Inquire, Extract, Retrieve and Re-rank: A Multi-Agent Approach for Conversational Recommender Systems

Heitor R. Coimbra[1], Tsang Ing Ren[2]

Centro de Informática
Universidade Federal de Pernambuco (UFPE) – Recife, PE – Brazil

{hrc, tir}@cin.ufpe.br

Abstract. *The overwhelming abundance of options on digital platforms presents significant challenges for users seeking personalized recommendations. Traditional recommendation systems, often reliant on historical user data and collaborative filtering, may not accurately capture current user preferences and can lack consistency and reliability, especially in cold-start scenarios. With the advent of Large Language Models (LLMs) capable of engaging in dynamic conversations, Conversational Recommender Systems (CRS) have emerged as a promising solution to enhance personalization through natural language interactions. We introduce a multi-agent CRS framework that leverages LLMs to provide personalized game recommendations without relying on historical data. Our system comprises five specialized agents: an Inquiry Agent to elicit user preferences, an Extractor Agent to structure these preferences into a user profile, a Semantic Retriever to semantically match preferences with game attributes, and two Re-rankers to refine the recommendations based on relevance and quality.*

We apply our framework to a comprehensive dataset of over 97,000 video games from Steam and employ synthetic personas generated by LLMs to simulate user interactions. The experimental results demonstrate the system's potential in effectively capturing user preferences and delivering tailored recommendations despite challenges associated with evaluating performance using synthetic data.

By advancing these AI-driven approaches, this work lays a foundation for more intuitive and responsive recommendation platforms and search engines. It transforms how users interact with digital environments by addressing the cold-start problem in recommendation systems, paving the way for enhanced user experiences through personalized, dialogue-based interactions.

1. Introduction

In the digital age, users face an overwhelming array of options, making the task of navigating online platforms increasingly complex. This challenge has been a persistent issue since the inception of web browsing. Traditional recommendation systems, which often rely on historical user data and collaborative filtering techniques [1], have sought to simplify this process. However, these systems usually need to be more accurate in capturing current user preferences, resulting in recommendations that may lack relevance, consistency, and reliability. The static nature of these traditional approaches does not accommodate modern users' dynamic and context-sensitive needs.

The limitations of historical data-driven recommendation systems are well-documented [2]. Collaborative filtering, for instance, presumes that the past behaviors and preferences of a cohort of users can effectively predict future choices for similar individuals. While this method has achieved a measure of success, it inherently fails to account for the fluid and evolving nature of user preferences, which myriad factors can influence in real time. Consequently, users often receive suggestions that feel impersonal and disconnected from their immediate context, detracting from the overall user experience [3].

The advent of Large Language Models (LLMs) marks a significant breakthrough in addressing these limitations [4]. LLMs possess the capability to understand and generate human-like text, facilitating more natural and intuitive interactions. Leveraging these capabilities, Conversational Recommendation Systems (CRS) have emerged as a revolutionary approach to personalizing user experiences on digital platforms. Unlike traditional systems, CRS engages users through a dialog-based interface, allowing for real-time, natural language interactions that dynamically adapt to user inputs [5].

This research explores the use of conversational agents powered by LLMs to operate exclusively on real-time user interactions, thereby eliminating the need for reliance on historical data. By focusing on developing and evaluating agentic frameworks and prompting techniques, this research aims to enhance the quality of recommendations and improve user experience. The core objective is to create more intuitive and responsive recommendation platforms that transform how users interact when looking for appropriate items that match their preferences.

In this paper, we will explore possible uses for LLMs not only as conversational agents but also as information extractors, item rankers, and as engines for a recommendation system, aiming to explore the upper-boundaries of CRS. We will examine the design and implementation of agentic frameworks that leverage LLMs, assess the impact of prompting techniques on recommendation quality, and evaluate user experience through synthetic user simulations. By advancing these AI-driven methodologies, we aspire to pave the way for a new era of recommendation systems that are not only more accurate and reliable but also more aligned with the real-time needs and preferences of users.

The insights from this research show that when shifting from a static, data-driven model to dynamic, interaction-based systems, we can better address the complexities of user preferences in the digital age. Ultimately, this project seeks to contribute to developing smarter, more adaptive recommendation systems that can provide users with an enriched and natural recommendation experience.

2. Background and Related Work

A few studies in the recent literature focus on advancing conversational recommender systems (CRSs) by leveraging large language models (LLMs), enabling a more dynamic and interactive recommendation process. These systems shift from traditional data-driven recommendation techniques to more natural, dialogue-based interaction frameworks that are pivotal in enhancing user experience and system adaptability.

Conversational Recommender Systems (CRS) have emerged as a promising solution to the challenges presented. By engaging users in dialogue, CRS can inquire and elicit current preferences, clarify ambiguities, and provide personalized recommendations without extensive reliance on historical data. Early foundational work focused on dialogue management and user intent understanding. For instance, Christakopoulou et al. [6] introduced methods to optimize conversational interactions, enabling systems to efficiently narrow down user preferences through strategic questioning. Sun et al. [7] expanded on this by leveraging neural attention networks to model user-system dialogues, enhancing the system's ability to interpret and respond to natural language inputs.

The integration of Large Language Models (LLMs) has significantly advanced the capabilities of CRS. LLMs can understand and generate human-like text, facilitating more natural and context-aware conversations. Recent studies have explored leveraging LLMs for zero-shot and few-shot learning in CRS, reducing the dependency on large annotated datasets.

He et al. [8] investigated the potential of LLMs as zero-shot conversational recommender systems. Their work demonstrates that LLMs, such as GPT-3, can generate personalized recommendations based solely on real-time user interactions without fine-tuning specific recommendation datasets. By designing effective prompts, the LLM interprets user preferences in natural language and provides relevant suggestions. This approach highlights the feasibility of deploying CRS in domains where historical data is sparse or unavailable.

Evaluating LLM-based CRS presents unique challenges, as traditional static metrics may not capture the interactive nature of conversations. Wang et al. [9] addressed this by introducing iEvaLM, an interactive evaluation framework for LLM-based CRS. iEvaLM employs user simulators powered by LLMs to mimic realistic user behaviors, allowing for dynamic assessment of CRS performance. The framework evaluates systems based on recommendation accuracy, conversational fluency, and user satisfaction, providing a comprehensive benchmark for future research.

Detecting breakdowns in CRS is crucial for maintaining user engagement and trust. Liu et al. [10] proposed using LLM-based user simulation to identify points where the CRS fails to understand user intents or provide satisfactory recommendations. By simulating diverse conversational scenarios, including ambiguous queries and unexpected behaviors, they demonstrated that LLM-based simulations effectively uncover system weaknesses and guide improvements.

To enhance item representation in LLM-based CRS, Zhang et al. [11] introduced the Item-Language Model (ILM). Recognizing that traditional item encodings may not fully leverage the language understanding capabilities of LLMs, ILM incorporates rich textual descriptions and metadata into item representations. This alignment enables the CRS to comprehend nuanced user preferences and generate more accurate recommendations.

Comprehensive overviews of LLMs in CRS are provided by Chen and Wang [12] and Li et al. [13]. Chen and Wang [12] discuss recent advancements, benefits, and challenges of integrating LLMs into CRS, including handling open-domain conversations and performing zero-shot recommendations. They highlight issues such as computational efficiency and managing long-term user preferences. Li et al. [13] offer a detailed review of methodologies for integrating LLMs into CRS, such as prompt engineering and hybrid models combining LLMs with traditional algorithms. Both works underscore the potential of LLMs to revolutionize CRS while acknowledging the challenges that remain.

In domain-specific applications, Silveira et al. [14] developed a conversational recommender system for the Steam video game platform. The system provides personalized game recommendations from the extensive Steam library by engaging users in interactive dialogues about their gaming preferences. Their evaluation demonstrated that the conversational approach enhances user satisfaction and facilitates the discovery of new games, exemplifying the practical benefits of CRS in a real-world setting.

Building upon these foundations, our research aims to explore using conversational agents powered by LLMs to operate exclusively on real-time, natural language interactions, eliminating the need for historical data reliance. We focus on developing and evaluating agentic frameworks and the impact of prompt engineering techniques on recommendation quality and user experience. Inspired by the evaluation methodologies of Wang et al. [4] and the user simulation approaches of Liu et al. [5], we aim to contribute to the field by addressing the challenges highlighted in the overviews by Chen and Wang [7] and Li et al. [8]. Using a synthetic video game recommendation dataset, our work seeks to advance AI-driven methodologies for more intuitive and responsive recommendation platforms, ultimately transforming how users interact with digital environments.

3. CD-ROMs and Printed Proceedings

The method proposed here focuses on developing a novel multi-agent iterative CRS completely independent from prior data and capable of making reliable and accurate recommendations for video games. Utilizing the Steam Games Dataset, which provides a set of data points for recommendations, a synthetic ground truth dataset was built to experiment with user simulation evaluation. This approach includes an examination of each agent and step in the workflow and a reliable treatment of the user's disclosed preferences on the CRS platform. The following subsections will elaborate on each phase of our methodology, providing a comprehensive overview of the processes and techniques employed.

A. Data Source

The Steam Games Dataset was obtained from Kaggle and was originally scraped using the Steam API in 2024. Steam, being the largest gaming platform on PC, offers a comprehensive repository of games, making it an ideal source for our research. The dataset compilation credits go to the original creator, and we extend our acknowledgment for their contribution.

Having a generalizable, zero-shot system perform on completely unseen data is part of the goal of this project, preparing the foundation for CRS to be integrated into multiple different domains.

The dataset comprises over 97,000 games available on the Steam platform and can be found at Kaggle. Each game entry includes several attributes pertinent to our research:

- AppID: A unique identifier for each game.
- Name: The title of the game.
- About the Game: A description provided by the developers.

- Metacritic Score: An aggregated score reflecting critical reception.
- Notes: Additional information or remarks about the game.
- Categories: Gameplay modes and features (e.g., Single-player, Multiplayer).
- Genres: The genre classification of the game (e.g., Action, Adventure).
- Tags: User-generated descriptors highlighting game features and themes.

To prepare the dataset for our CRS, we performed the following preprocessing steps:

- Data Cleaning: Removed entries with missing values and duplicate rows to ensure data integrity.
- Column Selection: Focused on the *Name*, *Categories*, *Genres*, and *Tags* columns, as these directly relate to user preferences and are essential for matching user interests with game attributes.
- Data Formatting: Ensured that the categorical data in *Categorie*s, *Genres*, and *Tags* were properly formatted for processing, converting any necessary fields into consistent string or list representations.

B. CRS Framework

The developed CRS framework is structured as a multi-agent system comprising four main components: the Inquiry Agent, the Extractor Agent, the Retriever, and two Re-ranker Agents (Tag Re-ranker and Game Re-ranker). Each component plays a specialized role in facilitating the recommendation process, and they collectively enable the CRS to interact with users, extract preferences, retrieve relevant games, and refine recommendations. Figure 1 illustrates the interaction between these components.

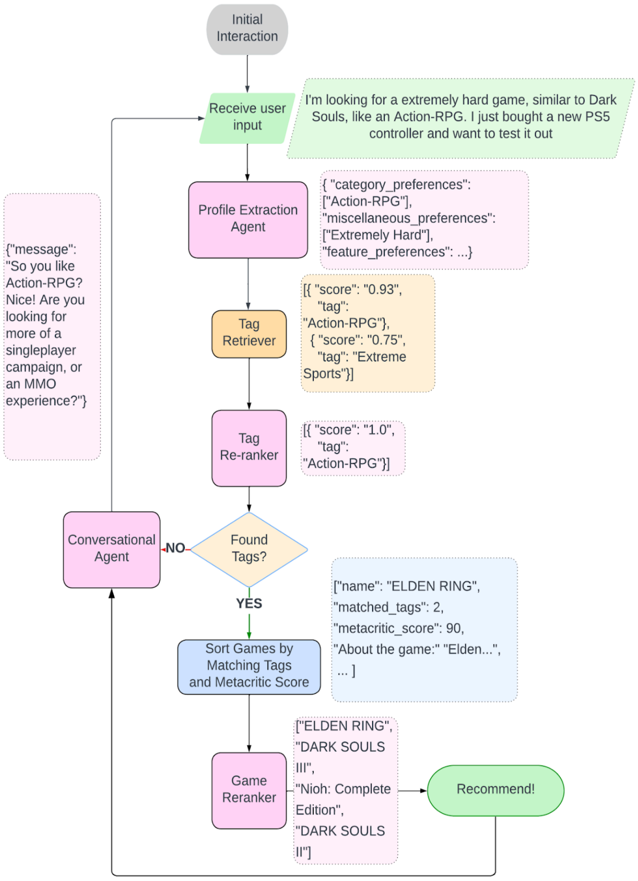

Fig. 1 Diagram of the System Architecture and Conversation Flow, from the initial interaction up to the recommendation. The user input is received, its preferences are extracted by the Extraction Agent, the closest tags are then matched by the Tag Retriever and re-ranked by the Tag Re-ranker. If the Tag Re-ranker validates the matched tags, a pool of games will be selected and then Re-ranked by the Game Re-ranker. When games are found in this method, a recommendation is made. After each iteration the Conversational Agent is triggered to inquire the user and elicit more preferences, regardless of a recommendation being made.

1) Inquiry Agent

The Inquiry Agent initiates the conversation with the user, aiming to elicit detailed preferences regarding the type of games the user is interested in. Implemented using an LLM with chain-of-thought prompting, the agent is programmed to maintain an inquisitive, curious, and helpful tone without suggesting specific games prematurely.
As for the prompt configuration, it instructs the agent to act as a conversational assistant for a video game enthusiast, focusing on exploring the user's interests in genres, features, and categories. It emphasizes brevity in responses and sets guidelines to avoid irrelevant topics.

2) Extractor Agent

The Extractor Agent processes the conversation between the user and the Inquiry Agent to extract structured user preferences. Utilizing an LLM, the Extractor generates a *UserProfile* object containing the following fields:

- category_preferences: A list of genres or categories the user is interested in (e.g., Action, Puzzle-Platformer, Detective).
- feature_preferences: A list of desired game features (e.g., Controller support, Local Multiplayer).
- miscellaneous_preferences: A list of less critical preferences or descriptors (e.g., Cozy, Fun, Chaotic).

The Extractor Agent receives not only the chat history, but also the last extracted UserProfile object to keep consistency and modify it if needed. The prompt guides the agent to:

- Analyze the chat history between the user and the Inquiry Agent.
- Extract and update the *UserProfile* based on the latest user inputs.
- Keep preferences concise and relevant to gaming terms suitable for database searching.

The Extractor ensures that all relevant preferences are captured and structured for the subsequent retrieval process.

3) **Tag Retrieval and Re-ranking**

The Retriever component is responsible for matching the extracted user preferences with relevant tags, categories and genres in the dataset. We implemented the Retriever using the *llama_index* library and the open-source embedding model *bge-small-en-v1.5* from BAAI to perform a semantic search.
We used the semantic similarity of the user's preferences and the tags found in the dataset to figure out which tags equate to the user's disclosed preferences.

1. Embedding Generation: Generate embeddings for user preferences and game attributes (Categories, Genres, Tags) using the embedding model.

2. Similarity Computation: Calculate cosine similarity scores between user preferences and game attributes.

3. Threshold Application: Apply a similarity threshold (≥ 1.5) to identify semantically relevant tags. The threshold was calculated through empirical experimentation, maximizing recall for similar tag matches.

4. Threshold Application: To refine the results from the Retriever, we employ a tag re-ranker agent. The Tag Re-ranker is prompted to validate and refine the matched tags by re-evaluating their relevance to the *UserProfile*. It filters out tags that may not accurately reflect the user's preferences. The chain-of-thought process enables the agent to consider the context and nuances in the user's preferences, potentially discarding irrelevant tags.

4) Game Selection and Re-ranking:

Having the matched tags, we filter games in the dataset that have no matching tag, genre or category with the user's preferences. Selected games are then sorted by most categories matched, and then secondarily sorted by their Metacritic score to guarantee relevant recommendations in the gaming world. Having a smaller pool of possible candidates, we use another Re-ranker agent to narrow down the potential recommendations by comparing the list of top candidates, taking as context all of the games information (description, notes, name, release date) and every information present in the UserProfile to those most aligned with the user's expressed interests.

The Game Re-ranker is prompted to reassess the list of top games obtained from the previous step. It considers comprehensive game information, including descriptions, notes, platforms, and all user preference data contained in the UserProfile object. The agent then re-scores the games based on alignment with the user's interests and may discard unsuitable games, whilst sorting the matching games considering all the user and game information. This step ensures that the final recommendations are highly relevant, personalized and sorted intelligently.

C. Implementation Details

The framework and experiment were implemented in Python 3.10, utilizing the following libraries and tools:

- Pandas: For data manipulation and processing.
- llama_index: This is for building and querying the local vector database.
- instructor: For reliably getting LLMs to respond with the structured output and chain-of-thought.
- OpenAI GPT-4o: Was selected for its balance between cost and effectiveness as the only language model for all agents.

All LLMs in the system employ chain-of-thought prompting to enhance reasoning capabilities and dialogue coherence. Also, the orchestration of the LLMs was fully customized, using mostly pure Python with no auxiliary tools for state tracking and agent management.

D. Workflow of the CRS

The CRS operates through the following steps in the conversation flow:

1. **Initiation**: The Inquiry Agent greets users and prompts them to share their gaming preferences.

2. **Preference Elicitation**: The user responds with details about the types of games they are interested in.

3. **Extraction**: After each user response, the Extractor Agent analyzes the conversation and updates the *UserProfile* with the extracted preferences.

4. **Retrieval**: The Retriever matches the user's preferences with game attributes in the dataset using semantic search.

5. **Tag Re-ranking**: The Tag Re-ranker refines the list of relevant tags, ensuring alignment with the *UserProfile*.

6. **Tag Re-ranking**: The Tag Re-ranker refines the list of relevant tags, ensuring alignment with the *UserProfile*.

7. **Game Re-ranking**: The Game Re-ranker reassesses and re-scores the top game candidates based on detailed user preferences.

8. **Recommendation**: The CRS presents the user with game recommendations that best match their interests.

9. **Continuous Iteration**: The user may continue the conversation for further recommendations, clarification, or even correcting his disclosed preferences.

For evaluation purposes, the default conversation length was set to 8 messages (including both user and agent messages). This allowed sufficient interaction for the system to gather meaningful preferences and provide recommendations. Future work may explore dynamic conversation lengths and the development of a router model capable of determining the optimal point to present recommendations based on the conversation's context.

4. Experiment Results

The primary objective of our experiments was to evaluate the performance of our multi-agent Conversational Recommender System (CRS) in providing accurate game recommendations in cold-start scenarios. We aimed to assess how effectively the system could capture user preferences through conversation and deliver relevant recommendations without relying on historical user data. To gauge the effectiveness of our multi-agent CRS, we compared it against a single-agent baseline system implemented using a Large Language Model (LLM). The single-agent system performs the entire recommendation task monolithically, engaging in conversation with the user, extracting preferences, and providing recommendations without the specialized modular components of our multi-agent framework. The model is also prompted to mention any recommendations in-between <game> tags, e.g. <game>Minecraft</game> so that we can then retrieve and evaluate which recommendations where done in that session

We utilized 50 synthetic user personas generated by a specifically prompted LLM for the evaluation. The LLM was prompted to describe each persona based on the characteristics of a single game sampled from the top games according to Metacritic scores. The agent described the personas providing detailed descriptions of the hypothetical user preferences without mentioning the specific games, enabling us to simulate realistic user interactions by prompting another LLM to roleplay as that user.

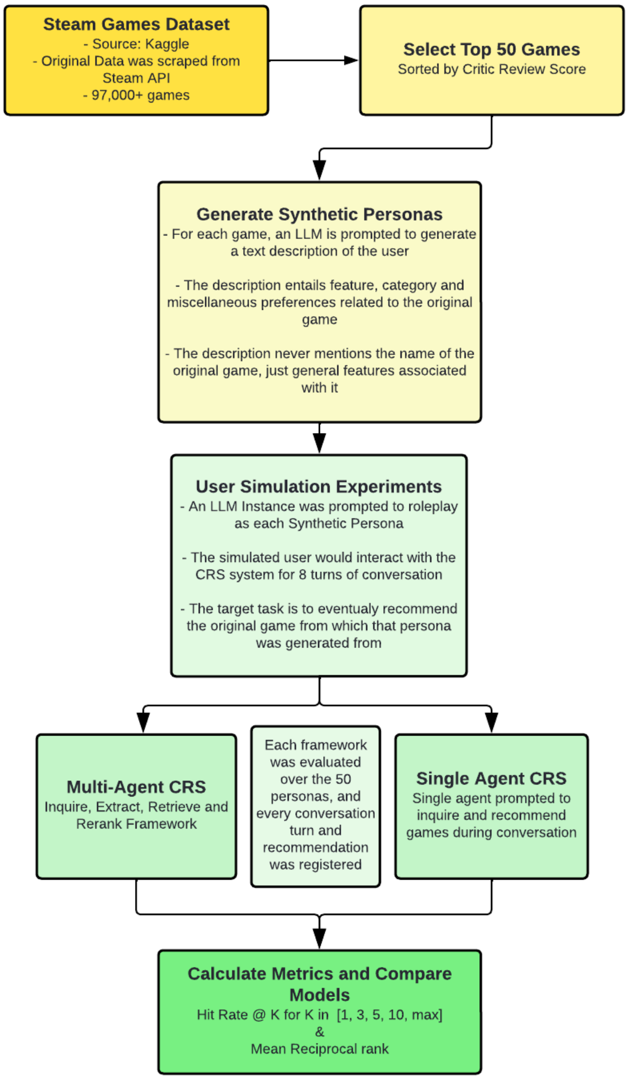

Fig. 2 Diagram showing the Data Ingestion process, Synthetic Persona Generation, User Simulation and Evaluation Workflow that were implemented for experimentation. From the initial dataset, 50 top games were selected based on the critic review score. Those 50 games were then source for the generation of synthetic personas, that were eventually used as roleplaying models for an User Simulator LLM that would simulate actual users interacting with the CRS system. Hit Rate @ K and Mean Reciprocal Rank were calculated based on the presence or not of the source game in the recommendation pool.

For each user persona, we simulated conversations between the user and both the multi-agent CRS and the single-agent baseline. The conversations were capped at a maximum of 8 turns (messages), consistent with our methodology. The user's responses were generated by LLMs acting according to the persona's description.

We evaluated the performance of both systems using the following metrics:

- **Hit Rate @ K:** Indicates the proportion of test cases where the desired game appears within the top K recommendations.
- **Mean Reciprocal Rank (MRR):** Measures the average reciprocal rank of the desired game in the full recommendation lists across all test cases.

We chose these metrics to assess the systems' ability to rank the target games highly and to capture the effectiveness of the recommendations from a user's perspective, considering the original game from where the user persona prompt was derived should be at least one of the recommendations, and ideally should be the top recommendation.

K	*Single Agent CRS*	***Our framework***
1	*0.02*	***0.44***
3	*0.04*	***0.60***
5	*0.06*	***0.66***
10	*0.12*	***0.70***
max	*0.30*	***0.70***

TABLE I

HIT RATE @ K ACROSS THE SINGLE AGENT CRS AND OUR FRAMEWORK FOR K AT 1, 3, 5, 10 AND FOR HIT RATE@MAX

Framework	*MRR*
Single-Agent CRS	*0.051*
Multi-Agent CRS	***0.537***

TABLE II

MRR ACROSS ALL RECOMMENDATIONS FOR THE SINGLE AGENT CRS AND THE MULTI-AGENT CRS

These experimental results show that the **multi-agent CRS** achieved both higher MRR and Hit Rate scores compared to the **single-agent LLM** baseline, demonstrating its effectiveness in ranking the target games more highly and accurately. At lower values of K (e.g., K=1), the multi-agent CRS showed a significant improvement in Hit Rate, suggesting that it is more likely to place the desired game at the very top of the recommendation list.

It is reasonable to understand why, as the specialized agents within the multi-agent CRS surely contributed to **better extraction** and utilization of user preferences, leading to more accurate recommendations. The Tag Retriever module had its impact in identifying actual categories inside the Steam Games Database to filter from, and then eventually both re-rankers had their part in eliminating and reprioritizing the pre-selected video games and tags.

In Figure 3 and 4, demonstrations of actual performance over unseen user personas are showcased for both the multi-agent CRS and a single agent LLM use-case.

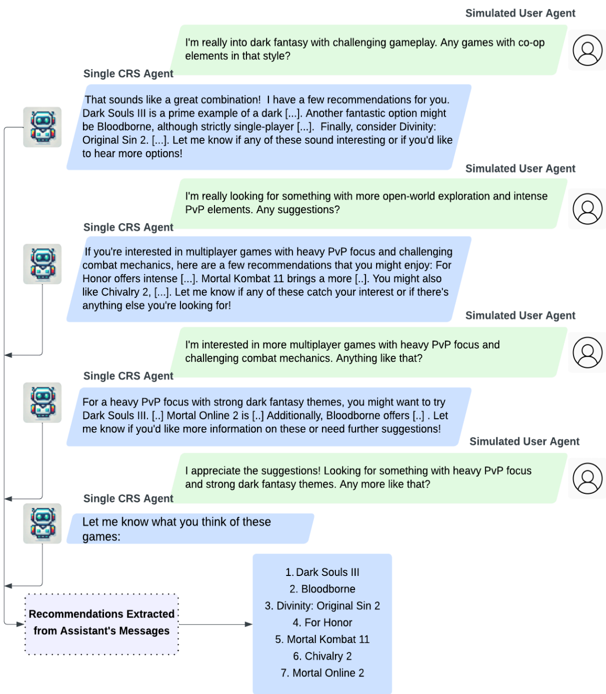

Fig. 3 Demonstration of a conversation flow using a Single Agent CRS

When comparing the two systems, one is clearly more capable, adaptable and explorative than the other. The first system generates recommendations on demand based on the user's immediate input with minimal follow-up. In contrast, the second system engaged the user in a more exploratory dialogue, eliciting detailed preferences through iterative questioning on specific gaming attributes, such as customization, developer preference, and co-op mechanics. The second system demonstrated greater adaptability and precision, yielding a more nuanced and user-tailored set of recommendations in most scenarios.

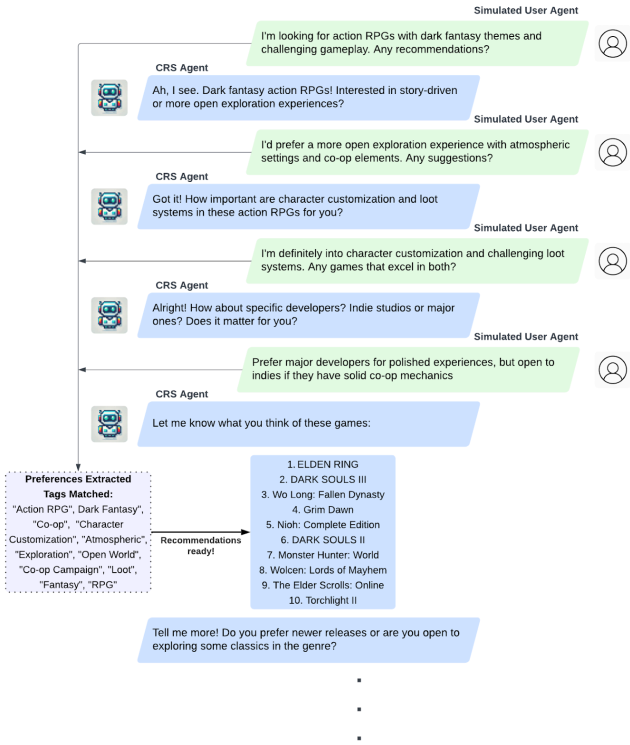

Fig. 4 Demonstration of a conversation flow using our Multi-Agent CRS

Even showing such impact, not only anecdotally but also statistically, it is reasonable to speculate and explore this problem with further evaluations using a more robust, reliable database for evaluating CRS systems. Evaluation is currently one of the hardest problems in this particular niche, considering the highly dynamic nature of iterative dialogue recommendations.

5. Conclusion

This research introduced a novel multi-agent Conversational Recommender System (CRS) designed to provide personalized video game recommendations in cold-start scenarios. By leveraging Large Language Models (LLMs) with chain-of-thought prompting and integrating semantic search techniques, our CRS framework effectively gathers user preferences through natural language conversations and delivers tailored game suggestions. The system's architecture, comprising specialized agents—Inquiry Agent, Extractor Agent, Retriever, and Re-rankers—demonstrates an innovative approach to addressing the challenges inherent in conversational recommendation systems.

The experimental evaluation of our CRS showed significant success in achieving its objectives while still leaving room for exploration and experimentation in the domain of evaluation systems for CRS. The multi-agent system built into this work could engage in meaningful dialogues with simulated users, extract relevant preferences, and provide game recommendations that, in most cases, matched the users' interests, even when the original target game was not provided. The integration of LLMs and semantic search contributed to the system's ability to process complex user inputs and refine recommendations based on a combination of categorical data and nuanced preferences, mitigating the degrees of freedom of the LLMs by pre-filtering items using actual category data.

Despite the promising aspects of the CRS, we encountered significant challenges in evaluating its performance. The reliance on synthetic user personas generated by LLMs introduced potential biases and limitations. Each persona was associated with only one game, restricting the diversity of user preferences and affecting the generalizability of the results. This approach, while necessary due to the lack of an existing end-to-end CRS dataset suitable for cold-start scenarios, impacts the validity of our evaluation and highlights the difficulties in assessing conversational recommender systems without real user interaction data.

The findings underscore the importance of robust evaluation methods in the development of CRS frameworks. The limitations encountered suggest that while synthetic data can provide initial insights, it may not fully capture the complexities of real-world user interactions. Consequently, our results indicate a need for caution in interpreting the system's effectiveness solely based on synthetic evaluations.

Looking forward, several avenues for future work can enhance the CRS and address the current limitations. Acquiring real user interaction data would significantly improve the evaluation's validity, providing a more accurate measure of the system's performance in practical applications. Developing more sophisticated user simulations, perhaps by associating personas with multiple games and incorporating a broader range of preferences, could also enrich the evaluation process. Additionally, implementing a dynamic conversation length or a router model to determine the optimal timing for recommendations may enhance user experience and system efficiency. Refinements to the semantic search and re-ranking mechanisms, as well as integrating some form of collaborative filtering, could further improve recommendation accuracy.

In conclusion, this research contributes to the field of conversational recommender systems by presenting a multi-agent framework that integrates LLMs and semantic search to address the cold-start problem in game recommendations. While challenges remain, particularly in evaluation, the approach lays a foundation for future research and development. The integration of advanced language models and semantic techniques holds significant promise for enhancing personalized recommendations in conversational settings. We encourage continued exploration and collaboration to overcome the identified challenges and advance the capabilities of conversational recommender systems, ultimately enriching user experiences across various domains.

References

[1] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in *Proc. 10th Int. Conf. World Wide Web (WWW '01)*, New York, NY, USA, 2001, pp. 285-295.

[2] Z. Dong, J. Zhu, W. Liu, and R. Tang, "Ten Challenges in Industrial Recommender Systems," *arXiv preprint arXiv: 2310.04804*, 2023.

[3] Gao, C., Lei, W., He, X., de Rijke, M., & Chua, T.-S. (2021). Advances and Challenges in Conversational Recommender Systems: A Survey. (v7 ed.) ArXiv. https://doi.org/10.48550/arXiv.2101.09459

[4] Z. He, Z. Xie, R. Jha, H. Steck, D. Liang, Y. Feng, B. P. Majumder, N. Kallus, and J. McAuley, "Large Language Models as Zero-Shot Conversational Recommenders," in *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management (CIKM '23)*, Birmingham, United Kingdom, Oct. 2023.

[5] A. Vats, V. Jain, R. Raja, and A. Chadha, "Exploring the Impact of Large Language Models on Recommender Systems: An Extensive Review," *arXiv preprint arXiv:2402.18590*, 2024.

[6] Christakopoulou, K., Radlinski, F., & Hofmann, K. (2016). "Towards Conversational Recommender Systems."

[7] Sun, Y., Zhang, C., Zhang, W., Liu, Y., Wang, C., Chen, J., & Wang, Y. (2018). "Conversational Recommender System." Proceedings of the 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, pp. 235-244.

[8] He, A., Chen, Y., & Li, B. (2023). "LLMs as Zero-Shot Conversational Recommender Systems." GitHub repository: https://github.com/AaronHeee/LLMs-as-Zero-Shot-Conversational-RecSys/

[9] Wang, J., Li, S., Zhang, Y., & Chen, X. (2023). "iEvaLM: An Interactive Evaluation Framework for LLM-based Conversational Recommender Systems." GitHub repository: https://github.com/RUCAIBox/iEvaLM-CRS

[10] Liu, Q., Zhao, H., & Sun, M. (2023). "LLM User Simulation for Detecting CRS Breakdowns." arXiv preprint arXiv:2305.14249.

[11] Zhang, L., Wang, T., & Zhou, J. (2023). "Item-Language Model for Item Encoding in LLM-based Conversational Recommender Systems." arXiv preprint arXiv:2306.02844.

[12] Chen, H., & Wang, Y. (2023). "An Overview of LLMs as Conversational Recommender Systems." arXiv preprint arXiv:2307.10081.

[13] Li, R., Zhang, S., Wang, W., Zhang, J., & Wu, L. (2020). "Conversational Recommendation: Formulation, Methods, and Evaluation." Information Sciences, 536, pp. 178-209.

[14] Silveira, Y., Solovykh, N., & Agell, N. (2021). "Conversational Recommender System for the Steam Video Game Platform." Proceedings of the 13th International Conference on Agents and Artificial Intelligence, pp. 350-356.