizing item reviews and metadata.

### 3.1 Prompt-Driven Intent Classification

After the user makes an utterance, LLM-prompting is used to determine whether the user expresses any of the four intents in Table 1, which are a subset of the recommendation dialogue intent taxonomy of Lyu et al. [18]. Table 3 outlines the prompts used in RA-Rec, with full prompt templates available in the GitHub repository (see Sec. 1). We take a multi-label intent classification approach to capture multiple intents that might be expressed in a single utterance — for example, the utterance “Does Washoku Bistro have parking?” should be classified using both the intents “Inquire” and “Provide preference” because it expresses a preference towards available parking. A larger set of user intents can be facilitated by updating the system’s prompts and initial state keys.

### 3.2 Semi-Structured NL Dialogue State Tracking

We store descriptions of user preferences and other important conversational elements such as rejected recommendations in a JSON state using the keys shown in Table 2 — two state examples are in Figures 1 and 2. While the keys provide structure, the state values are typically LLM-generated from the latest utterances, allowing the state to represent complex NL expressions of preference such as “I’m watching my weight” at a level of nuance and expressivity that would be impossible with predefined value sets. We thus refer to this state representation as a semi-structured NL dialogue state.

#### 3.2.1 State Elements

Since the goal of RA-Rec is recommendation, the most important components of the state maintain an up-to-date belief about user preferences, represented through hard (required) and soft (not required) constraints. In our restaurant recommendation demo, these constraints are represented with several domain-specific subkeys listed in Table 2, as well as an “others” subkey to capture any unspecified preference types. To adapt RA-Rec to a new domain, these restaurant-specific subkeys can be replaced with domain-specific subkeys with little effort from a system designer.

Other state elements include previously recommended, rejected, or accepted restaurants; more elements could be easily added to handle a wider set of (domain-specific) user intents and system actions. Most state values are LLM-generated (prompts are summarized in Table 3) and used downstream for action selection, recommendation, explanation, and QA.

### 3.3 Action Selection

The main system actions, summarized in Table 1 are Request Information, Recommend and Explain, and Answer. To understand our
Table 3: The main prompts used in RA-Rec – full templates can be found in the repository documentation (see Sec. 1 link).

<table>
<thead>
<tr>
<th>Component</th>
<th>Prompt</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>State Update</td>
<td>Classify Intent</td>
<td>Given a user utterance and description of an intent (e.g. inquire), identify whether the utterance expresses the intent</td>
</tr>
<tr>
<td></td>
<td>Update Constraints</td>
<td>Given a user utterance and the previous hard and soft constraints, update the constraints</td>
</tr>
<tr>
<td></td>
<td>Update Accepted/Rejected Item</td>
<td>Given a user utterance with intent “Accept/Reject Recommendation”, identify which item was accepted/rejected.</td>
</tr>
<tr>
<td>Recommendation and Explanation</td>
<td>Generate Recommendation Query</td>
<td>Given the top retrieved items, their metadata, and their top reviews, explain how these recommended items match the hard/soft constraints.</td>
</tr>
<tr>
<td></td>
<td>Explain Recommendations</td>
<td>Given an inquiry and relevant metadata entries, generate an answer.</td>
</tr>
<tr>
<td>QA</td>
<td>Determine QA Knowledge Source</td>
<td>Given a user inquiry about recommended items and those items’ metadata, identify which fields should be used to answer the inquiry, if any. If none, reviews will be used as the QA knowledge source.</td>
</tr>
<tr>
<td></td>
<td>Answer Using Metadata</td>
<td>Given an inquiry and relevant metadata entries, generate an answer.</td>
</tr>
<tr>
<td></td>
<td>Answer Using Reviews</td>
<td>Given an inquiry and retrieved reviews, generate an answer.</td>
</tr>
</tbody>
</table>

Request Information implementation, consider a user asking for a restaurant recommendation without giving a location preference – a recommendation may yield a restaurant in the wrong city! To avoid such premature recommendations with insufficient context, we identify mandatory preferences that the system must ask before recommending if not already provided by the user. In our demo, mandatory preferences are location and cuisine, type as shown in Table 2, but this selection is easily customized. Once mandatory preferences have been provided, the system will Answer if the user has made an inquiry and Recommend and Explain otherwise.

3.4 Retrieval-Augmented Recommendation and Explanation

To leverage expressive user review content in RA-Rec, we provide a novel adaptation of retrieval-augmented generation [16] for late fusion recommendation and explanation. To do this, we first generate a query based on semi-structured preferences in the dialogue state and then retrieve relevant items using RIR (Sec. 2.2) over both the item reviews and known metadata. This process is illustrated in Figure 2 with relevant prompts summarized in Table 3.

Specifically, after a NL query is generated from the hard and soft constraints in the state, we implement late fusion RIR to retrieve a list of top-k scoring items. Our implementation of late fusion RIR uses a TAS-B dense encoder [11] (a variant of BERT [9] fine-tuned for retrieval), dot product similarity, and approximate maximum-integer product search (MIPS) via FAISS [10] to enable scalability to large review corpora. After the top-k items (k = 2 in our demo) are retrieved, we use the metadata and top-scoring reviews for each item to generate a recommendation and LLM prompt-driven explanation of how these items match the dialogue state preferences.

3.5 Retrieval-Augmented Question Answering

As observed by Lyu et al. [18], the later stages of a recommendation conversational often involve a number of inquiries about the recommended item to confirm that it meets the user’s requirements. To address such QA, RA-Rec retrieves relevant reviews or metadata for each of the items in question and uses this retrieved information to generate an answer – with Table 3 outlining the prompts used in our QA approach. Our framework is capable of addressing both individual item questions such as "What kind of menu do they offer?" as well as comparative questions such as "How do their prices compare?" as demonstrated in the video (see Sec. 1). In more detail, the first step of QA uses prompting to determine whether an inquiry can be answered using available metadata, which is typically the best knowledge source for simple questions about common properties. In our restaurant recommendation demo, such common metadata fields include price, delivery availability, and parking information. If the inquiry cannot be answered with metadata, a NL query is generated from the user utterance and used to retrieve several reviews for each item in question. As discussed above, reviews are an expressive knowledge source, especially when inquiries and preferences are stated in complex ways. Finally, the retrieved reviews and metadata for each item are used to generate an answer to the question, which may include item comparisons.

3.6 RA-Rec System Summary

In summary, RA-Rec employs an LLM-driven, modular DST structure to facilitate a controllable recommendation dialogue that can connect complex NL user preferences to matching items using their reviews and metadata. Its JSON semi-structured NL state features configurable keys for domain-specific control while the LLM-updated state values are able to express NL nuance. This state supports novel retrieval-augmented recommendation, explanation, and QA, using scalable retrieval methods such as late fusion RIR and leveraging item reviews and metadata to generate responses.

4 DEMONSTRATION DETAILS

Our system is designed for easy adaptation to various domains, and as a demonstration, we present RA-Rec for restaurant recommendation – see Sec. 1 for demo links. Specifically, we use the Yelp Academic Dataset to obtain metadata and over 46K reviews for 1298 restaurants in Edmonton, Alberta. GPT-3.5-turbo is the LLM used for all prompting steps, but the RA-Rec framework is LLM-agnostic and will work with any prompt-based LLM model.

5 FUTURE WORK

RA-Rec is a flexible LLM prompt-driven architecture and thus opens many new directions for ConvRec systems to support natural user workflows [13, 18]. Key extensions include support for (1) active preference elicitation to narrow down large item spaces, (2) automatic generation of disambiguation and clarification questions, and (3) trade-off negotiation between multiple recommendations.

4 https://www.yelp.com/dataset/download 5 The median number of reviews per restaurant was 21.