Re2LLM: Reflective Reinforcement Large Language Model for Session-based Recommendation

Ziyan Wang[1], Yingpeng Du[1*], Zhu Sun[2], Haoyan Chua[1], Kaidong Feng[2,3], Wenya Wang[1], Jie Zhang[1]

[1]Nanyang Technological University
[2]Singapore University of Technology and Design
[3]Yanshan University
wang1753@e.ntu.edu.sg, dyp1993@pku.edu.cn,sunzhuntu@gmail.com,
haoyan001@e.ntu.edu.sg,kaidong3762@gmail.com,wangwy@ntu.edu.sg,zhangj@ntu.edu.sg

Abstract

Emerging advancements in large language models (LLMs) show significant potential for enhancing recommendations. However, prompt-based methods often struggle to find ideal prompts without task-specific feedback, while fine-tuning-based methods are hindered by high computational demands and dependence on open-source backbones. To address these challenges, we propose a Reflective Reinforcement Large Language Model (Re2LLM) for session-based recommendation, which refines LLMs to generate and utilize specialized knowledge effectively and efficiently. Specifically, we first devise the Reflective Exploration Module to extract and present knowledge in a form that LLMs can easily process. This module enables LLMs to reflect on their recommendation mistakes and construct a hint knowledge base to rectify them effectively. Next, we design the Reinforcement Utilization Module to train a lightweight retrieval agent that elicits correct LLM reasoning. This module recognizes hints as signals to facilitate LLM recommendations and learns to select appropriate hints from the constructed knowledge base using task-specific feedback efficiently. Lastly, we conduct experiments on real-world datasets and demonstrate the superiority of our Re2LLM over state-of-the-art methods.

Introduction

Session-based recommendation (SBR) (Wu et al. 2019) aims to capture users' dynamic preferences based on their previous interactions within a session. However, the interactions are often sparse, and users' profiles are inaccessible in anonymous sessions. Recently, large language models (LLMs) have been applied in recommendation to address these issues with their extensive knowledge and sophisticated reasoning, primarily through two approaches: prompt-based and fine-tuning-based methods. The former methods exploit in-context learning and prompt optimization (Hou et al. 2024; Wang and Lim 2023; Sun et al. 2024) to engage LLMs as recommenders without training as in Figure 1(a). However, the crafted prompts not only require extensive expert knowledge and human labor but also may not align well with LLMs' understanding of SBR tasks, making LLMs vulnerable to hallucinations without effective supervision. The latter methods focus on fine-tuning LLMs (Geng et al. 2022; Yang et al. 2023; Bao et al. 2023) with domain-specific knowledge (e.g., user-item interactions) in a supervised manner as in Figure 1(b). Nonetheless, such methods often suffer from high computational costs and reliance on open-source backbones. These drawbacks limit the practical application of existing LLM-based methods for SBR.

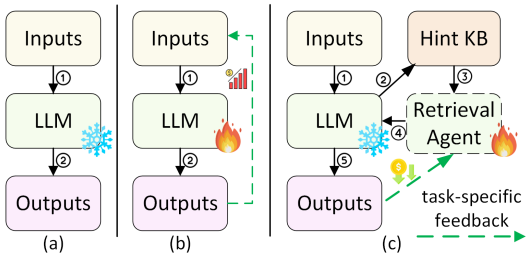

Figure 1: Our method (c) obtains effective task-specific feedback without costly fine-tuning in comparison to prompt-based (a) and fine-tuning-based (b) methods.

In this paper, we propose to direct LLMs to effectively and efficiently leverage specialized knowledge for SBR without costly fine-tuning, but there remain two primary challenges: (1) How can we effectively extract and craft specialized knowledge embedded within extensive user-item interactions to better align with LLM comprehension? (2) How can we enable LLMs to utilize the specialized knowledge efficiently for better recommendations, without relying on resource-intensive supervised fine-tuning? To overcome these challenges, we devise a Reflective Reinforcement Large Language Model (Re2LLM) for SBR. As illustrated in Figure 1(c), the LLM extracts specialized knowledge as hints into a knowledge base, and obtains effective task-specific feedback from the data without costly fine-tuning. This paradigm combines the strengths of LLMs' capabilities and the efficiency of lightweight retrieval. Re2LLM consists of two main components: the Reflective Exploration Module and the Reinforcement Utilization Module.

Specifically, ***Reflective Exploration Module*** leverages the self-reflection of LLMs (Madaan et al. 2023) to extract

*Corresponding author.

specialized knowledge understandable by LLMs. In detail, we employ LLMs to identify common errors in their SBR predictions, and then generate specialized knowledge (i.e., hints) to rectify these errors through self-reflections. Thus, the specialized knowledge can align seamlessly with LLM comprehension when summarized by the LLM itself. Besides, we construct a hint knowledge base that maintains the specialized knowledge using an automated filtering strategy to ensure its effectiveness and diversity. ***Reinforcement Utilization Module*** employs a lightweight retrieval agent to select relevant specialized knowledge instead of costly fine-tuning of LLMs. To overcome the lack of explicit labels or scores for hints, we train the agent through deep reinforcement learning (DRL). In detail, for each session, the agent action is to select hints from the hint knowledge base by perceiving the session's contextual information (i.e., observation state). Then, we measure the improvement (i.e., reward) of recommendation results owing to the selected hint, and use them (rewards, observation states, and actions) as the task-specific feedback to update the agent via Proximal Policy Optimization (PPO) (Schulman et al. 2017) strategy.

In summary, our key contributions are three-fold. **(1)** We propose a new learning paradigm beyond in-context learning and fine-tuning, which seamlessly bridges between general LLMs and specific recommendation tasks without costly supervised fine-tuning for LLMs. **(2)** Our method benefits from LLMs' self-reflection capabilities as well as the flexibility of the lightweight retrieval agent. This enables us to effectively extract specialized knowledge understandable by LLMs and efficiently utilize the knowledge by learning from task-specific feedback for better LLM-based SBR inference. **(3)** Our experiments demonstrate that Re2LLM outperforms state-of-the-art methods, including deep learning-based and LLM-based models, in both few-shot and full-data settings across two real-world datasets.

Literature Review

Session-based Recommendation. SBR methods model users' preferences through short and dynamic sessions. The classic FPMC (Rendle, Freudenthaler, and Schmidt-Thieme 2010) captures long-term user preference. With the development of deep learning, RNN (Hidasi et al. 2016), attention mechanism (Li et al. 2017; Liu et al. 2018), and GNN (Wu et al. 2019; Xu et al. 2019; Wang et al. 2020) have been introduced for modeling session interactions. Data augmentation (Tan, Xu, and Liu 2016) and novel ranking loss function (Hidasi and Karatzoglou 2018) are also proposed. Besides, many studies explore auxiliary information (e.g., attributes) (Hou et al. 2022; Xie, Zhou, and Kim 2022; Zhang et al. 2022; Chen et al. 2023) for better session profiling. MMSBR (Zhang et al. 2023c) leverages descriptive and numeric attributes to characterize intents.

Large Language Model for Recommendation. LLMs show potential in recommendation tasks with their extensive knowledge and strong reasoning capability. Prompt-based methods (Gao et al. 2023; Zhang et al. 2023a; Liu et al. 2024; Du et al. 2024; He et al. 2023; Dai et al. 2023) retrieve information from historical user-item interactions for direct outputs through prompt enhancement. LLMRank (Hou et al. 2024) and NIR (Wang and Lim 2023) are two representatives that utilize prompt templates to extract dynamic preferences for SBR. To enrich LLMs with task-specific supervision, either fully (Geng et al. 2022; Kang et al. 2023; Li et al. 2024) or parameter-efficient approaches (Bao et al. 2023; Ji et al. 2024; Yue et al. 2023) with LoRA (Hu et al. 2022) fine-tune LLMs for recommendation tasks. To merge the advantages of both sides, we employ a lightweight retrieval agent to select specialized knowledge summarized by LLMs.

Self-reflection in Large Language Models. LLMs can handle complex tasks through Chain-of-Thought (Wei et al. 2022) reasoning but are still prone to hallucination and incorrect reasoning (Pan et al. 2024). Self-reflection (Pryzant et al. 2023; Madaan et al. 2023; Yao et al. 2024), where LLMs iteratively refine outputs based on self-generated reviews without additional training, has been widely studied. However, LLMs' self-reflection in recommendation remains underexplored. To the best of our knowledge, DRDT (Wang et al. 2023) is the only work leveraging reflection for recommendation, but it uses a case-by-case reflection strategy, which lacks the specialized knowledge from a global view. Therefore, we propose maintaining a knowledge base to rectify common errors and enhance LLM reasoning for SBR.

Methodology

Problem Formulation. Let $\mathcal{V} = \{v_1, v_2, ..., v_N\}$ denote the item set with N items. Each session s with l interacted items is denoted as $s = \{v_1^s, v_2^s, ..., v_l^s\}$, where $v_i^s \in \mathcal{V}$. Besides, item side information (e.g., title) is available. $LLM(P_j)$ means the output of the LLM backbone given prompt P_j. Unlike traditional recommendation tasks, users' profiles and historical behaviors on other sessions are inaccessible in SBR, and its task is to predict the next item v_{l+1}^s that the user is likely to interact with based on session history s.

Model Overview. The overall architecture of our proposed Re2LLM approach is illustrated in Figure 2. Firstly, the Reflective Exploration Module facilitates the self-reflection of LLMs in identifying and summarizing their inference errors for SBR, thereby generating hints as specialized knowledge to avoid these potential errors. These hints are further maintained by the automated filtering strategy to ensure effectiveness and diversity and are stored in a knowledge base. Secondly, the Reinforcement Utilization Module employs DRL to train a lightweight retrieval agent based on task-specific feedback without costly fine-tuning. The agent learns to select relevant hints to guide LLMs in mitigating potential errors in SBR inference. Finally, the LLM conducts inference to deliver recommendations enhanced by the selected hints.

Reflective Exploration Module

To achieve accurate inference, LLMs need domain-specific knowledge for recommendation tasks. However, the main challenge is that the domain-specific knowledge is often embedded in the massive user-item interaction records, which may not align well with the comprehension capabilities of LLMs. Thus, it is essential to extract the specialized knowledge that is accessible to LLMs for recommendation

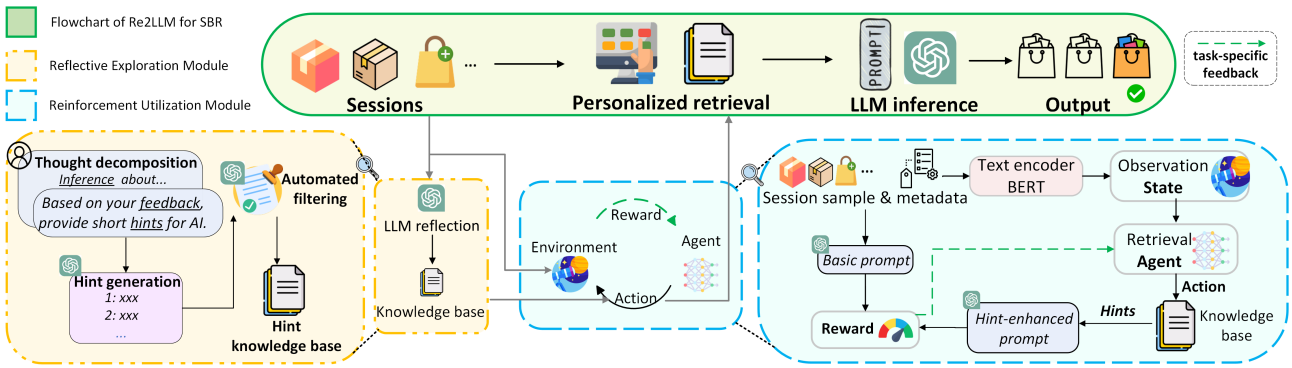

Figure 2: The overall architecture of our proposed Re2LLM approach. The overall flowchart (green) contains a Reflective Exploration Module (yellow) for the generation of session-aware hints as specialized knowledge and a Reinforcement Utilization Module (blue) for the learning to retrieve obtained specialized knowledge.

tasks. Therefore, we propose to leverage LLMs' strong self-reflection capability for the extraction of specialized knowledge as LLMs can understand the knowledge summarized by themselves more easily. Additionally, we use an automated filtering strategy to construct a knowledge base with effective and diverse hints for further utilization.

Self-Reflection Generation. To activate the self-reflection ability of LLMs, we propose to identify errors in LLMs' inference by comparing their predictions against the ground truth. Specifically, we first instruct the LLM to generate a ranking list $O(s)$ from a candidate set $\mathcal{C}$ containing $|\mathcal{C}|$ items based on current session s:

$$O(s) = LLM(P_1(s, \mathcal{C})), O \subseteq \mathcal{C}, \tag{1}$$

where P_1 is the *basic prompt* consisting of titles of interacted items in the session s, candidate set $\mathcal{C}$, and task instruction. The details are shown in Prompt 1 below.

Prompt 1 (basic prompt) *I watched/purchased the following movies/items in order: $\{s\}$. Based on these interactions, recommend a movie/item for me to watch/purchase next from a candidate set: $\{\mathcal{C}\}$. Please recommend from the candidate set. List top 10 recommendations in numbered bullet points.*

Then, we identify the incorrectly predicted sessions where the ranking list fails to hit the ground truth item at time step $l+1$ (i.e., $v^s_{l+1} \notin O(s)$). Afterward, we prompt the LLM to conduct self-reflection, that is, analyze and summarize the prediction errors, to generate potential hints in natural language to rectify such errors. These hints thus can be considered as specialized knowledge for better recommendations. Inspired by Chain-of-Thought prompting (Wei et al. 2022), our approach focuses on analyzing and alleviating these errors in a step-by-step and progressive reasoning chain, producing hints that are tailored to be compatible with LLM comprehension capabilities.

Particularly, LLMs can yield incorrect results in SBR due to various causes, which underscores the importance of having diverse and accurate hints to rectify the errors. To broaden the exploration of the diverse causes for the errors, we first only present the LLM with incorrectly predicted sessions and make it conduct self-reflection to identify possible causes for the errors without the ground truth item, as shown in Prompt 2. This allows the LLM to explore more diverse causes that may lead to errors. Then, to pursue more accurate causes for the errors, we further stimulate the LLM's analytical skills by revealing the ground truth item. Specifically, given the ground truth item, we ask the LLM to review previous diverse causes to select more relative ones, as shown in Prompt 3, thus identifying accurate causes of mistakes. Finally, for effective utilization in the next stage, we ask the LLM to summarize obtained causes as hints (known as specialized knowledge) in Prompt 4, which are more understandable for LLMs and easily improve the recommendation outcome. By following the provided reasoning chain (i.e., Prompts 2-4), the LLM can identify frequent and prominent errors in SBR inference, and then produce qualified hints to address potential errors effectively.

Prompt 2 *Question: $\{P_1\}$ (basic prompt). \n\n ChatGPT: $\{O(s)\}$ (top-10 results: 1. Casino Royale 2. Batman 3...). \n\n Now, know that none of these answers is the target. Infer about possible mistakes in the ChatGPT's predictions.*

Prompt 3 *The correct answer is $\{v^s_{l+1}\}$ (target). \n\n Analyze why ChatGPT missed the target item from a wide variety of aspects. Explain the causes for the oversights based on earlier analysis.*

Prompt 4 *Provide short hints (imperative sentences) for AI to try again according to each point of the previous step, without information leakage of the target item.*

Automated Filtering. Recognizing that no single hint can rectify all the errors by LLMs, we aim to build a hint knowledge base $\mathcal{H}$ to store the most effective hints for further utilization. We develop an automated filtering strategy to maintain the hint knowledge base by two key qualifications: effectiveness and non-redundancy.

For effectiveness, we only add a new hint to $\mathcal{H}$ if it can lead to performance improvement. Specifically, we first construct a *hint-enhanced prompt* P_5 (Prompt 5) to trigger the LLM for recommendation inference:

$$O^*(s) = LLM(P_5(s, \mathcal{C}, h)), \tag{2}$$

where $O^*(s)$ is the recommendation list obtained by the hint-enhanced prompt P_5. Then, we compare $O^*(s)$ with $O(s)$ obtained by Equation 1 using the basic prompt without

hint enhancement. The hint h is effective if it leads to a better prediction by the LLM, i.e., $v^s_{l+1} \in O^*(s) \& v^s_{l+1} \notin O(s)$.

Prompt 5 (hint-enhanced prompt) *I watched/purchased the following movies/items in order: $\{s\}$. Based on these interactions, recommend a movie/item for me to watch/purchase next from a candidate set: $\{\mathcal{C}\}$. Please recommend from the candidate set. List the top 10 recommendations in numbered bullet points.* ***Hint:*** $\{h\}$.

Meanwhile, generated hints can be redundant as multiple errors may have similar causes. This redundancy may result in a high cost of the maintenance and utilization of the hint knowledge base $\mathcal{H}$, e.g., larger size of the hint knowledge base and increased complexity in retrieving the targeted hint. To reduce the redundancy in the hint knowledge base $\mathcal{H}$, we only incorporate new hints that are distinct from existing ones. Specifically, we employ LLMs to detect the semantic similarity between the candidate hint and existing ones,

$$\sum\nolimits_{h \in \mathcal{H}} LLM(P_6(h', h)) == 0, \tag{3}$$

where the detail of P_6 is in Prompt 6. By this, only distinct candidate hint h' will be incorporated into the knowledge base $\mathcal{H}$ if it satisfies Equation 3.

Prompt 6 *'Does hint [h'] convey similar ideas to any of [h]? Return 1 if true, else return 0.'*

We iteratively update the hint knowledge base with qualified hints to correct the errors across different session patterns and thus improve recommendation results. The example outputs and hints are shown in Appendix A.

Reinforcement Utilization Module

To guide LLMs to infer more accurate SBR predictions, we propose to utilize the constructed hint knowledge base with proper selection to prevent errors in future LLM inference.

The absence of explicit labels on the hints' efficacy leads to a challenge in conducting supervision on hint selection, as computing the efficacy for all hints on each sample is costly. Fortunately, DRL with a replay buffer enables us to collect reward signals (i.e., hint efficacy) and update the network spontaneously to speed up. To this end, we employ the Proximal Policy Optimization (PPO) (Schulman et al. 2017) algorithm in DRL to ensure stable and efficient training of our retrieval agent. Specifically, our agent is trained to select the most relevant hints based on session-related context information, thereby preventing LLMs from similar reasoning mistakes for recommendation. In addition, the proposed DRL framework allows for the balance between exploitation and exploration. It is also compatible with more complex extensions such as the retrieval of multiple hints and multi-round agent-based interactions.

Markov Decision Process (MDP). We outline the basics of our DRL environment as the tuple $(\mathcal{Z}, \mathcal{A}, R, T)$, denoting state space, action space, reward function, and transition.

State. To model the session-related context information, we concatenate item titles and attributes into a text string and convert it into embedding for semantic extraction. We use pre-trained BERT (Devlin et al. 2019) as the text encoder due to its robust contextual language understanding capabilities. The state can be denoted as $\mathbf{z_s} = BERT(s)$, where $\mathbf{z_s} \in \mathcal{Z}$ is the d-dimensional output of the text encoder.

Action. To select relevant hints from the constructed knowledge base for LLMs' inference, we define a discrete action space for the agent, represented as $\mathbf{a} \in \mathcal{A}$, which is a $(|\mathcal{H}| + 1)$-dimensional vector. Here, $|\mathcal{H}|$ is the size of the knowledge base, and an action corresponds to either choosing a hint or opting not to select any.

Reward. To direct the agent in making accurate hint selections, we employ a comparative function R to provide the reward signals for the agent's actions. This function evaluates the improvement in the LLM prediction accuracy by the hint-enhanced prompt (Prompt 5) versus the basic prompt (Prompt 1). For each step, the reward value r is denoted as $r = m(O^*(s)) - m(O(s))$, where m is a recommendation evaluation metric (e.g., NDCG@10).

Transition. As the bandit problem in reinforcement learning, the environment produces a reward signal by the agent's action and then transits to the state of next session s', denoted as $T(\mathbf{z_s}) = \mathbf{z_{s'}}$. This setup can flexibly accommodate the scenario where multiple steps per session are involved for more sophisticated learning processes in our future studies.

Replay buffer. To facilitate efficiency in policy optimization, we maintain a replay buffer $D = (\mathbf{z_s}, \mathbf{a}, r, \mathbf{z_{s'}})$ to store the tuples of observation state, agent action, reward, and next observation state. With the records in the replay buffer, the retrieval agent can refine successful strategies and learn from erroneous trials. This technique significantly accelerates the DRL training as the LLM backbone is relatively slow for inference and producing reward signals.

PPO Training. To model the actions of the retrieval agent, we implement a policy network parameterized by MLPs to define our agent's policy $\pi_{\boldsymbol{\theta}}$, which maps the environmental space $\mathcal{Z}$ to the action space $\mathcal{A}$:

$$\mathbf{a} = softmax(\mathbf{W} \cdot \mathbf{z_s}), \tag{4}$$

where $\mathbf{W} \in \mathbb{R}^{(|H|+1) \times d}$ is the learnable weight matrix, and $\mathbf{z_s} \in \mathbb{R}^d$ is the current session state. The action corresponds to the largest value among the softmax logits of $\mathbf{a}$. During training, the retrieval agent adopts the ϵ-greedy ($\epsilon = 0.05$) strategy to explore the environment, with probability ϵ to take a random action while $(1 - \epsilon)$ to exploit the learned policy $\pi_{\boldsymbol{\theta}}$. Our objective function for PPO training can be formulated to maximize the total reward, given by,

$$\mathcal{L}_{PPO} = \mathbb{E}\left[R(\mathbf{z}, \mathbf{a}) - \beta KL(\pi_{\boldsymbol{\theta}_{old}}, \pi_{\boldsymbol{\theta}})\right], \tag{5}$$

where the first term is the reward measured by recommendation tasks, and the second term is the KL-divergence of policies with a coefficient β for policy updates. In this way, we train a retrieval agent with task-specific feedback via DRL for appropriate hint-enhanced prompts to improve LLM's recommendation performance.

Retrieval-Enhanced LLMs for Recommendation. With the constructed hint knowledge base and the trained retrieval agent, we achieve more accurate recommendations by automatically selecting appropriate hints for prompt enhancement. The recommendation output $\hat{O}$ is derived as follows:

$$\hat{O} = LLM(P^*(s, \mathcal{C}, \pi_{\boldsymbol{\theta}}(\mathbf{z}_s))), \tag{6}$$

where $\pi_{\boldsymbol{\theta}}(\mathbf{z_s})$ corresponds to the selected hint h by the trained retrieval agent for session s with its state $\mathbf{z}_s$. The overall pseudocode of Re2LLM is in Appendix B.

	Movie	Game
# Sessions	468,389	387,906
# Items	8,233	22,576
Avg. length	10.17	6.28
Utilized Item Side Info	title, genre, actor, year, country, director	title, category, tag, brand, description

Table 1: Statistics of datasets.

Complexity Analysis

The space complexity of Re2LLM is mainly attributed to the lightweight retrieval model implemented by MLP, that is, $O(df + pf^2)$ where d is the embedding dimension of session contexts, f is the hidden dimension, and p is the number of hidden layers. The time complexity of the Reflective Exploration Module is $\mathcal{O}(|S_{train}|T_1 + qT_2)$, where $|S_{train}|$ is the few-shot training size, q is the number of incorrectly predicted sessions required for the construction of $\mathcal{H}$ ($q \ll |S_{train}|$ in practice), T_1 is the inference time of LLM per session for SBR (Prompt 1 or 5), and T_2 is the inference time of LLM for reflection and automated filtering w.r.t. per incorrectly predicted session (Prompt 2-6). The time complexity of the Reinforcement Utilization Module is $O(w|S_{train}|T_1)$, where w is the DRL training epochs. The time complexity of Re2LLM inference is $O(|S_{test}|T_1)$, where $|S_{test}|$ is the size of test set. In summary, the complexity of Re2LLM is linear with $|S_{train}|$, $|S_{test}|$ and T_1, thus being scalable in real-world applications.

Experiments

In this section, we evaluate the effectiveness of the proposed method Re2LLM through comprehensive experiments and analysis. We aim to answer three research questions. **RQ1**: Does Re2LLM outperform deep learning and LLM-based baseline methods for SBR? **RQ2**: How do key components affect Re2LLM? **RQ3**: How do key hyper-parameters impact the performance of Re2LLM? Our code and data are available in the Supplementary Material.

Experimental Setup

Datasets We evaluate Re2LLM and baselines on two real-world datasets. *Movie* (Hetrec2011-Movielens) contains user ratings of movies and side information such as title, production year, and genre. *Game* ('Video Games' of the Amazon Review Dataset) contains users' reviews on various types of games and peripherals, and metadata such as title, brand, and tag. The statistics are in Table 1. We take user interactions within one day as a session sorted by the recorded timestamps and filter out sessions and items with less than 3 records. We do data augmentation for baselines by extending a session s with length $|s|$ to $(|s| - 1)$ sessions as training samples following (Tan, Xu, and Liu 2016).

Baselines We compare our proposed method with eight state-of-the-art deep learning and LLM-based methods as baselines. **FPMC** (Rendle, Freudenthaler, and Schmidt-Thieme 2010) combines matrix factorization with the first-order Markov chain. **GRU4Rec** (Hidasi et al. 2016) uses Gated Recurrent Unit (GRU), a typical RNN layer, to model whole sessions. **NARM** (Li et al. 2017) introduces the attention mechanism into RNNs to better model the sequential behavior. **SRGNN** (Wu et al. 2019) constructs session graphs and employs GNN layers to obtain embeddings. **GCEGNN** (Wang et al. 2020) further includes global-level information as an additional graph to enhance the model performance. **AttenMixer** (Zhang et al. 2023b) achieves multi-level reasoning over item transitions by the attention-based readout method. **LLMRank** (Hou et al. 2024) demonstrates the promising zero-shot inference ability of LLMs through recency-focused prompting and in-context learning. **NIR** (Wang and Lim 2023) employs a multi-step prompt to capture user preferences, select representative items, and finally perform the next-item prediction.

We introduce two settings for the evaluation of baseline methods. In the *full dataset setting*, baseline methods are trained on the entire dataset with augmentation using item IDs. In the *few-shot setting*, we examine whether our Re2LLM shows superior performance compared to the baselines trained with the same amount of data. For a fair comparison, we also incorporate the same item attributes used by Re2LLM for deep learning-based baselines as indicated by the '-attr' suffix. We aggregate the ID embedding and the embedding of item attributes as the item representation.

Evaluation Strategy and Metrics For each dataset, we apply the split-by-ratio strategy following (Sun et al. 2020) to obtain training, validation, and test sets by 7:1:2. We set the last item in each session as the target. We sample 49 negative items for each positive item (i.e., target) to form a candidate set with 50 items. We adopt two commonly used metrics for evaluation, normalized discounted cumulative gain ($NDCG@k$) and hit rate ($HR@k$), with $k = \{5, 10\}$.

Implementation Details Following (Sun et al. 2022), we adopt the Optuna (optuna.org) framework to optimize the hyperparameter of all methods efficiently. We conduct 20 trials to search for *learning rate*, *weight decay*, and *batch size*. For the rest hyper-parameters in baseline methods, we follow the suggested settings in the original papers. For our method Re2LLM, we set the knowledge base size to 20, and the few-shot training size to 500. For the full dataset setting, we use the entire training set. For the few-shot setting, we sample 500 training samples from the entire training set for all methods. We run 5 experiments with different random seeds and show the average performance. All methods are optimized by the Adam optimizer. We use the 'gpt-4' API as the backbone model. The average cost for each sample is around USD 0.02 during the hint generation, DRL training, and final evaluation. The average time for each LLM query is around 5.2 seconds. More details on implementation and computational requirements are in Appendix C.1 and C.2.

Overall Comparison (RQ1)

Table 2 shows the performance of our method Re2LLM and baseline methods under both full dataset setting and few-shot setting. The experimental results lead to several key conclusions. **First**, our Re2LLM significantly outperforms baselines in few-shot settings, with average improve-

Setting	Model	Movie HR@5	Movie HR@10	Movie NDCG@5	Movie NDCG@10	Game HR@5	Game HR@10	Game NDCG@5	Game NDCG@10
full dataset	FPMC	0.1492	0.2885	0.0922	0.1749	0.2463	0.3617	0.1788	0.2110
	GRU4Rec	0.2516	0.3861	0.1564	0.2103	0.2915	0.4107	0.2004	0.2392
	NARM	0.3867	0.5674	0.2605	0.3319	0.4743	0.5635	0.3732	0.4021
	SRGNN	0.3879	0.5720	0.2676	0.3342	0.4864	0.5335	0.3786	0.4128
	GCEGNN	0.3962	0.5895	0.2659	0.3281	0.4826	0.5523	0.4002	0.4332
	AttenMixer	0.3880	0.5727	0.2693	0.3472	0.4885	0.5838	0.3911	0.4226
	Imp.	4.14%*	0.14%	5.42%*	-3.39%	15.95%*	22.87%*	4.82%*	8.24%*
few-shot	NARM	0.2701	0.4412	0.1707	0.2298	0.1891	0.2765	0.1310	0.1586
	NARM-attr	0.3108	0.4835	0.1920	0.2481	0.1977	0.2915	0.1372	0.1672
	GCEGNN	0.3227	0.4813	0.2080	0.2584	0.1905	0.2798	0.1362	0.1659
	GCEGNN-attr	0.3254	0.4893	0.2136	0.2678	0.2019	0.3992	0.1408	0.1710
	AttenMixer	0.2774	0.4668	0.1774	0.2377	0.1899	0.2925	0.1357	0.1581
	AttenMixer-attr	0.3397	0.5056	0.2223	0.2795	0.2084	0.3121	0.1371	0.1727
	LLMRank	0.3649	0.5110	0.2436	0.2784	0.4947	0.6636	0.3898	0.4531
	NIR	0.3587	0.4901	0.2214	0.2648	0.5291	0.6434	0.4016	0.4346
	Re2LLM(Ours)	**0.4126**	**0.5735**	**0.2839**	**0.3358**	**0.5664**	**0.7173**	**0.4195**	**0.4689**
	Imp.	13.07%†	12.23%†	16.54%†	20.14%†	7.05%†	8.09%†	4.46%†	3.49%†

Table 2: Performance of all methods under two settings. The performance of our Re2LLM is in bold; the best performance achieved by baselines in full and few-shot settings are underlined with '_' and '_', respectively. The improvements (Imp.) of our model against the best-performed baselines in both settings are reported, where * and † indicate statistically significant improvement by t-test ($p<0.05$) for full* and few-shot† settings, respectively.

ments of 15.49% and 5.77% across all metrics when compared to the top-performing baseline models on Movie and Game, respectively. This is mainly attributed to the effectiveness of Re2LLM, which not only extracts specialized and comprehensible knowledge by self-reflection but also effectively utilizes the knowledge based on a well-trained retrieval agent for better recommendations. **Second**, Re2LLM (without data augmentation) achieves comparable and even better performance than baseline methods trained on full datasets (with data augmentation), which also verifies the effectiveness of our module design. Furthermore, the LLM-based methods (e.g., LLMRank and NIR) also achieve comparable performance to baseline methods trained on full datasets, which indicates the promising way of employing LLMs as recommenders. **Third**, deep learning-based models show a significant performance drop in the few-shot setting compared with the full dataset setting, pointing to the data sparsity issue as a primary challenge. **Last**, the incorporation of item attributes slightly boosts the performance of baseline methods, showing the potential of using side information of items in addressing the sparsity issue in SBR.

Ablation Studies (RQ2)

Table 3 shows the performance of all ablation studies. **First**, Re2LLM outperforms its variant w/o-HE, where we use basic prompt instead of Hint-Enhanced prompt. This indicates the necessity of activating the LLM's self-reflection mechanism with hint enhancement in our Reflective Exploration Module. **Second**, Re2LLM outperforms variants RAN and ALL, where we randomly sample hints from the hint knowledge base for prompt enhancement (RAN) or concatenate all hints in the knowledge base (ALL) instead of the learned selective retrieval (w/o-DRL). This proves the importance of our Reinforcement Utilization Module for training the retrieval agent with task-specific feedback. The variant RAN has the worst performance by random hints because inaccurate hints can mislead the LLM inference, resulting in negative impacts on recommendation. The variant ALL shows limited improvement compared to the variant w/o-HE, revealing that simply providing all hints into LLMs without session-aware selection is suboptimal. **Third**, we skip the Automated Filtering mechanism from the hint knowledge base construction. The variant w/o-AF achieves relatively better performance among all variants but is still worse than our Re2LLM. This is because the automated filtering mechanism can enhance the quality of extracted knowledge. **In summary**, the above results exhibit the efficacy of different modules in the proposed Re2LLM for more accurate SBR.

Models	Movie HR@5	Movie HR@10	Movie NDCG@5	Movie NDCG@10
w/o-HE	0.3923	0.5458	0.2704	0.3198
w/o-AF	0.4013	0.5592	0.2774	0.3289
RAN(w/o-DRL)	0.3542	0.4960	0.2446	0.2908
ALL(w/o-DRL)	0.3910	0.5434	0.2746	0.3235
Re2LLM	**0.4126**	**0.5735**	**0.2839**	**0.3358**
Models	**Game** HR@5	**Game** HR@10	**Game** NDCG@5	**Game** NDCG@10
w/o-HE	0.5462	0.6771	0.4182	0.4550
w/o-AF	0.5652	0.6923	0.4146	0.4602
RAN(w/o-DRL)	0.5072	0.6345	0.3722	0.4107
ALL(w/o-DRL)	0.5638	0.6897	0.4164	0.4592
Re2LLM	**0.5664**	**0.7173**	**0.4195**	**0.4689**

Table 3: The results of ablation study across all datasets.

Case Study (RQ2)

Figure 3 illustrates the case study where the LLM backbone fails to hit the target item without a hint (i.e., the variant w/o-HE). Our trained retrieval agent in Re2LLM successfully retrieves the tailored hint and hits the target item (Aladdin) at

Case Study 1: *Trained retriever*

Prompt: *I watched the following movies in order: {Titanic; Leaving Las Vegas...}. Based on these interactions, recommend a movie for me to watch next from a candidate set: {C}. Please recommend from the candidate set. List the top 10 recommendations in numbered bullet points. Hint: {h}.*

Answer by variant w/o-HE, *where* h = '': *1. The Day After Tomorrow; 2. The Bourne Identity; 3. Last Man Standing; 4. The Last Castle; 5. Resident Evil: Extinction; 6. Deep Blue Sea; 7. True Grit; 8. Courage Under Fire; 9. Kiss of the Dragon; 10. The Wrestler*

Answer by Re2LLM, *where* ***retrieved hint*** h = ***'Consider films that originated from other media, such as books, radio, or TV shows'***: *1. The Bourne Identity; 2. True Grit; 3. Aladdin; 4. The Parent Trap; 5. The Picture of Dorian Gray; 6. Breakfast at Tiffany's; 7. The Jazz Singer; 8. The Last Castle; 9. The Stepfather; 10. Cujo*

Answer by w/o-DRL(RAN), *where* ***random hint*** h = ***'Pay attention to actors who appear in multiple movies from the watched list and consider other films featuring these actors'***: *1. The Day After Tomorrow; 2. Taken; 3. The Bourne Identity; 4. Last Man Standing; 5. The Last Castle; 6. The Wrestler; 7. Redbelt; 8. The Stepfather; 9. Kiss of the Dragon; 10. Courage Under Fire*

Figure 3: Case study of Re2LLM.

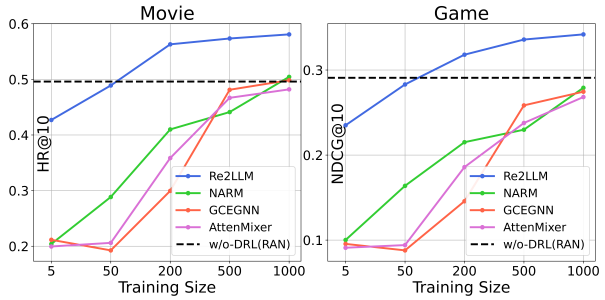

Figure 4: Impacts of few-shot training size. The dotted lines indicate RAN(w/o-DRL) performance.

the third place of the top-10 recommendation list, demonstrating its ability to learn from task-specific feedback. On the contrary, the variant RAN fails to replicate the correct prediction as a random hint may not be relevant to the session. In addition, when the LLM backbone already succeeds in hitting the target item without a hint, the trained retrieval agent can further improve recommendation results by hitting the target item in a higher-ranking position (refer to Appendix C.3.). On the contrary, a randomly selected hint can mislead the LLM and make it miss the target item. As a result, it is necessary to explore and utilize specialized knowledge (i.e., hints) properly for LLMs in SBR. The case study validates the efficacy of key modules in Re2LLM.

Hyper-parameter Analysis (RQ3)

Figure 4 shows the model performance under different few-shot training sizes for the retrieval agent. **First**, too few samples (e.g., less than 50) for model training results in inferior performance because the retrieval agent has limited exploration. The performance is even lower than random hint selection (RAN). **Second**, as the number of training samples increases, the performance of our method increases consistently, showing the growth of the generalization ability of the retrieval agent. Baseline methods remain inferior to Re2LLM in the few-shot setting. **Third**, there is only marginal improvement of Re2LLM when the number of samples exceeds a threshold, suggesting a trade-off between performance and efficiency.

We investigate the correlation between model performance and the size of the knowledge base constructed by the Reflective Exploration Module. As shown in Figure 5,

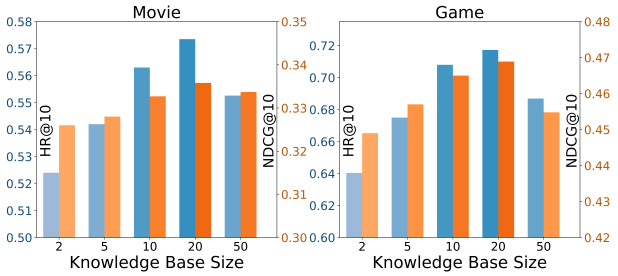

Figure 5: Impacts of the knowledge base size.

Reward	Movie		Game	
	HR@10	NDCG@10	HR@10	NDCG@10
absolute	0.5143	0.3147	0.7026	0.4542
comparative	**0.5735**	**0.3358**	**0.7173**	**0.4689**

Table 4: Performance of Re2LLM with two reward designs.

there is an improvement in both evaluation metrics as the size of the knowledge base increases from 2 to 20. The improvement is attributed to the diversity of the generated reflective hints by covering a broader range of common errors of LLM inference on SBR. However, when the knowledge base size becomes too large (e.g., 50), there is a slight performance drop due to the increased complexity and difficulty in the retrieval agent optimization. Thus, we suggest adopting a moderate knowledge base size.

Reward Design Table 4 reports the performance using two distinct reward designs for the Reinforcement Retrieval Module. The comparative reward $R = m(O^*(s)) - m(O(s)))$ measures the improvement achieved by the hint-enhanced prompt over the basic prompt. The absolute reward $R^{'} = m(O^*(s))$ evaluates the performance of the hint-enhanced prompt alone. We find that the comparative reward yields superior performance. This is because the comparative strategy focuses on improvement owing to the selected hint rather than its overall performance. As a result, we adopt the comparative reward for our Re2LLM.

Conclusion

In this paper, we propose Re2LLM, a Reflective Reinforcement LLM for SBR, aiming to improve performance by identifying and rectifying common errors in LLM inference. We present a novel learning paradigm that merges the capabilities of LLMs with efficient model training procedures. Specifically, Re2LLM harnesses the self-reflection ability of LLMs to capture specialized knowledge and create a hint knowledge base with an automated filtering strategy. Then, trained via DRL, a lightweight retrieval agent learns to select proper hints guided by task-specific feedback to facilitate session-aware inference for better recommendation results. We demonstrate the effectiveness of our method through extensive experiments in two real-world datasets across two evaluation settings. Future work could extend the retrieval agent with more flexible functionalities, such as integrating multiple hints and multi-modal contextual information.

Acknowledgements

This research is supported by the Agency for Science, Technology and Research (A*STAR) under its RIE 2025 – Industry Alignment Fund – Pre Positioning (IAF-PP) funding scheme (Project No: M23L4a0001). This research is also partially supported by the MOE AcRF Tier 1 funding (RG13/23). This research is also partially supported by the Ministry of Education, Singapore, under its MOE AcRF Tier 1, SUTD Kickstarter Initiative (SKI 2021_06_12).

References

Bao, K.; Zhang, J.; Zhang, Y.; Wang, W.; Feng, F.; and He, X. 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 (RecSys)*, 1007–1014.

Chen, Q.; Guo, Z.; Li, J.; and Li, G. 2023. Knowledge-enhanced Multi-View Graph Neural Networks for Session-based Recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, 352–361.

Dai, S.; Shao, N.; Zhao, H.; Yu, W.; Si, Z.; Xu, C.; Sun, Z.; Zhang, X.; and Xu, J. 2023. Uncovering ChatGPT's Capabilities in Recommender Systems. In *Proceedings of the 17th ACM Conference on Recommender Systems (RecSys)*, 1126–1132.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 4171–4186.

Du, Y.; Luo, D.; Yan, R.; Wang, X.; Liu, H.; Zhu, H.; Song, Y.; and Zhang, J. 2024. Enhancing Job Recommendation through LLM-Based Generative Adversarial Networks. *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 8363–8371.

Gao, Y.; Sheng, T.; Xiang, Y.; Xiong, Y.; Wang, H.; and Zhang, J. 2023. Chat-REC: Towards Interactive and Explainable LLMs-Augmented Recommender System. arXiv:2303.14524.

Geng, S.; Liu, S.; Fu, Z.; Ge, Y.; and Zhang, Y. 2022. Recommendation as Language Processing (RLP): A Unified Pretrain, Personalized Prompt & Predict Paradigm (P5). In *Proceedings of the 16th ACM Conference on Recommender Systems (RecSys)*, 299–315.

He, Z.; Xie, Z.; Jha, R.; Steck, H.; Liang, D.; Feng, Y.; Majumder, B. P.; Kallus, N.; and Mcauley, J. 2023. Large Language Models as Zero-Shot Conversational Recommenders. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management (CIKM)*, 720–730.

Hidasi, B.; and Karatzoglou, A. 2018. Recurrent Neural Networks with Top-k Gains for Session-based Recommendations. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management (CIKM)*, 843–852.

Hidasi, B.; Karatzoglou, A.; Baltrunas, L.; and Tikk, D. 2016. Session-based Recommendations with Recurrent Neural Networks. In *4th International Conference on Learning Representations (ICLR)*.

Hou, Y.; Mu, S.; Zhao, W. X.; Li, Y.; Ding, B.; and Wen, J.-R. 2022. Towards Universal Sequence Representation Learning for Recommender Systems. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, 585–593.

Hou, Y.; Zhang, J.; Lin, Z.; Lu, H.; Xie, R.; McAuley, J.; and Zhao, W. X. 2024. Large Language Models are Zero-Shot Rankers for Recommender Systems. In *European Conference on Information Retrieval (ECIR)*, 364–381.

Hu, E. J.; Shen, Y.; Wallis, P.; Allen-Zhu, Z.; Li, Y.; Wang, S.; Wang, L.; and Chen, W. 2022. LoRA: Low-Rank Adaptation of Large Language Models. In *International Conference on Learning Representations (ICLR)*.

Ji, J.; Li, Z.; Xu, S.; Hua, W.; Ge, Y.; Tan, J.; and Zhang, Y. 2024. GenRec: Large Language Model for Generative Recommendation. In *European Conference on Information Retrieval (ECIR)*, 494–502.

Kang, W.-C.; Ni, J.; Mehta, N.; Sathiamoorthy, M.; Hong, L.; Chi, E.; and Cheng, D. Z. 2023. Do LLMs Understand User Preferences? Evaluating LLMs On User Rating Prediction. arXiv:2305.06474.

Li, J.; Ren, P.; Chen, Z.; Ren, Z.; Lian, T.; and Ma, J. 2017. Neural Attentive Session-based Recommendation. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management (CIKM)*, 1419–1428.

Li, Y.; Du, H.; Ni, Y.; Zhao, P.; Guo, Q.; Yuan, F.; and Zhou, X. 2024. Multi-Modality is All You Need for Transferable Recommender Systems. In *2024 IEEE 40th International Conference on Data Engineering (ICDE)*, 5008–5021.

Liu, Q.; Chen, N.; Sakai, T.; and Wu, X.-M. 2024. ONCE: Boosting Content-based Recommendation with Both Open- and Closed-source Large Language Models. In *Proceedings of the Seventeen ACM International Conference on Web Search and Data Mining (WSDM)*, 452–461.

Liu, Q.; Zeng, Y.; Mokhosi, R.; and Zhang, H. 2018. STAMP: Short-Term Attention/Memory Priority Model for Session-based Recommendation. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD)*, 1831–1839.

Madaan, A.; Tandon, N.; Gupta, P.; Hallinan, S.; Gao, L.; Wiegreffe, S.; Alon, U.; Dziri, N.; Prabhumoye, S.; Yang, Y.; et al. 2023. Self-Refine: Iterative Refinement with Self-Feedback. *Advances in Neural Information Processing Systems (NeurIPS)*.

Pan, L.; Saxon, M.; Xu, W.; Nathani, D.; Wang, X.; and Wang, W. Y. 2024. Automatically Correcting Large Language Models: Surveying the Landscape of Diverse Automated Correction Strategies. *Transactions of the Association for Computational Linguistics (TACL)*, 484–506.

Pryzant, R.; Iter, D.; Li, J.; Lee, Y.; Zhu, C.; and Zeng, M. 2023. Automatic Prompt Optimization with "Gradient Descent" and Beam Search. In *Proceedings of the 2023 Confer-*

ence on Empirical Methods in Natural Language Processing (EMNLP), 7957–7968.

Rendle, S.; Freudenthaler, C.; and Schmidt-Thieme, L. 2010. Factorizing Personalized Markov Chains for Next-Basket Recommendation. In *Proceedings of the 19th International Conference on World Wide Web (WWW)*, 811–820.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal Policy Optimization Algorithms. arXiv:1707.06347.

Sun, Z.; Fang, H.; Yang, J.; Qu, X.; Liu, H.; Yu, D.; Ong, Y.-S.; and Zhang, J. 2022. DaisyRec 2.0: Benchmarking Recommendation for Rigorous Evaluation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 8206–8226.

Sun, Z.; Liu, H.; Qu, X.; Feng, K.; Wang, Y.; and Ong, Y. S. 2024. Large Language Models for Intent-Driven Session Recommendations. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, 324–334. NY, USA.

Sun, Z.; Yu, D.; Fang, H.; Yang, J.; Qu, X.; Zhang, J.; and Geng, C. 2020. Are We Evaluating Rigorously? Benchmarking Recommendation for Reproducible Evaluation and Fair Comparison. In *Proceedings of the 14th ACM Conference on Recommender Systems (RecSys)*.

Tan, Y. K.; Xu, X.; and Liu, Y. 2016. Improved Recurrent Neural Networks for Session-based Recommendations. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*, 17–22.

Wang, L.; and Lim, E.-P. 2023. Zero-Shot Next-Item Recommendation using Large Pretrained Language Models. arXiv:2304.03153.

Wang, Y.; Liu, Z.; Zhang, J.; Yao, W.; Heinecke, S.; and Yu, P. S. 2023. DRDT: Dynamic Reflection with Divergent Thinking for LLM-based Sequential Recommendation. arXiv:2312.11336.

Wang, Z.; Wei, W.; Cong, G.; Li, X.-L.; Mao, X.-L.; and Qiu, M. 2020. Global Context Enhanced Graph Neural Networks for Session-based Recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, 169–178.

Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; ichter, b.; Xia, F.; Chi, E.; Le, Q. V.; and Zhou, D. 2022. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 24824–24837.

Wu, S.; Tang, Y.; Zhu, Y.; Wang, L.; Xie, X.; and Tan, T. 2019. Session-based Recommendation with Graph Neural Networks. In *Proceedings of the Thirty-Third AAAI Conference (AAAI)*, 346–353.

Xie, Y.; Zhou, P.; and Kim, S. 2022. Decoupled Side Information Fusion for Sequential Recommendation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, 1611–1621.

Xu, C.; Zhao, P.; Liu, Y.; Sheng, V. S.; Xu, J.; Zhuang, F.; Fang, J.; and Zhou, X. 2019. Graph Contextualized Self-Attention Network for Session-based Recommendation. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI)*, 3940–3946.

Yang, F.; Chen, Z.; Jiang, Z.; Cho, E.; Huang, X.; and Lu, Y. 2023. PALR: Personalization Aware LLMs for Recommendation. arXiv:2305.07622.

Yao, W.; Heinecke, S.; Niebles, J. C.; Liu, Z.; Feng, Y.; Xue, L.; N, R. R.; Chen, Z.; Zhang, J.; Arpit, D.; Xu, R.; Mui, P. L.; Wang, H.; Xiong, C.; and Savarese, S. 2024. Retroformer: Retrospective Large Language Agents with Policy Gradient Optimization. In *The Twelfth International Conference on Learning Representations (ICLR)*.

Yue, Z.; Rabhi, S.; de Souza Pereira Moreira, G.; Wang, D.; and Oldridge, E. 2023. LlamaRec: Two-Stage Recommendation using Large Language Models for Ranking. arXiv:2311.02089.

Zhang, J.; Bao, K.; Zhang, Y.; Wang, W.; Feng, F.; and He, X. 2023a. Is ChatGPT Fair for Recommendation? Evaluating Fairness in Large Language Model Recommendation. In *Proceedings of the 17th ACM Conference on Recommender Systems (RecSys)*, 993–999.

Zhang, P.; Guo, J.; Li, C.; Xie, Y.; Kim, J. B.; Zhang, Y.; Xie, X.; Wang, H.; and Kim, S. 2023b. Efficiently Leveraging Multi-level User Intent for Session-based Recommendation via Atten-Mixer Network. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining (WSDM)*, 168–176.

Zhang, X.; Xu, B.; Ma, F.; Li, C.; Yang, L.; and Lin, H. 2023c. Beyond Co-Occurrence: Multi-Modal Session-Based Recommendation. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 1–12.

Zhang, X.; Xu, B.; Yang, L.; Li, C.; Ma, F.; Liu, H.; and Lin, H. 2022. Price DOES Matter! Modeling Price and Interest Preferences in Session-Based Recommendation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, 1684–1693.