Unleashing the Retrieval Potential of Large Language Models in Conversational Recommender Systems

Ting Yang
Department of Computer Science
Hong Kong Baptist University
Hong Kong, China
cstyang@comp.hkbu.edu.hk

Li Chen
Department of Computer Science
Hong Kong Baptist University
Hong Kong, China
lichen@comp.hkbu.edu.hk

ABSTRACT

Conversational recommender systems (CRSs) aim to capture user preferences and provide personalized recommendations through interactive natural language interaction. The recent advent of large language models (LLMs) has revolutionized human engagement in natural conversation, driven by their extensive world knowledge and remarkable natural language understanding and generation capabilities. However, introducing LLMs into CRSs presents new technical challenges. Directly prompting LLMs for recommendation generation requires understanding a large and evolving item corpus, as well as grounding the generated recommendations in the real item space. On the other hand, generating recommendations based on external recommendation engines or directly integrating their suggestions into responses may constrain the overall performance of LLMs, since these engines generally have inferior representation abilities compared to LLMs. To address these challenges, we propose an end-to-end large-scale CRS model, named as **ReFICR**, a novel LLM-enhanced conversational recommender that empowers a retrievable large language model to perform conversational recommendation by following retrieval and generation instructions through lightweight tuning. By decomposing the complex CRS task into multiple subtasks, we formulate these subtasks into two types of instruction formats: *retrieval* and *generation*. The hidden states of ReFICR are utilized for generating text embeddings for retrieval, and simultaneously ReFICR is fine-tuned to handle generation subtasks. We optimize the contrastive objective to enhance text embeddings for retrieval and jointly fine-tune the large language model objective for generation. Our experimental results on public datasets demonstrate that ReFICR significantly outperforms baselines in terms of recommendation accuracy and response quality. Our code is publicly available at the link: https://github.com/yt556677/ReFICR.

CCS CONCEPTS

• **Information systems** → **Recommender systems**; **Language models**.

RecSys '24, October 14–18, 2024, Bari, Italy
© 2024 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-0505-2/24/10
https://doi.org/10.1145/3640457.3688146

KEYWORDS

Conversational Recommender Systems, Retrievable Large Language Models, Instruction Tuning

ACM Reference Format:
Ting Yang and Li Chen. 2024. Unleashing the Retrieval Potential of Large Language Models in Conversational Recommender Systems. In *18th ACM Conference on Recommender Systems (RecSys '24), October 14–18, 2024, Bari, Italy.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3640457.3688146

1 INTRODUCTION

Traditional recommendation systems rely on users' historical interactions, such as clicks and purchases, to provide recommendations. Considering the changes in user preferences over time, conversational recommender systems (CRSs) allow users to actively seek recommendations, alter their intentions, and provide immediate feedback through natural language conversation with the system. Previous methods [6, 43, 45, 48, 51] have introduced external knowledge to enhance the representation of users' preferences for recommendations and have utilized pre-trained language models (PLMs) for dialogue generation. However, encoding external knowledge may require additional modules, and smaller-scale PLMs may not yield satisfactory generation performance.

Recently, large language models (LLMs) such as ChatGPT [32] and GPT-4 [33] have significantly advanced the learning paradigm for conversational tasks [28]. Because these models are pre-trained on large-scale datasets and possess a considerable number of model parameters, they have demonstrated remarkable capabilities in natural language understanding and response generation. For CRSs, several studies have investigated generating recommendations directly by prompting LLMs to serve as zero-shot conversational recommenders [16, 44]. However, due to the large number of items and the emergence of new items in practical situations, training LLMs to memorize the entire corpus within their parameters for direct recommendation generation is time-consuming and costly. Moreover, LLMs are likely to cause hallucination problems, which refer to the generation of items that deviate from the factual recommendation space. Grounding the generated recommendations in the real item space hence remains a challenge [12].

To address these challenges, some studies have attempted to integrate LLMs with external recommendation engines (such as the traditional CRS models or embedded retrieval models) to make recommendations [10–12, 14]. In such CRSs, the LLMs may function as a controller, managing the dialogue workflow and performing appropriate actions at each turn. Although this approach may avoid hallucinated recommendations of zero-shot CRSs, the utilization

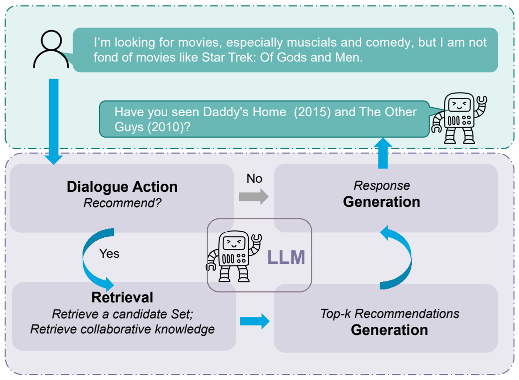
I'm looking for movies, especially musicals and comedy, but I am not fond of movies like Star Trek: Of Gods and Men.
Have you seen Daddy's Home (2015) and The Other Guys (2010)?
Dialogue Action
Recommend?
No
Response
Generation
Yes
LLM
Retrieval
Retrieve a candidate Set;
Retrieve collaborative knowledge
Top-k Recommendations
Generation

Figure 1: Workflow of our proposed ReFICR.

of an external recommendation engine may lead to inferior recommendation performance because it typically has smaller parameters compared to LLMs and hence may result in inferior representations.

In this paper, we introduce an end-to-end LLM-based CRS model, called **ReFICR** for **Re**trievable large language model **F**ollowing **I**nstructions for **C**onversational **R**ecommendation. ReFICR aims to mitigate the limitations of existing LLM-based CRSs by leveraging the powerful representation and generation capabilities of an LLM, without the need for an external recommendation engine. In particular, we lightly tune a retrievable large language model to execute CRS subtasks, such as *retrieval* and *recommendation generation*, by following retrieval and generation instructions. The text inputs, along with retrieval instructions, are fed into ReFICR to obtain representations. It is also fine-tuned to respond to generation instructions by producing appropriate responses.

The workflow of ReFICR can be seen in Figure 1. At each turn, ReFICR will predict the next dialogue action according to the user's intention. If the action is to provide recommendations, it will retrieve both a candidate set from the item corpus and collaborative knowledge from similar users' conversations for generating the top-k recommendations, as well as a natural language response aligned with the recommendation result. If the next action is not to provide recommendations, the model will directly generate a response to the user's query.

Specifically, for the *retrieval* subtasks, the current user's conversation context, along with the task instructions, can be embedded into query representation by utilizing ReFICR's hidden states as text embeddings. We perform candidate set retrieval and retrieval of similar users' conversations based on the similarity of text embeddings. The *generation* subtasks, including recommendation and response generation, can be formulated as language modeling tasks. ReFICR can generate recommendations by ranking the retrieved candidate set with augmented knowledge, and generate natural language responses aligned with the conversation context. We concretely freeze a backbone model, GRITLM [29], which is a large language model fine-tuned from Mistrial-7B [21] on text embedding tasks and generation tasks, and adopt QLoRA [9] for efficient finetuning. By jointly tuning the model with a contrastive objective to enhance text embeddings for retrieval and a language modeling objective for generation, our ReFICR can achieve desired performance in terms of both recommendation accuracy and response quality.

Our contributions are summarized as follows:

- We present ReFICR, a novel LLM-enhanced conversational recommender that empowers a retrievable large language model to perform retrieval and generation for CRSs.
- We unify the subtasks of an CRS into two forms: *retrieval* and *generation*, which can be seamlessly aligned by LLMs through instruction tuning.
- The experiments on public datasets demonstrate the superior performance of our model against baseline methods, validating the effectiveness of this work.

2 RELATED WORK

2.1 Conversational Recommender Systems

Conversational recommender systems (CRSs) capture user preferences through iterative conversations to provide personalized recommendations [20]. Existing works regard the CRS as the combination of a recommendation component and a conversation component, for which separate modules are built to fulfill each task. For instance, ReDial [24] utilizes an autoencoder-based recommendation module along with a hierarchical recurrent encoder-decoder network [36] to generate responses. For further improving recommendation performance of CRSs, KBRD [6] and KGFS [51] introduce structured knowledge (such as DBpedia [4] and ConceptNet [37]) to enhance the user's preference representation. C^2-CRS [52] and RevCore [27] introduce unstructured knowledge like user reviews through transformers.

To address the discrepancy between separate modules, unified frameworks have been proposed to share useful features and knowledge for the recommendation and conversation tasks within a single model. UniCRS [45] uses knowledge-enhanced prompts to fine-tune DialoGPT [50] to perform both recommendation and response generation. BARCOR [43] is based on pre-trained BART [23] to uniformly fine-tune recommendation and generation tasks, and utilizes a specific knowledge graph for CRSs. MESE [48] uses the items' meta-information to enhance the representation using a pre-trained GPT-2 [34] and an item representation encoder. Although these models are able to unify both tasks in a single model, additional knowledge about the context or items is often required to enhance the user's preference representation. PECRS [35] directly formulates an CRS as natural language processing tasks, applying a parameter-efficient plug-in module based on the frozen pre-trained model GPT-2 [34] to handle the generation and item recommendation tasks.

With the advance in large language models (LLMs), some studies have recently been conducted to use LLMs for CRSs. [16] shows that LLMs, especially closed-source models like ChatGPT [32] and GPT-4 [33], can outperform fine-tuned CRS models in zero-shot setting. LLMCRS [11] and RecLLM [12] employ LLMs to build end-to-end CRSs. More concretely, LLMCRS uses existing CRS models, such as KBRD [6] and KGFS [51], to perform user preference elicitation and recommendation tasks. RecLLM utilizes external retrieval models to obtain the candidate set and employs LLMs to perform

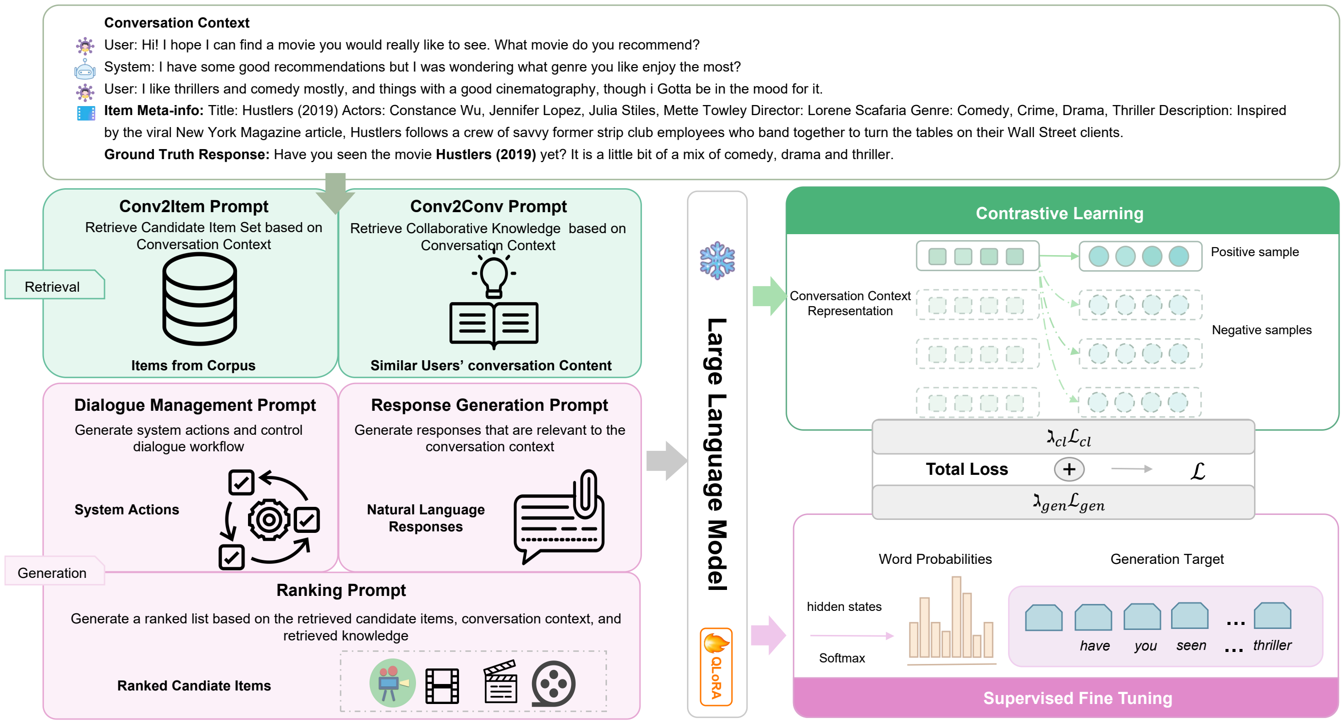
Conversation Context
User: Hi! I hope I can find a movie you would really like to see. What movie do you recommend?
System: I have some good recommendations but I was wondering what genre you like enjoy the most?
User: I like thrillers and comedy mostly, and things with a good cinematography, though i Gotta be in the mood for it.
Item Meta-info: Title: Hustlers (2019) Actors: Constance Wu, Jennifer Lopez, Julia Stiles, Mette Towley Director: Lorene Scafaria Genre: Comedy, Crime, Drama, Thriller Description: Inspired by the viral New York Magazine article, Hustlers follows a crew of savvy former strip club employees who band together to turn the tables on their Wall Street clients.
Ground Truth Response: Have you seen the movie Hustlers (2019) yet? It is a little bit of a mix of comedy, drama and thriller.
Retrieval
Conv2Item Prompt
Retrieve Candidate Item Set based on Conversation Context
Items from Corpus
Conv2Conv Prompt
Retrieve Collaborative Knowledge based on Conversation Context
Similar Users' conversation Content
Generation
Dialogue Management Prompt
Generate system actions and control dialogue workflow
System Actions
Response Generation Prompt
Generate responses that are relevant to the conversation context
Natural Language Responses
Ranking Prompt
Generate a ranked list based on the retrieved candidate items, conversation context, and retrieved knowledge
Ranked Candiate Items
Large Language Model
QLoRA
Contrastive Learning
Positive sample
Conversation Context Representation
Negative samples
$\lambda_{cl}\mathcal{L}_{cl}$
Total Loss
$\mathcal{L}$
$\lambda_{gen}\mathcal{L}_{gen}$
Word Probabilities
Generation Target
hidden states
Softmax
have
you
seen
...
thriller
Supervised Fine Tuning

Figure 2: Overview of our ReFICR. The example conversation is from INSPIRED dataset [15]. The prompts for retrieval subtasks and generation subtasks are fed into the LLM. Contrastive learning is utilized to improve text embeddings for retrieval, and the large language model objective is jointly trained for generation.

preference understanding and generate recommendation explanations. MACRS [10] accomplishes CRS tasks through collaborating with multiple LLM-based agents.

Different from previous LLM-enhanced CRS work, we employ a retrievable large language model to perform retrieval and recommendation generation by following instructions, without relying on external recommendation engines.

2.2 Instruction Tuning

Retrieval with instructions is an emerging area for exploration. Traditionally, models such as OpenAI's embeddings [31] and E5 [41] are trained on unsupervised text pairs and subsequently fine-tuned on small-scale annotated datasets to enhance the quality of text embeddings. However, the performance of these models deteriorates when being applied to new tasks or domains. Therefore, INSTRUCTOR [38] proposes to fine-tune the retrieval model with instructions, enabling the model to encode the text input into different embeddings with varying tasks and goals. In addition, retrieval with instructions can further improve the performance of retrieval models compared to those that only have text [1]. This is because instructions provide additional context and guidance to the retrieval model, helping to clarify ambiguous user queries and understand the specific requirements of the retrieval task. To acquire high-quality instruction data, one related work [42] synthesizes a dataset of diverse text embedding tasks by prompting LLMs. Mistral-7B [21] is then fine-tuned on this dataset to learn text embeddings. However, the generation performance of LLMs may be compromised when being trained only for text embeddings. Therefore, CRITLM [29] is proposed to perform embedding and generation simultaneously by fine-tuning Mistral-7B [21].

For recommender systems, there exists a gap between the training tasks of LLMs and recommendation task due to limited recommendation data at the pretraining stage. TALLRec [3] converts the recommendation task, i.e., whether an item would be liked by a user, into an instruction format that includes the task instruction, the task input, and the task output. They leverage the LLaMA-7B [40] model with LoRA [18] for fine-tuning. InstructRec [49] formulates the recommendation task into natural language instructions from four aspects: preferences, intentions, tasks, and content. Then, the backbone Flan-T5 [8] model is fine-tuned based on the constructed instructions. BIGRec [2] fine-tunes a large language model using the constructed sequential recommendation instructions data. It then matches the generated item tokens with the corresponding actual items by calculating the distance between their embeddings. TransRec [26] proposes to represent item information based on multifaceted identifiers, namely item ID, item name, and item attributes. Each user's interaction sequence is then converted into different aspects of instructions for bridging LLMs to recommendations.

In our work, we unify the CRS subtasks into retrieval and generation forms, and formulate them into instruction data formats. Our

Table 1: Illustrative examples of instructions for CRS subtasks. <s> and </s> are the start and end tokens, <|user|>, <|embed|>, <|assistant|> are special tokens related to the user, the embedding, and the system respectively.

Task	Input Instruction	Output
Conv2Item	<s><|user|>{Query Instruction}: Retrieve relevant items based on user conversation history<|Embed|>{Conversation Context} <s><|user|>{Sample Instruction}: Represent the item for retrieval<|Embed|>{Item Description}	Text embeddings
Conv2Conv	<s><|user|>{Query Instruction}: Given a user's conversation history, retrieve conversations from other users with similar intents<|Embed|>{Conversation Context} <s><|user|>{Sample Instruction}: Represent the conversation context for similar user intention retrieval<|Embed|>{Conversation Context}	Text embeddings
Ranking	<s><|user|>Rank the candidate items, each identified by a unique number in square brackets, based on their relevance score to the conversation context and referring to the retrieved knowledge. - Candidate Items:{} - Conversation context:{} -Retrieved Knowledge:{} Output the top {} results from most relevant to least relevant, listing the identifiers on separate lines	<|assistant|> {Ranked candidate list}</s>
Dialogue Management	<s><|user|>Analyze the conversation context: {}. Determine the user's intention and suggest a system dialogue action. Provide your explanation and suggested action, enclosed in special tokens <a></a>	<|assistant|>{Next system action}</s>
Response Generation	<s><|user|>Act as an intelligent conversational recommender system. When responding, adhere to these guidelines: - Conversation Context:{} - Use this to inform your dialogue. -Recommended Items:{} - When available, include these in your response. - Response Rules: With Items: Seamlessly incorporate the recommended items within <item></item> into the response. Without Items: Generate a contextually relevant response that assists the user	<|assistant|> {Response}</s>

model is then fine-tuned using the constructed instruction data in order to effectively perform retrieval and generation tasks.

3 METHODOLOGY

3.1 Overview

In our approach, we choose GRITLM [29] as our base model due to its exceptional performance on the massive text embedding benchmark and a variety of generation tasks. For recommendation task, we adopt a two-stage pipeline. *First*, candidate items are retrieved from a predefined item corpus. *Next*, recommendations are generated by ranking these candidate items with reference to the retrieved collaborative knowledge. For dialogue generation, we introduce dialogue management task, which controls the dialogue workflow and response generation. These tasks can be reshaped into two forms: ***retrieval*** (i.e. candidate set retrieval and collaborative knowledge retrieval) and ***generation*** (i.e., dialogue management, ranking, and response generation).

The framework of our proposed ReFICR is depicted in Figure 2. The prompts for the retrieval and generation tasks are embedded and fed through the large language model into the last hidden states. For the retrieval task, bidirectional attention is applied to the input, and mean pooling is performed on the last hidden states to yield text embeddings. For the generation task, the model employs causal attention on the input and predicts the next tokens. The contrastive objective is jointly optimized with the language model objective to train the model for both retrieval and generation tasks.

3.2 Problem Statement

Conversational recommender systems are able to interactively elicit the user's preferences during multi-turn conversations and take actions based on the user's current intention. Formally, let H denote the conversation history between all users and a conversational recommender. $C_i = \{u_j\}_{j=1}^{n}$ is a conversation context between a user i and the recommender, where $C_i \in H$ and u_j is the j-th utterance produced by either the user or the recommender. The item corpus is denoted as $I = \{e_k : d_k\}_{k=1}^{m}$, where e_k represents an item and d_k is its corresponding description that can be obtained from its meta information. The goal of the conversational recommender can be defined as follows: at the t-th conversation turn, given the conversation history $C_i = \{u_j\}_{j=1}^{t-1}$ and item corpus I, the system determines whether to recommend or not according to the user's intention. If it will recommend, the item set I_t is selected from the item corpus I as the recommendation set, and a natural language response u_t containing I_t is generated. Otherwise, the response u_t is directly generated to respond to the user's query.

3.3 Contrastive Learning for Retrieval

We formulate the retrieval task in our CRS in instruction format, so that, for the same text input, we can customize it to different embeddings based on specific instructions. We define two types of retrieval subtask: *Conv2Item*, where candidate items are retrieved from the item corpus I given the current conversation context C_i; *Conv2Conv*, where conversations with similar intentions are retrieved from all other users' conversations H. The *Conv2Item* subtask aims to narrow down the candidate set for downstream ranking, while the *Conv2Conv* subtask is intended to introduce collaborative knowledge.

The prompt templates of *Conv2Item* and *Conv2Conv* can be seen in Table 1. The model takes the text (e.g., conversation context, item description) along with the retrieval task instructions as input and outputs the corresponding text embeddings, which are tensors of numbers. To be specific, the conversation context C_i, as an input query, is encoded into a hidden representation according to the instruction of the retrieval task. The documents can be retrieved based on the similarity between the query and the document embeddings. The documents are the items in *Conv2Item* and the conversations of similar users in *Conv2Conv*. To learn the text embeddings, our ReFICR is fine-tuned using a contrastive learning objective, which involves bringing semantically similar text pairs together and pushing apart irrelevant pairs.

We construct retrieval instruction instances for the two subtasks using the ReDial [24] and INSPIRED [15] datasets. To identify positive and negative samples for *Conv2Item*, given the conversation context C_i as a query, S_i can represent its corresponding positive sample, which is the recommended item's description d_i in the ground truth response. I_c and I_s represent the query instruction and sample instruction, respectively, as shown in Table 1.

We then obtain hard negative samples through the following two steps:

First, since our backbone model GRITLM [29] has been fine-tuned on limited conversational retrieval tasks, it is difficult to acquire high-quality hard negative samples directly using the GRITLM embeddings. Therefore, we first derive our initial model, denoted as ReFICR_INIT, based on GRITLM, to improve the text embeddings. We achieve this by adopting the contrastive learning loss function [13] for the positive pair (C_i, S_i) with a mini-batch of N pairs, defined as follows:

$$\mathcal{L}_{init} = -\frac{1}{N}\sum_{i=1}^{N} \log \frac{e^{\mathbf{sim}(\mathbf{E_g}[I_c \oplus C_i], \mathbf{E_g}[I_s \oplus S_i])/\tau}}{\sum_{j=1}^{N} e^{\mathbf{sim}(\mathbf{E_g}[I_c \oplus C_i], \mathbf{E_g}[I_s \oplus S_j])/\tau}} \tag{1}$$

where $\oplus$ represents the concatenation of the input text and its corresponding instruction, τ is a temperature hyperparameter, $\mathbf{E_g}(\cdot)$ indicates that GRITLM [29] is used as the encoder to obtain the text embeddings, and $\text{sim}(\cdot,\cdot)$ is the cosine similarity between the text pairs.

Second, we construct hard negative samples using the embeddings of our initial model ReFICR_INIT, denoted as $\mathbf{E_r}(\cdot)$. We select hard negative samples from the item corpus that are close to the conversation context but are not the ground truth items. We calculate the similarity between C_i and each item description $S'_j \neq d_i$ in the item corpus I to select the k items with the highest s_{neg} scores, denoted as $\mathcal{N} = \{S'_j\}_{j=1}^{k}$. The score s_{neg} is computed as follows:

$$s_{neg} = \mathbf{sim}(\mathbf{E_r}([I_c \oplus C_i]), \mathbf{E_r}[I_s \oplus S'_j]) \tag{2}$$

For the *Conv2Conv* subtask, the objective is to retrieve the conversation context of other users with similar intentions to the current user's (i.e., C_i) from H. Because there is no ground truth data available for this subtask, positive and negative samples can be obtained as follows: *Initially*, a conversation history $C_j \in H$ is selected as the positive sample, denoted as S_i, if it shares the same ground truth recommendation item with C_i. The positive pairs (C_i, S_i), along with the text pairs at the first stage of *Conv2Item*, are optimized with the contrastive objective in Equation (1) for the same purpose. *Next*, instead of using in-batch negatives, we improve the quality of negative samples. For C_i and $\forall C_j \in H$, the k conversation histories with the highest scores s_{neg}, denoted as $\mathcal{N} = \{S'_j\}_{j=1}^{k}$, are chosen as the hard negative samples if their ground truth items differ from C_i. The score is calculated as follows:

$$s_{neg} = \mathbf{sim}(\mathbf{E_r}[I_c \oplus C_i], \mathbf{E_r}[I_s \oplus C_j]) - \mathbf{sim}(\mathbf{E_r}[d_i], \mathbf{E_r}[d_j]) \tag{3}$$

where d_i and d_j are item descriptions of the ground truth recommendation corresponding to C_i and C_j, respectively.

For both retrieval subtasks, we consider the positive pair (C_i, S_i) and the negative pairs $\mathcal{N} = \{(C_i, S'_j)\}_{j=1}^{k}$. To maximize the distance in the positive pair and minimize the distance in the negative pairs, we optimize the contrastive objective [47]:

$$\mathcal{L}_{C_i} = \frac{e^{\mathbf{sim}(\mathbf{E_g}[I_c \oplus C_i], \mathbf{E_g}[I_s \oplus S_i])/\tau}}{e^{\mathbf{sim}(\mathbf{E_g}[I_c \oplus C_i], \mathbf{E_g}[I_s \oplus S_i])/\tau} + \sum_{(C_i, S'_j) \in \mathcal{N}} e^{\mathbf{sim}(\mathbf{E_g}[I_c \oplus C_i], \mathbf{E_g}[I_s \oplus S'_j])/\tau}} \tag{4}$$

$$\mathcal{L}_{cl} = -\frac{1}{H}\sum_{C_i \in H} \log(\mathcal{L}_{C_i}) \tag{5}$$

3.4 Supervised Fine-tuning for Generation

In this subsection, we first introduce three generation subtasks, followed by the instruction data construction and LLM optimization.

For recommendation generation, after retrieving the candidate set by executing the *Conv2Item* task and collaborative knowledge by executing the *Conv2Conv* task, we perform a ranking task. Because large language models demonstrate robust listwise ranking capabilities [17, 39], we prompt the LLM to comprehend the users' preferences and rank the candidate items based on their relevance. By referring to the retrieved candidate items and collaborative knowledge, our model can ground the generated recommendations in factual item space, potentially mitigating hallucinated generation.

For conversation-related tasks, the control of the dialogue workflow is the core of an CRS. Executing the dialogue management task allows for providing reasonable system action based on the user's intention at each turn. Considering simplicity and flexibility, we model the dialogue management as a generation task. In each turn, our model takes the conversation context as input and generates a sequence of system actions expressed in natural language. The response generation task is to generate fluent responses that align with the conversation context and recommendation results.

Each instance of instruction data for generation tasks consists of an instruction that describes the task and an anticipated output. The prompt templates are shown in Table 1. The (instruction-output) pairs are constructed by applying prompt templates to text-label pairs from existing annotated CRS datasets [5, 15, 24]. To be more specific, the dialogue management instruction data is constructed based on the IARD dataset [5], where user intents and recommender actions are annotated at each dialogue turn. The instruction is to analyze the conversation context and predict the next system action with an explanation. The specific conversation context and the system's output are obtained from data examples in the dataset. For example, consider the following conversation context: "*I don't really like horror movies. What about thrillers?*" The annotated user intent is "Give Feedback and Reject", and the corresponding labeled system action is "Recommend". An anticipated output for the next system action, along with an explanation, can be constructed as follows: "<a>Recommend</a>, since the user has expressed 'dislike' for the recommended item, the system should provide alternative recommendations in the next interaction." Here, <a> and </a> are special tokens to identify the system action.

For the ranking task, we prompt our ReFICR to rank based on the conversation context, the retrieved candidate set, and the corresponding retrieved collaborative knowledge. The model leverages its intrinsic knowledge to comprehend the conversation context and incorporates the external knowledge to generate a ranked list of items, ordered from the most to the least relevant. To generate coherent responses that correspond to users' queries, the ground truth response for each conversation context within the datasets [15, 24] is considered as the anticipated output for the response generation task.

With the constructed instruction data, we can optimize the large language model through instruction tuning, which essentially is a supervised fine-tuning approach. Given that both the instruction and the target output can be expressed in natural language, we can consolidate the training process into a unified sequence-to-sequence framework. Specifically, we minimize the following loss for model training:

$$\mathcal{L}_{gen} = - \sum_{(x,y)\in\mathcal{D}} \sum_{t=1}^{|y|} log(P_\theta(y_t|y_{<t}, x)) \tag{6}$$

where x and y represent instruction input and output respectively, y_t is the predicted next word, $y_{<t}$ is the sequence of all the previous words, and θ is the model parameter.

Finally, we define the loss function of ReFICR to incorporate both $\mathcal{L}_{cl}$ and $\mathcal{L}_{gen}$ as follows:

$$\mathcal{L} = \lambda_{cl}\mathcal{L}_{cl} + \lambda_{gen}\mathcal{L}_{gen} \tag{7}$$

We concretely employ the parameter-efficient fine-tuning approach QLoRA [9], for reducing memory usage when fine-tuning large language models while preserving task performance.

4 EXPERIMENTS

4.1 Datasets

We conducted experiments on two public English conversational recommendation datasets, namely ReDial [22] and INSPIRED [15], to evaluate the performance of our model. These datasets were collected in the scenario of providing movie recommendations by crowd-sourcing workers on Amazon Mechanical Turk. The ReDial dataset comprises a training set of 10,006 dialogues and a test set of 1,342 dialogues. The movie set for ReDial was gathered from film type entities of DBpedia [4]. The INSPIRED dataset consists of 1,001 dialogues, with 801 dialogues used for training, 99 for validation, and the remaining 99 for testing. The dataset contains a total of 17,869 recommended movies, along with their meta information.

4.2 Baselines

We have considered the representative baseline models including traditional CRS models and LLM-based models, to be compared with our model ReFICR.

The traditional CRS baseline models include:

- **ReDial [24]:** This model leverages HRED [36] framework for conversation module and autoencoder for recommendation.
- **KBRD [6]:** This model incorporates knowledge from DBpedia [4], alongside the items mentioned in the conversation history, to represent user preferences for recommendations. The Transformer architecture is employed for performing the conversation task.
- **KGSF [51]:** This model utilizes external knowledge graphs, DBpedia [4], and ConceptNet [37], to introduce item-level and word-level knowledge when recommending. The response generation module is a knowledge-enhanced encoder-decoder based on the Transformer architecture.
- **BARCOR [43]:** This model fine-tunes recommendation and dialogue tasks within a single pre-trained language model BART [23] and utilizes the constructed knowledge graph CORG to enhance the representation of items.
- **UniCRS [45]:** This model is based on the pre-trained language model DialoGPT [50]. The recommendation and dialogue subtasks are formulated into a prompt learning form.
- **MESE [48]:** This model includes an item encoder and a pre-trained language model GPT-2 [34]. The item encoder converts the meta-information of the item into embeddings. The pre-trained model combines the item representation and the conversation context to make recommendations and generate responses.
- **PECRS [35]:** This model proposes a parameter-efficient plugin module and employs the frozen pre-trained language model GPT-2 [34] as a backbone to unify response generation and item recommendation tasks.

The LLM-based baseline models include:

- **GPT-3.5-turbo [32]** and **GPT-4 [33]**: They are closed-source large language models developed by OpenAI.
- **Vicunna-13B [7]**: This is a representative open-source large language model.
- **GRITLM [29]**: This model is fine-tuned based on Mistrial-7B [21] for handling both generation and embedding tasks through generative and representational instruction tuning.

4.3 Evaluation Metrics

Following the previous works on CRS [35, 51], we adopted *Recall@k* as the evaluation metric for the recommendation task, where $k \in \{1, 5, 10, 50\}$. For the response generation task, we utilized *Perplexity (PPL)* to measure the negative log-likelihood of the correct sequence output by the model, which is a common metric for language modeling tasks. Additionally, we adopted *Distinct@n*, where $n \in \{2, 4\}$, to measure the diversity of the generated responses.

4.4 Implementation

In our experiments, we initialized our ReFICR model with the large language model, GRITLM-7B [29], because of its outstanding performance on the massive text embedding and generation benchmark. ReFICR was fine-tuned for 2 epochs with a batch size of 2. We adopted the AdamW optimizer with a learning rate of 2e-5 and a warmup rate of 0.03. For retrieval tasks, the maximum token lengths of the queries and samples were set to 512 and 1024 respectively.

Table 2: Recommendation performance on the two datasets. The number in bold denotes the best result regarding that metric and the second best is underlined. † means the results are quoted from their original papers.

Method	ReDial			INSPIRED		
	Recall@1	Recall@10	Recall@50	Recall@1	Recall@10	Recall@50
ReDial [24]	0.018	0.056	0.182	0.006	0.065	0.288
KBRD [6]	0.033	0.179	0.350	0.058	0.134	0.230
KGSF [51]	0.037	0.186	0.371	0.064	0.139	0.239
BARCOR [43]	0.024	0.138	0.329	0.036	0.136	0.297
UniCRS [45]	0.049	0.214	0.421	0.096	0.236	0.393
MESE† [48]	0.056	0.256	0.455	0.048	0.135	0.301
PECRS† [7]	0.058	0.225	0.416	0.057	0.179	0.337
Vicunna-13B [16]	0.031	0.164	-	0.067	0.104	-
GPT-3.5-turbo [32]	0.039	0.168	-	0.051	0.150	-
GPT-4 [33]	0.045	0.194	-	0.091	0.194	-
GRITLM [29]	0.030	0.169	0.302	0.064	0.183	0.263
ReFICR	**0.061**	**0.305**	**0.532**	**0.135**	**0.274**	0.396

Table 3: Response generation performance on the two datasets.

Method	ReDial			INSPIRED		
	PPL ↓	Dist-2 ↑	Dist-4 ↑	PPL ↓	Dist-2 ↑	Dist-4 ↑
ReDial [24]	28.1	0.225	0.236	286.9	0.406	2.205
KBRD [6]	17.9	0.263	0.432	34.89	0.484	3.782
KGSF [51]	5.60	0.289	0.519	27.14	0.545	4.778
UniCRS [45]	11.6	0.413	0.745	17.11	3.680	5.036
MESE† [48]	12.9	0.822	1.313	-	-	-
PECRS† [7]	8.98	0.820	2.154	-	-	-
ReFICR	**4.80**	**1.478**	**2.817**	**12.51**	**4.11**	**6.17**

Table 4: The results of the ablation study on the INSPIRED dataset for the recommendation subtasks including Conv2Item, Conv2Conv, and Ranking. The symbol '/' denotes "without".

Method	INSPIRED	
	Recall@1	Recall@5
Conv2Item+Random Ranking	0.018	0.128
Conv2Item/instruction	0.060	0.142
Conv2Item	0.073	0.205
Conv2Item+Ranking+*Conv2Conv*/FT	0.074	0.206
Conv2Item+Ranking	0.084	0.207
Conv2Item+Ranking+Conv2Conv	**0.135**	**0.229**

The maximum token length for the generation task was 2048. If the sentence length exceeded the limit, it would be truncated. The ratio of positive to hard negative samples used for contrastive learning was 1:9. We implemented parameter-efficient fine-tuning using QLoRA with a rank of 64, an alpha value of 16, and a dropout rate of 0.05, using the PEFT library. For experimental details regarding LLM-based baselines, we adhered to the settings and prompts utilized in the previous works [16] for GPT-3.5-turbo [32], GPT-4 [33], and Vicuna-13B [7], as zero-shot conversational recommenders. We employed GRITLM [29] to encode the conversation context and item descriptions, predicting recommendations by calculating the cosine similarity between the representations of the conversation context and the items. The settings and implementations of baselines were based on their released open-source codes. The results for MESE [48] and PECRS [35] were quoted from the original papers. Our experiments were conducted on a single NVIDIA A100 80G GPU.

4.5 Overall Performance

4.5.1 Recommendation. Table 2 presents the recommendation results from our proposed model, ReFICR, and the baselines on two datasets: ReDial and INSPIRED. The results demonstrate that ReFICR outperforms all the baselines on both datasets, especially in terms of *Recall@10* and *Recall@50* for ReDial, and *Recall@1* and *Recall@10* for INSPIRED. This indicates that the retrieval and ranking processes in our model are effective for improving recommendation accuracy. Moreover, ReFICR can enhance the text embeddings through instruction tuning, resulting in its superior performance. More specifically, ReFICR achieves better performance compared to traditional CRS methods, including those that utilize external knowledge graphs to enhance user preference representation such as KGFS [51] and UniCRS [45], as well as models that leverage item descriptions for recommendations such as MESE [48] and PECRS [35]. ReFICR does not rely on additional modules, and it can tailor the text embeddings to different specific tasks by using instructions. The results verify that the text embeddings of large language models are powerful for obtaining better user preference representations.

In comparison to LLM-based models, directly prompting large language models to elicit user preferences within a conversation context can generate recommendations with comparable or even superior performance to those produced by traditional CRS methods. ReFICR can further improve recommendation performance through fine-tuning. In particular, compared to GRITLM [29] from which ReFICR is initialized, the performance can be significantly improved by constructing CRS subtask instructions and fine-tuning, especially on the Redial dataset. This might be attributed to the fact

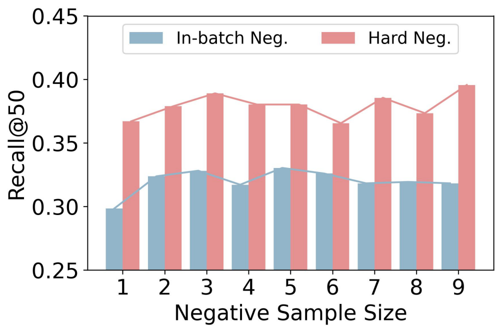

Figure 3: Performance comparison between different negative samples using Recall@50 on the INSPIRED Dataset.

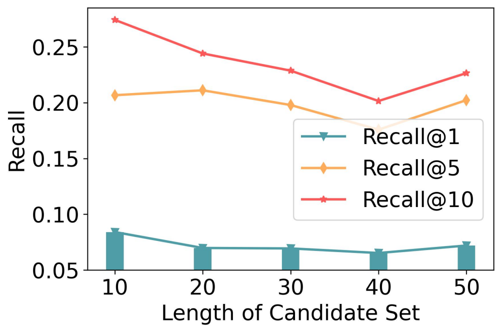

Figure 4: Ranking generation performance regarding different lengths of retrieved candidate set on the INSPIRED dataset.

that GRITLM primarily focuses on the information retrieval benchmark during tuning, which creates a gap from the recommendation task of CRSs.

4.5.2 Response Generation. The experimental results of response generation are summarized in Table 3. The dialogue models employed in the compared baseline methods can be grouped into three categories: Redial [24] based on recurrent neural networks (RNNs); KBRD [6] and KGFS [51] utilizing Transformer architecture; and UniCRS [45], MESE [48], and PECRS [7] with pre-trained language models. As generative models have evolved from RNNs to the Transformer architecture and, subsequently, to PLMs, the results w.r.t. *PPL* have shown a decreasing trend, while those w.r.t. *Distinct@n* have shown an increasing trend. This indicates the models' improving performance in generating coherent and diverse responses.

Notably, our proposed ReFICR model outperforms all baselines in terms of *Perplexity (PPL)* and *Distinct* metrics, thereby verifying that ReFICR can generate more diverse responses while maintaining consistency with the ground truth. This may be attributed to the fact that our model is pre-trained on vast datasets, which cover a wide range of topics and contexts. Furthermore, fine-tuning the model on the response generation task may improve its ability to generate responses aligned with the ground truth.

4.6 Ablation Study and Analysis

Our ReFICR incorporates several components for improving the quality of recommendations. To verify the effectiveness of each component, we conducted the ablation study based on the INSPIRED dataset. The results are shown in Table 4 in terms of *Recall@1* and *Recall@5*. For recommendation, ReFICR executes the *Conv2Item* subtask that retrieves the top-50 candidate items, and the *Conv2Conv* task that retrieves other user queries with similar intentions to the current user's and the corresponding system-recommended items. Then, the candidate items are ranked according to the retrieved collaborative knowledge.

We compared ReFICR with the following variants: (1) To verify the effectiveness of the retrieval instruction, we conducted an experiment where we removed the instruction when retrieving the candidate set, denoted as *Con2Item/Instruction*. Our observations indicate that the model's performance drops dramatically without instruction. (2) We evaluated the model's performance by removing the ranking task and only computing Recall using the retrieved candidate set, denoted as *Con2Item*. The results indicate that ranking the candidate item set can further improve the recommendation performance. We also examined the effect of random ranking by randomly permuting the candidate set, denoted as *Conv2Item+Random Ranking*. Our findings indicate that the model ranks items according to the instructions rather than yielding random results. (3) For the retrieved collaborative knowledge, we removed the knowledge at the ranking stage, denoted as *Conv2Item+Ranking*, and found that introducing collaborative knowledge can enhance the ranking performance. However, if the subtask of retrieving collaborative knowledge is not fine-tuned, denoted as *Conv2Item+Ranking+Conv2Cov/FT*, the model may introduce noise during ranking, resulting in inferior performance compared to ranking directly without knowledge. Finally, with the fine-tuned retrieved knowledge integrated into the ranking stage, our ReFICR model outperform its variants.

4.7 Hyper-parameter Study

We analyze the impact of hyper-parameters related to negative samples on the performance of recommendations during retrieval instruction tuning. Figure 3 shows the *Recall@50* results of our model on the INSPIRED dataset. It indicates that introducing hard negative samples can improve the model's performance during tuning when being compared to in-batch negative samples. Due to limited computing resources, we restricted our evaluation to a range of 1 to 9 negative samples. We selected 1 positive sample and 9 negative samples for retrieval instruction tuning. Additionally, the impact of candidate length on model ranking performance is illustrated in Figure 4. The experimental results indicate that increasing the length of the candidate item set does not enhance the ranking performance, but may compromise the model's inference speed. It is conjectured that this may arise from the increased ranking difficulty of large language models, which is caused by the growing number of candidate items. Hence, we chose 10 as the length of the candidate set„ striking a balance between performance and efficiency.

Table 5: A case study of ReFICR

Conversation: R: *What type of movies do you like?* S: *I like romance, sometimes SCI-Fi. But I don't think I want to watch anything scary. I'm looking for something that I can really escape into.* **Ground truth response:** *Have you watched **Interstellar**?*
Dialogue Management: Explanation: The seeker has specific preference for the type of movie he wants to watch, which are romance and SCI-Fi, but he does not want to watch anything scary. Suggested Action: <a>Recommend</a>
Retrieved Top-10 Candidate Set: [231]Zoe [1431]Her [1902]Seeking a Friend for the End of the World [50]Ad Astra [598]Arrival [1396]Gravity [1217]Interstellar [846]The Martian [297]A Wrinkle in Time [4033]Star man
Retrieved Collaborative Knowledge: The recommender suggested Interstellar (2014) to the user according to preferences of similar users. The referred content is: Seeker: *Can you help me with finding a movie trailer?* Recommender: *Absolutely! What genre of movies to you like?* Seeker: *Sci-fi and Action ...* Recommender: *Awesome! I recommend one of my personal favorites, Interstellar.*
Ranked List: [**1217**, 1902, 1396, 846, 297, 50, 598, 4033, 1431, 231]
Generated Response: *Have you seen <item>interstellar</item>? It's a science fiction epic that also weaves in elements of human drama and romance.*

4.8 Case Study

As shown in the Table 5, we present an example of the recommendations and response generated by our model. *Firstly*, ReFICR can comprehend the user's current intention and suggest a reasonable system action with an explanation. If the suggested action is recommendation, the model can retrieve the top-10 candidate items and relevant knowledge. *Next*, our model can rank the candidate items by considering the relevance to the conversation context and output the corresponding ranked items. *Finally*, given both the conversation context and the recommendation, our model is capable of generating a coherent response that includes the recommended item.

5 CONCLUSION

In this work, we propose ReFICR, a novel end-to-end instruction-following conversational recommender model. We decompose the complex CRS task into five subtasks: candidate set retrieval, collaborative knowledge retrieval, ranking, dialogue management, and response generation. We transform these subtasks into their corresponding retrieval and generation instructions by utilizing prompt templates. Contrastive learning is employed to enhance text embeddings, which are customized for different retrieval subtasks through specific retrieval instructions. Additionally, supervised fine-tuning is applied to improve performance of generation subtasks. Experimental results show that ReFICR outperforms the baseline methods, demonstrating its promising recommendation performance and ability to generate diverse and informative responses.

However, although we employ parameter-efficient fine-tuning where only a limited number of parameters are tuned, LLMs may still face catastrophic forgetting when being fine-tuned on the specific tasks. In the future, we will validate the performance of our model on tasks such as embedding tasks from the MTEB benchmark [30] and open-ended generation tasks from AlpacaEval [25]. We will also explore continual learning approaches, such as replay-based methods [19, 46], to mitigate the catastrophic forgetting issue. On the other hand, the computational cost and inference efficiency can be critical to the use of LLMs in practical situations. To address this challenge, acceleration and parallel computing techniques might be utilized. We will also investigate knowledge distillation from large language models to small models in our future work to potentially enhance computational efficiency and facilitate the deployment of these models in resource-constrained environments.

ACKNOWLEDGMENTS

This work is supported by Hong Kong Baptist University IG-FNRA project (RC-FNRA-IG/21-22/SCI/01) and Key Research Partnership Scheme (KRPS/23-24/02).

REFERENCES

[1] Akari Asai, Timo Schick, Patrick Lewis, Xilun Chen, Gautier Izacard, Sebastian Riedel, Hannaneh Hajishirzi, and Wen-tau Yih. 2022. Task-aware retrieval with instructions. *arXiv preprint arXiv:2211.09260* (2022).

[2] Keqin Bao, Jizhi Zhang, Wenjie Wang, Yang Zhang, Zhengyi Yang, Yancheng Luo, Chong Chen, Fuli Feng, and Qi Tian. 2023. A Bi-Step Grounding Paradigm for Large Language Models in Recommendation Systems. arXiv:2308.08434 [cs.IR]

[3] Keqin Bao, Jizhi Zhang, Yang Zhang, Wenjie Wang, Fuli Feng, and Xiangnan He. 2023. Tallrec: An effective and efficient tuning framework to align large language model with recommendation. In *Proceedings of the 17th ACM Conference on Recommender Systems*. 1007–1014.

[4] Christian Bizer, Jens Lehmann, Georgi Kobilarov, Sören Auer, Christian Becker, Richard Cyganiak, and Sebastian Hellmann. 2009. Dbpedia-a crystallization point for the web of data. *Journal of web semantics* 7, 3 (2009), 154–165.

[5] Wanling Cai and Li Chen. 2020. Predicting user intents and satisfaction with dialogue-based conversational recommendations. In *Proceedings of the 28th ACM Conference on User Modeling, Adaptation and Personalization*. 33–42.

[6] Qibin Chen, Junyang Lin, Yichang Zhang, Ming Ding, Yukuo Cen, Hongxia Yang, and Jie Tang. 2019. Towards Knowledge-Based Recommender Dialog System. arXiv:1908.05391 [cs.CL]

[7] Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E Gonzalez, et al. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality. *See https://vicuna. lmsys. org (accessed 14 April 2023)* 2, 3 (2023), 6.

[8] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2024. Scaling instruction-finetuned language models. *Journal of Machine Learning Research* 25, 70 (2024), 1–53.

[9] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2024. Qlora: Efficient finetuning of quantized llms. *Advances in Neural Information Processing Systems* 36 (2024).

[10] Jiabao Fang, Shen Gao, Pengjie Ren, Xiuying Chen, Suzan Verberne, and Zhaochun Ren. 2024. A Multi-Agent Conversational Recommender System. arXiv:2402.01135 [cs.IR]

[11] Yue Feng, Shuchang Liu, Zhenghai Xue, Qingpeng Cai, Lantao Hu, Peng Jiang, Kun Gai, and Fei Sun. 2023. A large language model enhanced conversational recommender system. *arXiv preprint arXiv:2308.06212* (2023).

[12] Luke Friedman, Sameer Ahuja, David Allen, Terry Tan, Hakim Sidahmed, Changbo Long, Jun Xie, Gabriel Schubiner, Ajay Patel, Harsh Lara, et al. 2023. Leveraging Large Language Models in Conversational Recommender Systems. *arXiv preprint arXiv:2305.07961* (2023).

[13] Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. Simcse: Simple contrastive learning of sentence embeddings. *arXiv preprint arXiv:2104.08821* (2021).

[14] Yunfan Gao, Tao Sheng, Youlin Xiang, Yun Xiong, Haofen Wang, and Jiawei Zhang. 2023. Chat-rec: Towards interactive and explainable llms-augmented recommender system. *arXiv preprint arXiv:2303.14524* (2023).

[15] Shirley Anugrah Hayati, Dongyeop Kang, Qingxiaoyang Zhu, Weiyan Shi, and Zhou Yu. 2020. Inspired: Toward sociable recommendation dialog systems. *arXiv*

preprint arXiv:2009.14306 (2020).

[16] Zhankui He, Zhouhang Xie, Rahul Jha, Harald Steck, Dawen Liang, Yesu Feng, Bodhisattwa Prasad Majumder, Nathan Kallus, and Julian McAuley. 2023. Large language models as zero-shot conversational recommenders. In *Proceedings of the 32nd ACM international conference on information and knowledge management.* 720–730.

[17] Yupeng Hou, Junjie Zhang, Zihan Lin, Hongyu Lu, Ruobing Xie, Julian McAuley, and Wayne Xin Zhao. 2024. Large language models are zero-shot rankers for recommender systems. In *European Conference on Information Retrieval.* Springer, 364–381.

[18] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685* (2021).

[19] Jianheng Huang, Leyang Cui, Ante Wang, Chengyi Yang, Xinting Liao, Linfeng Song, Junfeng Yao, and Jinsong Su. 2024. Mitigating catastrophic forgetting in large language models with self-synthesized rehearsal. *arXiv preprint arXiv:2403.01244* (2024).

[20] Dietmar Jannach, Ahtsham Manzoor, Wanling Cai, and Li Chen. 2021. A survey on conversational recommender systems. *ACM Computing Surveys (CSUR)* 54, 5 (2021), 1–36.

[21] Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7B. *arXiv preprint arXiv:2310.06825* (2023).

[22] Wenqiang Lei, Xiangnan He, Yisong Miao, Qingyun Wu, Richang Hong, Min-Yen Kan, and Tat-Seng Chua. 2020. Estimation-action-reflection: Towards deep interaction between conversational and recommender systems. In *Proceedings of the 13th International Conference on Web Search and Data Mining.* 304–312.

[23] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461* (2019).

[24] Raymond Li, Samira Ebrahimi Kahou, Hannes Schulz, Vincent Michalski, Laurent Charlin, and Chris Pal. 2018. Towards deep conversational recommendations. *Advances in neural information processing systems* 31 (2018).

[25] Xuechen Li, Tianyi Zhang, Yann Dubois, Rohan Taori, Ishaan Gulrajani, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. 2023. Alpacaeval: An automatic evaluator of instruction-following models.

[26] Xinyu Lin, Wenjie Wang, Yongqi Li, Fuli Feng, See-Kiong Ng, and Tat-Seng Chua. 2023. A multi-facet paradigm to bridge large language model and recommendation. *arXiv preprint arXiv:2310.06491* (2023).

[27] Yu Lu, Junwei Bao, Yan Song, Zichen Ma, Shuguang Cui, Youzheng Wu, and Xiaodong He. 2021. RevCore: Review-Augmented Conversational Recommendation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli (Eds.). Association for Computational Linguistics, Online, 1161–1173. https://doi.org/10.18653/v1/2021.findings-acl.99

[28] Bonan Min, Hayley Ross, Elior Sulem, Amir Pouran Ben Veyseh, Thien Huu Nguyen, Oscar Sainz, Eneko Agirre, Ilana Heintz, and Dan Roth. 2023. Recent advances in natural language processing via large pre-trained language models: A survey. *Comput. Surveys* 56, 2 (2023), 1–40.

[29] Niklas Muennighoff, Hongjin Su, Liang Wang, Nan Yang, Furu Wei, Tao Yu, Amanpreet Singh, and Douwe Kiela. 2024. Generative representational instruction tuning. *arXiv preprint arXiv:2402.09906* (2024).

[30] Niklas Muennighoff, Nouamane Tazi, Loïc Magne, and Nils Reimers. 2022. MTEB: Massive text embedding benchmark. *arXiv preprint arXiv:2210.07316* (2022).

[31] Arvind Neelakantan, Tao Xu, Raul Puri, Alec Radford, Jesse Michael Han, Jerry Tworek, Qiming Yuan, Nikolas Tezak, Jong Wook Kim, Chris Hallacy, et al. 2022. Text and code embeddings by contrastive pre-training. *arXiv preprint arXiv:2201.10005* (2022).

[32] OpenAI. 2022. Introducing chatgpt. https://openai.com/blog/chatgpt

[33] OpenAI. 2023. GPT-4 Technical Report. https://doi.org/10.48550/arXiv.2303.08774

[34] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog* 1, 8 (2019), 9.

[35] Mathieu Ravaut, Hao Zhang, Lu Xu, Aixin Sun, and Yong Liu. 2024. Parameter-Efficient Conversational Recommender System as a Language Processing Task. arXiv:2401.14194 [cs.CL]

[36] Alessandro Sordoni, Yoshua Bengio, Hossein Vahabi, Christina Lioma, Jakob Grue Simonsen, and Jian-Yun Nie. 2015. A hierarchical recurrent encoder-decoder for generative context-aware query suggestion. In *proceedings of the 24th ACM international on conference on information and knowledge management.* 553–562.

[37] Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 31.

[38] Hongjin Su, Weijia Shi, Jungo Kasai, Yizhong Wang, Yushi Hu, Mari Ostendorf, Wen-tau Yih, Noah A Smith, Luke Zettlemoyer, and Tao Yu. 2022. One embedder, any task: Instruction-finetuned text embeddings. *arXiv preprint arXiv:2212.09741* (2022).

[39] Weiwei Sun, Lingyong Yan, Xinyu Ma, Shuaiqiang Wang, Pengjie Ren, Zhumin Chen, Dawei Yin, and Zhaochun Ren. 2023. Is ChatGPT good at search? investigating large language models as re-ranking agents. *arXiv preprint arXiv:2304.09542* (2023).

[40] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971* (2023).

[41] Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2022. Text embeddings by weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533* (2022).

[42] Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. 2024. Improving Text Embeddings with Large Language Models. arXiv:2401.00368 [cs.CL]

[43] Ting-Chun Wang, Shang-Yu Su, and Yun-Nung Chen. 2022. BARCOR: Towards A Unified Framework for Conversational Recommendation Systems. *arXiv preprint arXiv:2203.14257* (2022).

[44] Xiaolei Wang, Xinyu Tang, Xin Zhao, Jingyuan Wang, and Ji-Rong Wen. 2023. Rethinking the Evaluation for Conversational Recommendation in the Era of Large Language Models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, Houda Bouamor, Juan Pino, and Kalika Bali (Eds.). Association for Computational Linguistics, Singapore, 10052–10065. https://doi.org/10.18653/v1/2023.emnlp-main.621

[45] Xiaolei Wang, Kun Zhou, Ji-Rong Wen, and Wayne Xin Zhao. 2022. Towards unified conversational recommender systems via knowledge-enhanced prompt learning. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining.* 1929–1937.

[46] Yifan Wang, Yafei Liu, Chufan Shi, Haoling Li, Chen Chen, Haonan Lu, and Yujiu Yang. 2024. Inscl: A data-efficient continual learning paradigm for fine-tuning large language models with instructions. *arXiv preprint arXiv:2403.11435* (2024).

[47] Lingling Xu, Haoran Xie, Zongxi Li, Fu Lee Wang, Weiming Wang, and Qing Li. 2023. Contrastive learning models for sentence representations. *ACM Transactions on Intelligent Systems and Technology* 14, 4 (2023), 1–34.

[48] Bowen Yang, Cong Han, Yu Li, Lei Zuo, and Zhou Yu. 2021. Improving Conversational Recommendation Systems' Quality with Context-Aware Item Meta Information. *arXiv preprint arXiv:2112.08140* (2021).

[49] Junjie Zhang, Ruobing Xie, Yupeng Hou, Wayne Xin Zhao, Leyu Lin, and Ji-Rong Wen. 2023. Recommendation as instruction following: A large language model empowered recommendation approach. *arXiv preprint arXiv:2305.07001* (2023).

[50] Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. 2019. Dialogpt: Large-scale generative pre-training for conversational response generation. *arXiv preprint arXiv:1911.00536* (2019).

[51] Kun Zhou, Wayne Xin Zhao, Shuqing Bian, Yuanhang Zhou, Ji-Rong Wen, and Jingsong Yu. 2020. Improving conversational recommender systems via knowledge graph based semantic fusion. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining.* 1006–1014.

[52] Yuanhang Zhou, Kun Zhou, Wayne Xin Zhao, Cheng Wang, Peng Jiang, and He Hu. 2022. C^2-CRS: Coarse-to-Fine Contrastive Learning for Conversational Recommender System. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining.* 1488–1496.