Reinforced Prompt Personalization for Recommendation with Large Language Models

WENYU MAO, School of Cyber Science and Technology, University of Science and Technology of China, Hefei, China

JIANCAN WU, University of Science and Technology of China, Hefei, China

WEIJIAN CHEN, Hefei Comprehensive National Science Center, Hefei, China

CHONGMING GAO and XIANG WANG, University of Science and Technology of China, Hefei, China

XIANGNAN HE, School of Information Science and Technology, University of Science and Technology of China, Hefei, China

Designing effective prompts can empower LLMs to understand user preferences and provide recommendations with intent comprehension and knowledge utilization capabilities. Nevertheless, recent studies predominantly concentrate on task-wise prompting, developing fixed prompt templates shared across all users in a given recommendation task (e.g., rating or ranking). Although convenient, task-wise prompting overlooks individual user differences, leading to inaccurate analysis of user interests. In this work, we introduce the concept of instance-wise prompting, aiming at personalizing discrete prompts for individual users. Toward this end, we propose Reinforced Prompt Personalization (RPP) to realize it automatically. To improve efficiency and quality, RPP personalizes prompts at the sentence level rather than searching in the vast vocabulary word-by-word. Specifically, RPP breaks down the prompt into four patterns, tailoring patterns based on multi-agent and combining them. Then the personalized prompts interact with LLMs (environment) iteratively, to boost LLMs' recommending performance (reward). In addition to RPP, to improve the scalability of action space, our proposal of RPP+ dynamically refines the selected actions with LLMs throughout the iterative process. Extensive experiments on various datasets demonstrate the superiority of RPP/RPP+ over traditional recommender models, few-shot methods, and other prompt-based methods, underscoring the significance of instance-wise prompting in LLMs for recommendation. Our code is available at https://github.com/maowenyu-11/RPP.

CCS Concepts: • **Computing methodologies** → *Discrete space search*; **Multi-agent reinforcement learning**; • **Information systems** → **Recommender systems**;

Additional Key Words and Phrases: Prompt optimization, Recommendation systems, Reinforcement learning

1 Introduction

Recommender Systems (RSs) are widely applied in real life to offer personalized item recommendations that align with user preference. Recently, **Large Language Models (LLMs)**, such as ChatGPT, GPT4, PaLM-E, and LLaMA, have demonstrated exceptional capabilities in semantic understanding, intent reasoning, and knowledge utilization [64]. These advancements empower LLMs to address recommendation tasks by modeling users' behavior into natural language, attracting considerable attention [7, 11, 24].

To unleash the recommendation capabilities of LLMs, a promising solution lies in tailoring prompts for the specific recommendation task [11, 62], beyond fine-tuning the massive parameters [1] that can be resource-hungry. Existing methods [10, 16, 60] have investigated the *task-wise prompting*, dedicated to establishing fixed prompt templates shared across all users. To summarize, most of these task-wise prompts for recommendation tasks include the following patterns:

—*Role-playing* assigns a specific role for LLMs to play [21, 65] in the recommendation tasks, enabling LLMs to respond with domain knowledge aligned to that role.
—*History records* provides the history interaction sequence of the target user, which the LLMs can utilize as in-context information for subsequent recommendations [16, 25].
—*Reasoning guidance* guides LLMs with essential steps in the reasoning process to make recommendations, incorporating strategies such as **Chain-of-Thought (CoT)** [16, 56].
—*Output format* defines the desired format of the LLMs' output in the specific tasks, such as the scores for rating prediction [11] and order numbers for ranking tasks [16].

However, we argue that applying fixed task-wise prompts for all users [20] may fail to fully explore the capability of LLMs to provide personalized recommendations. First, users' varied intentions pose challenges for the one-fitting-all prompting to analyze dynamically, which may result in mediocre recommendation performance. Take Figure 1 as an example, the first user's preference for the science fiction film "The Fly" is reflected in the watching history of 10 movies from 2 weeks ago. In contrast, the second user's preference for the comedy film "Pink Flamingos" is evident in the watching history of the recent two films. Thus, task-wise prompts, such as instructing LLMs to recommend based on a fixed length of users' watching history may neglect users' dynamic preferences. Secondly, the performance of LLMs is sensitive to the expression of prompts [34]. Fixed prompt templates sacrifice the benefits that varying expressions of prompts can bring to the output of LLMs. Therefore, we propose personalizing *instance-wise prompts* for different users in the recommendation tasks.

While conceptually appealing, personalizing the discrete prompts for individual users is challenging. On the one hand, the methods of manual creation [2] or heuristical crafting [19]) are labor-intensive and resource-expensive to tailor prompts for each user. On the other hand, methods based on supervised learning [32, 44] are unreliable due to the absence of labels for optimal prompts. To bridge this gap, we propose personalizing prompts automatically with **Reinforcement Learning (RL)**, drawing inspiration from recent advances in optimizing prompts for **Natural Language Processing (NLP)** tasks [5, 63]. The advantage of RL [54] lies in its ability to learn autonomously through interactions with the environment, rather than depending on manual effort
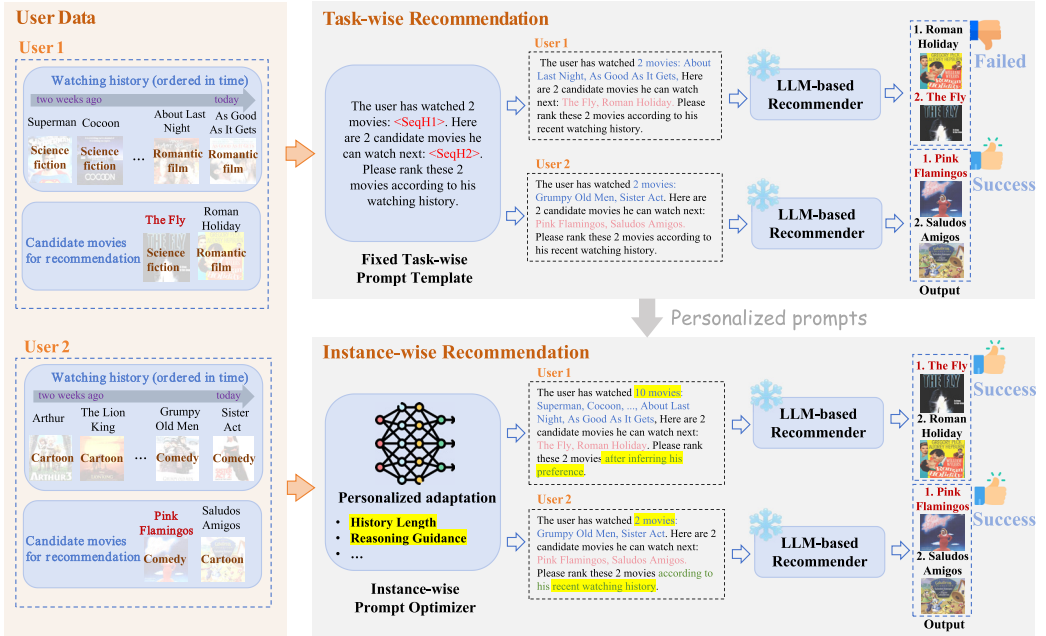
Fig. 1. Comparison between task-wise prompting and instance-wise prompting for recommendation. The ground truth for candidate movies that the users will watch next is marked in red. The personalized parts in prompts for different users are highlighted with yellow .

or labeled data. However, existing works which optimize prompts with RL in NLP tasks either search for tokens within an extensive vocabulary [5] or conduct basic editing operations [63] on the prompts, which are constrained by computational resource limitations and the reliability of the generated prompts. To apply RL for personalizing instance-wise prompts in the recommendation tasks, two fundamental problems are crucial.

—How to reduce the search space to enhance efficiency?
—How to ensure the quality of the personalized prompts?

To tackle these problems, we introduce **Reinforced Prompt Personalization (RPP)** to personalize high-quality prompts for each user efficiently. Instead of searching word-by-word in the extensive vocabulary [5], we limit the search space by exploring the various expressions of the four patterns at the sentence level, i.e., {*role-playing*, *history records*, *reasoning guidance*, *output format*}. We view the prompt optimization process as a combinatorial optimization problem with limited search space [23, 27–29]. To ensure prompt quality, we meticulously design the diverse expressions of each pattern based on the nature of recommendation tasks, which can guide LLMs to understand users' intent and make recommendations from multiple perspectives. We specify these high-quality expressions as actions, in contrast to simply treating different editing operations (such as "add" or "delete") as the actions [63]. The personalized prompts interact with LLMs which function as the environment, to maximize LLMs' recommending performance which serves as the reward. Specifically, to select sentences (actions) for the four patterns with specific expertise respectively, RPP is designed based on **Multi-Agent RL (MARL)** under **Centralized Training with Decentralized Execution (CTDE)** paradigm [33]. Under CTDE, each agent has an individual policy network based on Advantage Actor-Critic [39] algorithm, as personalizing each specific prompt pattern

requires distinct expertise. This global state contains information about the personalized prompts and the ranking results generated by LLMs in the current round. The action space is carefully designed with each action collected through collaborative deliberation between human experts and LLMs. Moreover, to improve the flexibility and quality of the action space, we dynamically refine the selected actions with LLMs during the iterative process to enhance scalability, referred to as RPP+.

We assess the effectiveness of our approach in ranking tasks across three widely used public datasets (i.e., **MovieLens-1M (ML-1M)** [12], Games [38], and Lastfm [3]). Experimental results demonstrate that RPP/RPP+ can significantly improve the recommendation performance of LLMs, outperforming several traditional recommender models (e.g., BPRMF [42], LightGCN [13], SASRec [22]), few-shot methods (e.g., VQ-Rec [14], UniSRec [15]), and other prompt-based methods (e.g., manual prompts [16], enumeration, and GRIPS [41]). This highlights the significance of tailoring prompts for each individual user and the superiority of RPP/RPP+ in enhancing the recommendation ability of LLMs. Our key contributions are as follows:

— We propose RPP/RPP+, a general prompt optimization framework to tailor instance-wise prompts for individual users.
— Our methodology generates personalized prompts at the sentence level with MARL, achieving the balance between search efficiency and prompt quality.
— Extensive experiments on three benchmark datasets demonstrate the superiority of RPP/RPP+, highlighting the necessity of personalizing instance-wise prompts.

2 Preliminaries

Here, we introduce task-wise prompting and instance-wise prompting in the recommendation tasks.

2.1 Task-Wise Prompting

Most LLM-based RSs frame recommendation as a language modeling task [4, 10, 53], with a fixed prompt template shared across all users in a task. The fixed task-wise prompts typically consist of four patterns, including "role-playing," "history records," "reasoning guidance," and "output format," as exemplified as follows:

Task-Wise Prompt Example

Role-Playing:
You are a movie expert.
History Records:
I've watched these movies <SeqH1> recently.
Reasoning Guidance:
Please rank these candidate movies <SeqH2> after inferring my preference from my watching history.
Output Format:
Please only output the ranking results with order numbers. Do not explain the reason or include any other words.

Here, <SeqH1> and <SeqH2> are the lists of movie titles from the watching history and candidate movies, respectively. Each sentence in the prompt template corresponding to a fixed pattern that remains unchanged regardless of the difference of users. The one-size-fits-all approach lacks

the flexibility to adapt the prompt to catch individual users' preferences, leading to inaccurate recommendation results.

2.2 Instance-Wise Prompting

To match the need for personalization, going beyond task-wise prompting, we introduce instance-wise prompting for recommendation, which tailors prompts for each user instance to elicit high-quality recommendations. Formally, for a user u, we take his/her historical interactions $\mathcal{H} = \{i_k\}_{k=1}^n$ paired with the candidate items $C = \{i_j\}_{j=1}^m$ as the input x. Then, the objective of prompt personalization can be formulated as

$$p* = \mathrm{argmax}_{p \in \mathbb{P}} \mathrm{R}(y_{LM}(p, x)), \qquad (1)$$

where $\mathbb{P}$ is the space of possible prompts. The LLM observes x embedded in the prompt to yield recommendation results y_{LM}, reflecting the user preference on candidate items. The function $\mathrm{R}(\cdot)$ quantifies the alignment between the predicted results and ground truth, with higher values indicating greater alignment. Thus, the task becomes identifying $p*$, the optimal prompt that maximizes alignment for each individual user.

3 Methodology

To enhance LLMs' recommendation capability with effective prompts, we introduce RPP/RPP+ to personalize instance-wise prompts with MARL, where RPP+ enhances RPP by incorporating a "refine" block. Comprising four actor-critic networks, the MARL serves as the core component to select actions from the four patterns and create personalized instance-wise prompts for different users. These optimized prompts are then used to prompt an LLM-based recommender for recommendations, while the MARL is trained iteratively to maximize rewards from the outputs of LLM, as shown in Figure 2. In this section, we begin by formulating prompt personalization as a **Markov Decision Process (MDP)**, followed by a detailed description of RPP/RPP+'s components, including the action space, state space, reward function, and policy architecture.

3.1 Formulation of RL for Prompt Personalization

Manual creation or heuristical crafting may fail to personalize discrete prompts automatically and efficiently. Building upon the idea of using RL for prompt personalization, we view the prompt generation as selecting actions from natural language space and formulate the problem as a MDP $\langle \mathcal{S}, \mathcal{A}, T, R, \gamma \rangle$ to optimize prompts in an instance manner, wherein,

- $\mathcal{S}$ is the state set, where $\mathbf{s} \in \mathcal{S}$ indicates the state containing information about the current prompt and recommendation result.
- $\mathcal{A}$ is the action set, where each action $a \in \mathcal{A}$ corresponds to a sentence selected from a specific pattern (e.g., "You are a movie expert" for "role-playing").
- T is the transition function, where $t(\mathbf{s}_{t+1}|\mathbf{s}_t, a_t)$ is the transition possibility that action a_t in state $\mathbf{s}_t$ at time t will lead to state $\mathbf{s}_{t+1}$ at time $t + 1$. In our task, $t(\mathbf{s}_{t+1}|\mathbf{s}_t, a_t)$ represents the possibility of LLMs' recommendation results at the $t + 1$ iteration.
- R is the reward function, $r(\mathbf{s}_t, a_t)$ determines the reward $r \in R$ for agent after interacting with the LLMs. To integrate with recommendation tasks, we define the reward r as the recommendation performance evaluated according to LLMs' answers.
- γ is the discount factor that controls the agent's emphasis on future rewards compared to immediate rewards.

For a user u, we initialize prompt p_0 with sentences randomly selected from K patterns (i.e., role-playing, history records, reasoning guidance, and output format). As depicted in Figure 2, we
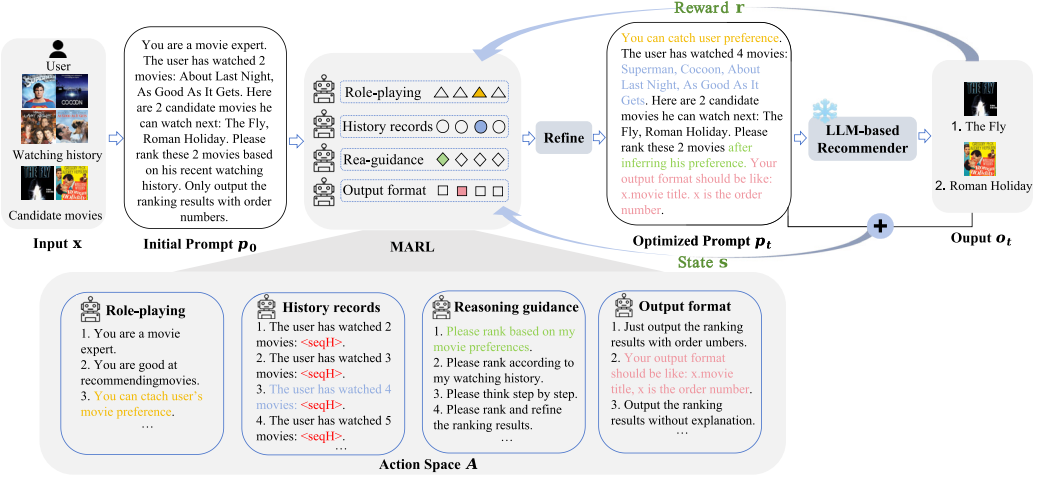
Fig. 2. The framework of our proposed RPP/RPP+. MARL serves as the core component to personalize instance-wise prompts with four distinct patterns, using corresponding agents. It is trained iteratively to maximize rewards based on the outputs of the frozen LLM-based recommender. Once trained, MARL can select optimal actions from four patterns to generate personalized prompts for each user based on their data, effectively prompting the LLM-based recommender for tailored recommendations. In addition to RPP, the "Refine" block is designed for RPP+ to enhance the flexibility and quality of the selected actions, utilizing other LLMs to dynamically refine the selected actions before prompting the LLM-based recommender.

employ a MARL under the CTDE paradigm [33]. Each agent z_k is responsible for personalizing a specific pattern with individual action space $\mathcal{A}_k$, where $k \in \{1, 2, ..., K\}$, K is the number of agents (or patterns). Prompt personalization can be formulated as seeking policies $\pi_k(a_t^{(k)}|\mathbf{s}_t)$ that selects actions $a_t^{(k)}$ from the action set $\mathcal{A}_k$ based on the global state $\mathbf{s}_t$, generating a prompt $p_t = concat(a_t^{(1)}, \ldots, a_t^{(K)})$ at step t. The generated prompt p_t interacts with LLMs (environments) to enhance LLMs' recommendation performance (reward) by maximizing the objective of our multi-agent. Formally, the objective can be defined as

$$Q\left(\mathbf{s}_t, \{a_t^{(k)}\}_{k=1}^K\right) = \mathbb{E}_{\substack{\mathbf{s}_{t'} \sim \mathcal{S} \\ \{a_{t'}^{(k)}\} \sim \pi_k\}_{k=1}^K}} \left[\sum_{t'=t}^{\infty} \gamma^{t'-t} \cdot r\left(\mathbf{s}_{t'}, \{a_{t'}^{(k)}\}_{k=1}^K\right) \right],$$

$$\max_{\{\pi_k\}_{k=1}^K} \mathcal{J}\left(\{\pi_k\}_{k=1}^K\right) = \mathbb{E}_{\substack{\mathbf{s}_t \sim \mathcal{S} \\ \{a_t^{(k)} \sim \pi_k\}_{k=1}^K}} \left[Q(\mathbf{s}_t, \{a_t^{(k)}\}_{k=1}^K) \right], \tag{2}$$

where $Q\left(\mathbf{s}_t, \{a_t^{(k)}\}_{k=1}^K\right)$ denotes the expected cumulative reward for MARL at step t which can be estimated by Q-value function. By optimizing the policies π_k and maximizing the objective $\mathcal{J}$, we can obtain the optimal prompt p_t for each user during iterations.

3.2 Action Space

Designing an effective action set is crucial when optimizing prompts with RL. The key challenge is balancing search efficiency and prompt quality. To manage this tradeoff, we develop the action space comprising several subspaces $\{\mathcal{A}_k\}_{k=1}^K$, each representing a distinct pattern and equipped with multiple sentences as actions. Inspired by previous studies [16, 46, 62] that have investigated task-wise prompts utilizing role-playing, history records, reasoning guidance, and output format patterns,

we explore personalizing instance-wise prompts by optimizing these four patterns. To improve searching efficiency, we optimize each pattern by selecting actions (sentences) from $\{\mathcal{A}_k\}_{k=1}^K$, as depicted in Figure 2. The sentence-level optimization for these four patterns enhances the efficiency of RPP/RPP+, rather than optimizing word-by-word within the vast vocabulary. Moving to the goal of ensuring prompt quality, each pattern $\mathcal{A}_k$ consists of multiple sentences meticulously designed for the recommendation tasks, free from ambiguous words, and encompassing diverse perspectives for LLMs to analyze. These sentences represent the possible actions in action space $\mathcal{A}_k$ for the corresponding agent z_k.

For implementation, the action set is jointly established by both humans and LLMs for specific tasks, such as movie ranking tasks. Each action is meticulously crafted by the LLMs and rigorously screened by humans to ensure its task relevance and encompass diverse analytical perspectives, thereby guaranteeing prompt quality. Moreover, to enhance the flexibility of personalized prompts, we leverage LLMs to refine the actions selected at each iteration, enabling a more adaptable expression for the generated prompts. And we name it as RPP+. Below we explain the design of the action subset for each pattern in detail.

Role-Playing. This pattern assigns a specific role to LLMs [46, 47] to generate responses that are consistent with a particular profession. For instance, embodying the role of LLM as a movie expert allows LLMs to leverage their extensive knowledge about the movie domain. In the context of recommendation, the actions of the "role-playing" pattern include: (1) Positioning the LLM as an expert to leverage the domain knowledge (e.g., "You are a movie expert"); (2) enhancing recommendation capabilities of LLM by assigning them as a recommender (e.g., "You are good at recommending movies"); (3) instructing LLM to discern users' inner intents, akin to a psychologist's role, such as "You are good at catching people's movie interest."

History Records. The user's historical interactions constitute critical contextual information for eliciting high-quality recommendation [16, 30]. Intuitively, certain users exhibit short-term interests best captured by recent interactions, while others having long-term preferences require a longer interaction history. Therefore, we personalize history records by varying the interaction sequence length to match users' dynamic preferences. Specifically, with an initial length l_0, at each step t, the agent's action is deciding the number of additional interactions $\hat{l}_t$ to introduce in chronological order, that is, providing $l_t = l_{t-1} + \hat{l}_t$ historical context from near to far in time. By formulating the sequence expansion as action, the agent can learn an adaptive policy to tail each user's sequence to the short versus long-term interests.

Reasoning Guidance. This pattern offers guidelines for LLMs to systematically decompose complex tasks into multiple reasoning steps, incorporating strategies such as CoT [56], reflexion [49], and refinement [35]. Since recommendation lies in inferring user intentions based on their behaviors, it is crucial to introduce intermediate steps to move from the surface to the essence. We include six action sub-categories within the "reasoning guidance" pattern: (1) direct recommendation without intermediate steps (e.g., "Please rank these candidate movies in order of priority from highest to lowest"); (2) explicitly instructing LLM to recommend based on the user's interaction history (e.g., "Please rank these candidate movies according to my watching history"); (3) prompting LLM to infer user preference before suggesting items (e.g., "Please rank these candidate movies based on my movie preferences which are inferred from my watching history"); (4) allowing LLM to deduce without specific instructions (e.g., "Please rank these candidate movies and think step by step"); (5) utilizing the refinement and update mechanism of LLMs in the process of generating recommendation results (e.g., "Please rank these candidate movies and refine the ranking results according to my watching history"); (6) requiring LLM to calculate the similarity between candidate

items and user preference before recommending items (e.g., "Please rank these candidate movies after calculating the similarity between them and my movie preferences, according to my watching history").

Output Format. This pattern defines the desired format of the LLMs' output in recommendation tasks, thereby avoiding invalid output and facilitating accurate evaluation. We provide multiple output format options tailored to the recommendation task: (1) outputting titles and order numbers, with each item on a separate line (e.g., "Please only output the ranking results with order numbers. Split these order numbers with a line break"); (2) outputting the recommendation results with order numbers without any unnecessary steps in the answer (e.g., "Attention! Just output the ranking results with order numbers and ignore any unnecessary steps"); (3) answering without explanatory text (e.g., "Please only output the ranking results with order numbers. Do not explain the reason or include any other words"); (4) exemplifying the output format [2, 66] (e.g., "Please only output the ranking results with order numbers. Your output format should be like this: x. movie title, x is the order number").

We should emphasize that the presented action set serves as a representative example of personalized prompt optimization in movie recommendations. However, this framework can be generalized to various recommendation tasks (e.g., rating prediction, item ranking) and domains (e.g., games, music) through the substitution of task requirements or domain-specific terminology. Moreover, the current action set can be expanded with additional choices belonging to each pattern. These demonstrate the flexibility and scalability of the proposed framework.

Refine Actions in RPP+. In addition to RPP, the "refine" block designed for RPP+ can dynamically refine the selected actions with other LLMs before prompting the LLM-based recommenders during each iteration, which can automatically enhance the flexibility and quality of the selected actions. To refine the selected sentence action with LLMs, we use the prompt as "Please refine this sentence to effectively prompt LLMs for recommendations." After the "refine" block of RPP+, we obtain a more polished sentence for prompting compared to that generated by RPP.

3.3 State Space

The state space provides key observations for informing the agent's actions. An effective state should include sufficient information about the current environment. Toward this goal, we initialize the state $\mathbf{s}_0$ with user-specific information. Subsequently, at step t, we update state $\mathbf{s}_t$ with the current prompt p_t and LLMs' ranking output o_t to include sufficient information about the environment. We now detail the process of acquiring these states.

Initializing State $\mathbf{s}_0$. Since state information after decisions is unavailable initially, we initialize the state $\mathbf{s}_0$ by incorporating users' personalized features to differentiate between them. Here we set the user-specific features as user embedding derived from the traditional recommender models (e.g., LightGCN [13]), which can capture collaborative signals with graphs [57, 59]. Formally, given the ID information of user u and item i, we obtain their embeddings:

$$\mathbf{u}, \mathbf{i} = f(u, i),$$
$$\mathbf{s}_0 = \mathbf{u}, \tag{3}$$

where $\mathbf{u}$ and $\mathbf{i}$ denote the representation of user u and item i, respectively, $f(\cdot)$ represents the encoder of traditional recommender models. Based on the personalized information of each user, RPP/RPP+ makes policy to obtain the initial actions $a_0^{(k)}$ and initial prompts p_0.

Updating State $\mathbf{s}_t$. To assess the prompt quality and LLMs' output comprehensively, we view the current prompt p_t and LLMs' ranking output $o_t = [\hat{i}_1, \cdots, \hat{i}_j, \cdots, \hat{i}_M]$ from LLMs as the shared state

of agents, where $\hat{i}_j$ is the jth recommended item in the output list, M is the number of candidates. To obtain the representation of states at step t, we leverage a pre-trained language model (i.e., BERT [6]) and a sequential model (i.e., GRU) to encode p_t and o_t, respectively. Formally, we have

$$\mathbf{e}_t^{(p)} = BERT(p_t),$$
$$\mathbf{e}_t^{(o)} = GRU(\hat{\mathbf{i}}_1, \hat{\mathbf{i}}_2, \cdots, \hat{\mathbf{i}}_M),$$
$$\mathbf{s}_t = \mathbf{e}_t^{(p)} + \mathbf{e}_t^{(o)}, \tag{4}$$

where $\hat{i}_j$ denotes the embedding of item $\hat{i}_j$, $\mathbf{e}_t^{(p)}$ and $\mathbf{e}_t^{(o)}$ are the representations of prompt p_t and ranking result o_t, respectively.

3.4 Actor-Critic Based Architecture of the Multi-Agent and Reward Function

To personalize each pattern individually with the corresponding agent, the MARL is structured with four A2C agents [39] under CTDE [33, 52], where each agent has independent parameters but has access to the global state, as shown in Figure 2. The Actor and Critic of each agent are two-layer fully connected neural networks parameterized by $\theta_g^{(k)}$ and $\theta_h^{(k)}$, denoted as $g^{(k)}(\cdot)$ and $h^{(k)}(\cdot)$, respectively. Specifically, each Actor $g^{(k)}(\cdot)$ makes policy to obtain action $a_t^{(k)}$ and its corresponding probability $prob_t^{(k)}$ based on the state $\mathbf{s}_t$, while each Critic $h^{(k)}(\cdot)$ evaluates the value $v_t^{(k)}$ of state $\mathbf{s}_t$. Formally, we have

$$a_t^{(k)}, prob_t^{(k)} = g^{(k)}(\mathbf{s}_t),$$
$$v_t^{(k)} = h^{(k)}(\mathbf{s}_t). \tag{5}$$

Since the aim is to boost the recommendation ability of LLMs by personalizing instance-wise prompts, we leverage **Normalized Discounted Cumulative Gain (NDCG)** directly as the reward, which is the performance metric for ranking tasks. Here we adopt $NDCG@M$ as the reward evaluated according to the answer of LLMs, where M is the number of candidate items and we set it as 10 in our experiments. Formally, given the recommendation output o_t at step t, we can define the reward r_t as below:

$$r_t = \text{NDCG@M}(o_t). \tag{6}$$

Given the current and future reward $r_t, r_{t+1}, \cdots, r_{t+n}$, current and future value $v_t^{(k)}, \cdots, v_{t+n}^{(k)}$ evaluated by the kth Critic, we can define the optimization objectives $L_a^{(k)}$ and $L_c^{(k)}$ for Actors and Critics as follows:

$$\hat{R}_t = r_{t+1} + \gamma r_{t+2} + \cdots + \gamma^{n-1} r_{t+n} + \gamma^n v_{t+n}^{(k)}, \tag{7}$$

$$L_c^{(k)} = \frac{1}{N} \sum_t (\hat{R}_{t-1} - v_t^{(k)}), \tag{8}$$

$$L_a^{(k)} = \frac{1}{N} \sum_t \log(prob_t^{(k)}(\hat{R}_{t-1} - v_t^{(k)})), \tag{9}$$

where γ is a discount factor for emphasis on short-term reward while assigning less weight to long-term reward, $\hat{R}_t$ denotes the cumulative sum of future reward.

3.5 Algorithm

The algorithm for our framework is presented in Algorithm 1.

Algorithm 1: Training RPP for Each Epoch

Input: Initial Actors $\{g^{(k)}\}_{k=1}^4$ and Critics $\{h^{(k)}\}_{k=1}^4$, initial state $\mathbf{s}_0$, learning rate α_a, α_c for Actors and Critics
Output: Actor networks $\{g^{(k)}\}_{k=1}^4$

1: **for** t=0, 1, ... **do**
2:      **for** k=1:4 **do**
3:          $a_t^{(k)}, prob_t^{(k)} \leftarrow g^{(k)}(\mathbf{s}_t)$
4:          $v_t^{(k)} \leftarrow h^{(k)}(\mathbf{s}_t)$
5:      **end for**
6:      $p_t = refine(concat(a_t^{(1)}, \ldots, a_t^{(K)}))$
7:      $o_t \leftarrow y_{LM}(p_t, x)$ #interact with LLMs
8:      Update the state $\mathbf{s}_{t+1}$ according to Equation 4
9:      Obtain the reward r_t according to Equation 6
10:     **for** k=1:4 **do**
11:         $\theta_g^{(k)} \leftarrow \theta_g^{(k)} - \nabla_{\theta_g^{(k)}} L_a^{(k)}$ # optimize Actors' parameters
12:         $\theta_h^{(k)} \leftarrow \theta_h^{(k)} - \nabla_{\theta_h^{(k)}} L_c^{(k)}$ # optimize Critics' parameters
13:     **end for**
14: **end for**

---

4 Experiments

In this section, we conduct extensive experiments with three public datasets in RSs to evaluate our proposed framework RPP/RPP+ on the ranking tasks and answer the following questions:

— RQ1: Can personalizing instance-wise prompts with our framework RPP/RPP+ effectively enhance the recommendation ability of LLMs?
— RQ2: How well can the framework RPP/RPP+ generalize on diverse types of LLMs?
— RQ3: What's the respective contribution that each pattern of the framework can make to our task?
— RQ4: To what extent is the framework sensitive to variations in hyper-parameters?

We evaluate the effectiveness of RPP/RPP+ on LLaMa2-7B-chat, comparing it against various baselines including traditional recommender models, few-shot methods, and prompt-based methods. Besides, we extend the evaluation on diverse frozen pre-trained LLMs to measure the RPP/RPP+'s generalization capability. To gain insights into the impact of different patterns in prompts, we conduct an ablation study on personalizing each pattern. Furthermore, we investigate the sensitivity of RPP/RPP+ to hyper-parameters. The case study is employed to provide a tangible illustration of the significance of personalizing prompts in enhancing LLMs' recommendation ability.

4.1 Experimental Settings

4.1.1 Datasets. We conduct experiments on three public datasets for RSs—ML-1M [12], Games [38], and Lastfm [3]. ML-1M is a widely used movie dataset that contains 1 million ratings given by users on movies. The Games dataset is a collection of user reviews and ratings for various video games available on the Amazon platform. The Lastfm dataset contains a rich collection of user listening histories, including user profiles, artist names, and track names. Following previous work [16], the reviews or ratings are regarded as interactions. The users and items with fewer than five interactions are excluded from the analysis. The interactions for each user are organized sequentially based on their timestamps, arranging the oldest interactions first. The detailed statistics of the datasets are listed in Table 1.

4.1.2 Baselines. We employ several baseline models for comparison, categorizing them into three types: (1) traditional recommender models, including Pop, BPRMF [42], LightGCN [13], and

Table 1. Statistics of Datasets

Dataset	#Users	#Items	#Interactions
ML-1M	6,040	3,885	1,000,210
Games	50,545	16,858	389,718
Lastfm	2,100	17,632	92,835

SASRec [22]; (2) few-shot models, such as BM25 [43], VQ-Rec [14], and UniSRec [15]; (3) prompt optimization methods, including manual prompts [40, 45], enumeration that is a kind of heuristic prompts [9, 19, 41], and GRIPS [41].

Traditional Recommender Models. Pop relies on item popularity for recommendations but overlooks personalization. BPRMF [42] combines Bayesian methods and matrix factorization techniques to decompose the user-item interaction matrix into a latent factor matrices. LightGCN [13] is a simplified and efficient framework designed for recommendation tasks using **Graph Convolutional Networks (GCNs)**. SASRec [22] leverages a self-attention encoder to comprehensively analyze the long-term dependencies present in user behaviors. And it predicts subsequent interactions through a feed-forward network. They are evaluated on the candidate sets after training on the training dataset with all interactions following the setting of LLMRank [16].

Few-Shot Methods. BM25 [43] is a versatile information retrieval algorithm that enhances the accuracy of item ranking by considering item frequency. VQ-Rec [14] suggests an item representation scheme to acquire Vector-Quantized item representations specifically designed to facilitate transferable sequential recommendation; UniSRec [15] leverages item description text to acquire transferable representations, which can be implemented by a lightweight item encoding architecture and two contrastive pre-trained tasks. Following the setting of LLMRank [16], we leverage their publicly available pre-trained models to evaluate their performance on ranking the candidate items without training on the target dataset.

Prompt-Based Methods. Manual prompts [40, 45] are prompts designed by humans to guide LLMs' behavior, relying on human expertise and domain knowledge. Here we meticulously craft a task-wise prompt for the ranking tasks, encompassing the key patterns within a prompt. Enumeration entails automatically searching within several possible prompts and selecting the best one. Here we design diverse prompts with variations for the ranking tasks, adopting enumeration followed by selection. GRIPS [41] performs edit operations on prompts automatically for LLMs with gradient-free search. Here we adapt it as an instance-wise prompting baseline for ranking tasks.

4.1.3 Frozen Pre-Trained LLMs. To evaluate the generalization ability of RPP/RPP+, we extend the experiments on diverse frozen pre-trained LLMs, including LLaMa2, ChatGPT, and Alpaca. They serve as the representative LLMs of three types respectively: the open source LLM, which can be inferred using pre-trained parameters; the black-box LLM, which solely offers an accessible API; and the fine-tuned LLM, tailored specifically for the recommendation task. LLaMa2 is an open source LLM released by Meta whose performance has been widely recognized. Here we select the version of LLaMa2-7B-chat as the frozen LLM. ChatGPT is a black-box LLM, which is excellent across a wide array of tasks. We conduct experiments by utilizing the API of GPT-3.5-turbo from OpenAI and set the temperature to 0.2. Alpaca is a lightweight instruction-tuning strategy based on LLaMa. We select LLama2-7B as the backbone and fine-tune Alpaca with Lora [17], which adds trainable rank decomposition matrices to each layer of the transformer architecture in the frozen LLM [1], enabling easier and more efficient fine-tuning. We transform the recommendation data

for ranking tasks into instruction-tuning data, ensuring that LLMs grasp the specific nuances of the ranking task. The format of the data for instruction tuning can be articulated as follows:

Data Example for Instruction Tuning

Instruction: "Here is my interaction history: <SeqH1>. There are several candidate items: <SeqH2>. Please rank these candidate items by measuring the possibilities that I would like to interact with. Please think step by step. Merely output the ranking results and order numbers. Split these order numbers with line break."

Output: <SeqH3>

<SeqH1>, <SeqH2>, and <SeqH3> are the lists of item titles. Subsequently, we employ the formatted "Instruction" and "Output" as the instruction examples to train Lora on ranking tasks. Then we view the fine-tuned Alpaca as a frozen LLM to conduct experiments.

4.2 Implementation Details

We configure the initial length of interaction history l_0 as 1 and the number of candidate items to be $M = 10$. In our experiments, we set 3, 9, and 5 sentence choices for "role-playing," "reasoning guidance," and "output format" patterns in the action spaces, respectively. Candidate items are randomly selected from the training dataset and include a ground-truth item sourced from the test dataset. Therefore, for ranking tasks, the HitRatio@10 metric is always 1, rendering this metric meaningless. We set the γ in the reward as 0.95. To reduce the randomness of LLMs, we set the temperature as 0.2, since a lower temperature favors more conservative and deterministic outputs. Besides, we report the average performance with corresponding standard deviations of prompt-based methods and RPP/RPP+, based on at least five repeated runs. Traditional recommender models and few-shot-based methods are executed on an NVIDIA RTX 3090Ti, while prompt-based methods and our framework, based on LLMs, run on an NVIDIA A100. During the training phase, we randomly select 200 users as training examples to optimize the multi-agent strategy employed in RPP/RPP+. We terminate the iterative training process if the highest NDCG@10 does not update over 7 iterations or if the total iterations exceed 15. Subsequently, during the testing phase, the multi-agent, now equipped with an optimized policy, undergoes evaluation on a separate set of 100/200 test examples. Our experiments show that on the training set, after three rounds of prompt optimization, the optimized prompts can achieve superior performance on ChatGPT for most instances, while it requires two, seven, and eight iterations on the ML-1M, Games, and Lastfm datasets for LLaMa2-7B-chat, respectively, and approximately eight iterations for Alpaca. Consequently, during the inference process, we terminate the iterations at the corresponding rounds for different datasets and LLMs. We only require users' collaborative embeddings to initialize the state, along with their interaction history and candidate items as input. To address invalid responses from LLMs, we carefully design prompts to enforce LLMs' output format, reducing these occurrences. Additionally, we trim and pad the LLMs' recommendations as 10 while processing invalid responses.

4.3 Overall Results (RQ1)

To answer RQ1, we evaluate the effectiveness of our framework on LLaMa2-7B-chat by comparing the performance of RPP/RPP+ with diverse baselines. We adopt the evaluating metrics of NDCG@1, NDCG@5, and NDCG@10 to access LLMs' capability in ranking items, the results are listed in Table 2. Our observations are as follows: (1) Task-wise prompting methods (i.e., manual prompts and enumeration) exhibit inferior recommendation performance compared to traditional recommender

Table 2. Performance Comparison of Our Framework on LLaMa2-7B-Chat with Baselines across Diverse Datasets

Methods	ML-1M			Games			Lastfm		
	N@1	N@5	N@10	N@1	N@5	N@10	N@1	N@5	N@10
Pop	0.26	0.60	0.63	0.39	0.61	0.67	0.78	0.86	0.88
BPRMF	0.44	0.71	0.74	0.57	0.75	0.78	0.71	0.80	0.84
LightGCN	0.33	0.62	0.65	0.47	0.70	0.73	0.79	0.82	0.88
SASRec	<u>0.68</u>	<u>0.84</u>	<u>0.85</u>	<u>0.69</u>	<u>0.83</u>	<u>0.85</u>	<u>0.81</u>	<u>0.88</u>	<u>0.89</u>
BM25	0.08	0.2	0.43	0.27	0.45	0.57	-	-	-
UniSRec	0.12	0.35	0.47	0.25	0.45	0.56	-	-	-
VQ-Rec	0.10	0.33	0.47	0.14	0.33	0.48	-	-	-
Manual prompts (task-wise)	$0.03_{\pm0.02}$	$0.35_{\pm0.01}$	$0.41_{\pm0.01}$	$0.04_{\pm0.01}$	$0.30_{\pm0.02}$	$0.35_{\pm0.02}$	$0.05_{\pm0.03}$	$0.39_{\pm0.01}$	$0.43_{\pm0.01}$
Enumeration (task-wise)	$0.04_{\pm0.00}$	$0.36_{\pm0.00}$	$0.42_{\pm0.01}$	$0.04_{\pm0.00}$	$0.34_{\pm0.01}$	$0.37_{\pm0.01}$	$0.05_{\pm0.00}$	$0.40_{\pm0.01}$	$0.44_{\pm0.02}$
GRIPS (instance-wise)	$0.43_{\pm0.01}$	$0.70_{\pm0.03}$	$0.73_{\pm0.05}$	$0.62_{\pm0.04}$	$0.67_{\pm0.02}$	$0.72_{\pm0.02}$	$0.19_{\pm0.03}$	$0.64_{\pm0.01}$	$0.66_{\pm0.04}$
RPP (instance-wise)	$0.82_{\pm0.03}$ ↑	$0.85_{\pm0.03}$ ↑	$0.87_{\pm0.02}$ ↑	$0.79_{\pm0.03}$ ↑	$0.82_{\pm0.03}$ ↑	$0.85_{\pm0.02}$ ↑	$0.87_{\pm0.01}$ ↑	$0.89_{\pm0.01}$ ↑	$0.91_{\pm0.01}$ ↑
Improvement	0.78	0.49	0.45	0.75	0.48	0.48	0.82	0.49	0.47
Relative improvement (%)	1,950	136	107	1,875	141	130	1,640	123	107
RPP+ (instance-wise)	$0.82_{\pm0.01}$ ↑	$0.85_{\pm0.00}$ ↑	$0.87_{\pm0.01}$ ↑	$0.83_{\pm0.02}$ ↑	$0.85_{\pm0.01}$ ↑	$0.86_{\pm0.01}$ ↑	$0.93_{\pm0.02}$ ↑	$0.94_{\pm0.01}$ ↑	$0.95_{\pm0.02}$ ↑
Improvement	0.78	0.49	0.45	0.79	0.51	0.49	0.88	0.54	0.48
Relative improvement (%)	1,950	136	107	1,975	150	132	1,760	135	109

The results of the best-performing baseline are denoted with an underline. The results of our method which outperform other prompt-based methods are indicated with ↑, while those that surpass the best-performing baseline are highlighted in red. The performance improvement of RPP/RPP+ compared with the best-performing task-wise prompting method (i.e., Enumeration) is listed in the "improvement."

models, few-shot methods, and instance-wise prompting methods (i.e., GRIPS and RPP/RPP+). For example, the best-performing baseline in traditional recommender models, SASRec, surpasses the best-performing task-wise prompting method (enumeration) by 0.64, 0.48, and 0.43 in NDCG@1, NDCG@5, and NDCG@10 metrics, respectively, on the ML-1M dataset. This validates the weakness of task-wise prompts for LLMs. (2) Instance-wise prompting implemented by RPP/RPP+ makes the frozen pre-trained LLaMa2-7B-chat surpass traditional recommender models across a variety of datasets. For instance, RPP demonstrates superior performance compared to the leading baseline in traditional recommender models (SASRec), achieving improvements of 0.14, 0.01, and 0.02 in terms of the NDCG@1, NDCG@5, and NDCG@10 metrics on the ML-1M. (3) The framework of RPP/RPP+ outperforms other instance-wise promoting methods in the ranking tasks, such as GRIPS. This proves the high quality of prompts generated by RPP/RPP+. We conduct multiple tests and the minimal standard deviations demonstrate that the performance improvements resulting from RPP/RPP+ are statistically significant. Additionally, we adopt the MRR@5, MRR@10, and HitRatio@5 metrics to evaluate the performance of RPP/RPP+ in Table 3, as these are widely used benchmarks. The results provide valuable insights that highlight the superiority of our method, which achieves state-of-the-art performance compared to the baselines. These overall results illustrate that due to the ability to personalize prompts from instance-wise, RPP/RPP+ enables LLMs to comprehend users' personalized intents better and provide more accurate recommendations.

4.4 Generalization Ability (RQ2)

To answer RQ2, we extend the evaluation of RPP/RPP+ on diverse frozen LLMs (i.e., LLaMa2-7B-chat, ChatGPT, and Alpaca) and compare the performance of RPP/RPP+ with other prompt-based baselines. The results are listed in Table 4. RPP/RPP+ can generalize well on diverse LLMs and prominently enhance the ranking performance of LLMs. For example, RPP outperforms enumeration in terms of the NDCG@10 metric on LLaMa-7B-chat, ChatGPT, and fine-tuned Alpaca by 0.48, 0.24,

Table 3. Additional Metrics Comparison of Our Framework on LLaMa2-7B-Chat with Baselines across Diverse Datasets, Where "M@" Means MRR Metrics and "H@" Means HitRatio Metrics

Methods	ML-1M			Games			Lastfm		
	M@5	M@10	H@5	M@5	M@10	H@5	M@5	M@10	H@5
SASRec	0.67	0.78	0.80	0.67	0.76	0.87	0.71	0.75	0.81
Manual prompts (task-wise)	$0.24_{\pm 0.00}$	$0.25_{\pm 0.01}$	$0.69_{\pm 0.02}$	$0.29_{\pm 0.01}$	$0.39_{\pm 0.02}$	$0.70_{\pm 0.02}$	$0.27_{\pm 0.02}$	$0.29_{\pm 0.02}$	$0.76_{\pm 0.01}$
Enumeration (task-wise)	$0.28_{\pm 0.02}$	$0.29_{\pm 0.03}$	$0.70_{\pm 0.01}$	$0.38_{\pm 0.01}$	$0.45_{\pm 0.01}$	$0.72_{\pm 0.00}$	$0.31_{\pm 0.01}$	$0.32_{\pm 0.00}$	$0.84_{\pm 0.01}$
GRIPS (instance-wise)	$0.52_{\pm 0.03}$	$0.54_{\pm 0.02}$	$0.74_{\pm 0.02}$	$0.51_{\pm 0.03}$	$0.52_{\pm 0.04}$	$0.73_{\pm 0.02}$	$0.50_{\pm 0.01}$	$0.51_{\pm 0.03}$	$0.76_{\pm 0.02}$
RPP (instance-wise)	$0.80_{\pm 0.01}$↑	$0.82_{\pm 0.03}$↑	$0.86_{\pm 0.03}$↑	$0.80_{\pm 0.02}$↑	$0.83_{\pm 0.02}$↑	$0.84_{\pm 0.03}$↑	$0.85_{\pm 0.01}$↑	$0.87_{\pm 0.02}$↑	$0.87_{\pm 0.02}$↑
RPP+ (instance-wise)	$0.82_{\pm 0.02}$↑	$0.83_{\pm 0.01}$↑	$0.87_{\pm 0.00}$↑	$0.85_{\pm 0.01}$↑	$0.86_{\pm 0.00}$↑	$0.87_{\pm 0.01}$↑	$0.88_{\pm 0.01}$↑	$0.89_{\pm 0.00}$↑	$0.89_{\pm 0.01}$↑

The results of the best-performing baseline are denoted with an underline. The results of our method which outperform other prompt-based methods are indicated with ↑.

Table 4. Performance of RPP and RPP+ on Diverse Frozen LLMs

Methods	ML-1M			Games			Lastfm		
	N@1	N@5	N@10	N@1	N@5	N@10	N@1	N@5	N@10
Manual prompts (L)	$0.03_{\pm 0.02}$	$0.35_{\pm 0.01}$	$0.41_{\pm 0.01}$	$0.04_{\pm 0.01}$	$0.30_{\pm 0.02}$	$0.35_{\pm 0.02}$	$0.05_{\pm 0.03}$	$0.39_{\pm 0.01}$	$0.43_{\pm 0.01}$
Enumeration (L)	$0.04_{\pm 0.00}$	$0.36_{\pm 0.00}$	$0.42_{\pm 0.01}$	$0.04_{\pm 0.00}$	$0.34_{\pm 0.01}$	$0.37_{\pm 0.01}$	$0.05_{\pm 0.00}$	$0.40_{\pm 0.01}$	$0.44_{\pm 0.02}$
GRIPS (L)	$0.43_{\pm 0.01}$	$0.70_{\pm 0.03}$	$0.73_{\pm 0.05}$	$0.62_{\pm 0.04}$	$0.67_{\pm 0.02}$	$0.72_{\pm 0.02}$	$0.19_{\pm 0.03}$	$0.64_{\pm 0.01}$	$0.66_{\pm 0.04}$
RPP (L)	$0.82_{\pm 0.03}$↑	$0.85_{\pm 0.03}$↑	$0.87_{\pm 0.02}$↑	$0.79_{\pm 0.03}$↑	$0.82_{\pm 0.03}$↑	$0.85_{\pm 0.02}$↑	$0.87_{\pm 0.01}$↑	$0.89_{\pm 0.01}$↑	$0.91_{\pm 0.01}$↑
RPP+ (L)	$0.82_{\pm 0.01}$↑	$0.85_{\pm 0.00}$↑	$0.87_{\pm 0.01}$↑	$0.83_{\pm 0.02}$↑	$0.85_{\pm 0.01}$↑	$0.86_{\pm 0.01}$↑	$0.93_{\pm 0.02}$↑	$0.94_{\pm 0.01}$↑	$0.95_{\pm 0.02}$↑
Manual prompts (C)	$0.25_{\pm 0.03}$	$0.53_{\pm 0.03}$	$0.57_{\pm 0.03}$	$0.24_{\pm 0.06}$	$0.46_{\pm 0.05}$	$0.52_{\pm 0.05}$	$0.13_{\pm 0.09}$	$0.41_{\pm 0.05}$	$0.50_{\pm 0.06}$
Enumeration (C)	$0.29_{\pm 0.00}$	$0.55_{\pm 0.00}$	$0.62_{\pm 0.01}$	$0.31_{\pm 0.00}$	$0.50_{\pm 0.01}$	$0.56_{\pm 0.02}$	$0.24_{\pm 0.02}$	$0.47_{\pm 0.03}$	$0.52_{\pm 0.02}$
GRIPS (C)	$0.41_{\pm 0.03}$	$0.55_{\pm 0.02}$	$0.65_{\pm 0.01}$	$0.25_{\pm 0.02}$	$0.55_{\pm 0.01}$	$0.62_{\pm 0.02}$	$0.41_{\pm 0.02}$	$0.64_{\pm 0.02}$	$0.69_{\pm 0.04}$
RPP (C)	$0.35_{\pm 0.04}$↑	$0.58_{\pm 0.03}$↑	$0.65_{\pm 0.02}$↑	$0.53_{\pm 0.02}$↑	$0.67_{\pm 0.01}$↑	$0.73_{\pm 0.01}$↑	$0.66_{\pm 0.02}$↑	$0.83_{\pm 0.02}$↑	$0.83_{\pm 0.01}$↑
RPP+ (C)	$0.38_{\pm 0.07}$↑	$0.59_{\pm 0.02}$↑	$0.66_{\pm 0.01}$↑	$0.59_{\pm 0.02}$↑	$0.69_{\pm 0.02}$↑	$0.75_{\pm 0.01}$↑	$0.67_{\pm 0.02}$↑	$0.84_{\pm 0.01}$↑	$0.85_{\pm 0.01}$↑
Manual prompts (A)	$0.04_{\pm 0.04}$	$0.24_{\pm 0.16}$	$0.34_{\pm 0.04}$	$0.09_{\pm 0.02}$	$0.34_{\pm 0.04}$	$0.43_{\pm 0.02}$	$0.06_{\pm 0.06}$	$0.29_{\pm 0.16}$	$0.40_{\pm 0.05}$
Enumeration (A)	$0.06_{\pm 0.01}$	$0.30_{\pm 0.02}$	$0.37_{\pm 0.02}$	$0.14_{\pm 0.01}$	$0.39_{\pm 0.01}$	$0.43_{\pm 0.02}$	$0.12_{\pm 0.00}$	$0.38_{\pm 0.01}$	$0.44_{\pm 0.01}$
GRIPS (A)	$0.06_{\pm 0.01}$	$0.63_{\pm 0.02}$	$0.65_{\pm 0.00}$	$0.02_{\pm 0.01}$	$0.60_{\pm 0.02}$	$0.63_{\pm 0.02}$	$0.19_{\pm 0.02}$	$0.68_{\pm 0.01}$	$0.70_{\pm 0.00}$
RPP (A)	$0.82_{\pm 0.1}$↑	$0.92_{\pm 0.01}$↑	$0.93_{\pm 0.03}$↑	$0.96_{\pm 0.00}$↑	$0.96_{\pm 0.00}$↑	$0.97_{\pm 0.00}$↑	$0.91_{\pm 0.03}$↑	$0.91_{\pm 0.03}$↑	$0.93_{\pm 0.01}$↑
RPP+ (A)	$0.84_{\pm 0.12}$↑	$0.94_{\pm 0.05}$↑	$0.94_{\pm 0.05}$↑	$0.96_{\pm 0.00}$↑	$0.96_{\pm 0.01}$↑	$0.97_{\pm 0.00}$↑	$0.92_{\pm 0.00}$↑	$0.92_{\pm 0.00}$↑	$0.92_{\pm 0.00}$↑

The results of our framework which outperform other prompt-based baselines are indicated with ↑, and the best-performing methods are highlighted in red. L, C, and A denote LLMs of LLaMa2-7B-chat, ChatGPT, and Alpaca, respectively.

and 0.53, respectively, on the Lastfm dataset. The relatively less significant impact on ChatGPT's performance might be attributed to its stronger robustness to prompts compared with other LLMs. The consistent performance improvement across the three types of frozen LLMs demonstrates the effectiveness and generalization ability of RPP/RPP+.

The results for Q1 and Q2 underscore the importance of instance-wise prompting and the effectiveness of RPP/RPP+ from a practical standpoint, confirming our viewpoint that personalizing instance-wise prompts can enhance LLMs' recommendation ability. Furthermore, dynamically refining actions during the iterative process can lead to a slight performance improvement of RPP+ compared to RPP, possibly due to more flexible and higher-quality prompts after refinement. However, the improvement is not particularly significant ranging from 0.00 to 0.06 and requires additional computational resources during the LLM refinement process, the tradeoff between efficiency and performance can be considered when choosing between the RPP and RPP+ in practice.

Table 5. Ablation Study on the Respective Impacts of Four Patterns on RPP

Methods	ML-1M			Games			Lastfm		
	N@1	N@5	N@10	N@1	N@5	N@10	N@1	N@5	N@10
Manual prompts	$0.03_{\pm0.02}$	$0.35_{\pm0.01}$	$0.41_{\pm0.01}$	$0.04_{\pm0.01}$	$0.30_{\pm0.02}$	$0.35_{\pm0.02}$	$0.05_{\pm0.03}$	$0.39_{\pm0.01}$	$0.43_{\pm0.01}$
Enumeration	$0.04_{\pm0.00}$	$0.36_{\pm0.00}$	$0.42_{\pm0.01}$	$0.04_{\pm0.00}$	$0.34_{\pm0.01}$	$0.37_{\pm0.01}$	$0.05_{\pm0.00}$	$0.40_{\pm0.01}$	$0.44_{\pm0.02}$
GRIPS	$0.43_{\pm0.01}$	$0.70_{\pm0.03}$	$0.73_{\pm0.05}$	$0.62_{\pm0.04}$	$0.67_{\pm0.02}$	$0.72_{\pm0.02}$	$0.19_{\pm0.03}$	$0.64_{\pm0.01}$	$0.66_{\pm0.04}$
Role-playing	$0.76_{\pm0.01}$	$0.80_{\pm0.01}$	$0.84_{\pm0.01}$	$0.66_{\pm0.05}$	$0.72_{\pm0.05}$	$0.76_{\pm0.04}$	$0.84_{\pm0.02}$	$0.87_{\pm0.02}$	$0.89_{\pm0.02}$
History records	$0.65_{\pm0.10}$	$0.72_{\pm0.07}$	$0.77_{\pm0.06}$	$0.67_{\pm0.00}$	$0.71_{\pm0.01}$	$0.76_{\pm0.00}$	$0.81_{\pm0.02}$	$0.85_{\pm0.01}$	$0.87_{\pm0.01}$
Reasoning guidance	$0.78_{\pm0.01}$	$0.81_{\pm0.01}$	$0.84_{\pm0.00}$	$0.74_{\pm0.04}$	$0.78_{\pm0.04}$	$0.80_{\pm0.04}$	$0.85_{\pm0.02}$	$0.88_{\pm0.02}$	$0.89_{\pm0.02}$
Output format	$0.74_{\pm0.00}$	$0.77_{\pm0.04}$	$0.82_{\pm0.03}$	$0.67_{\pm0.02}$	$0.71_{\pm0.02}$	$0.76_{\pm0.02}$	$0.83_{\pm0.03}$	$0.86_{\pm0.04}$	$0.89_{\pm0.03}$
RPP	$0.82_{\pm0.03}$	$0.85_{\pm0.03}$	$0.87_{\pm0.02}$	$0.79_{\pm0.03}$	$0.82_{\pm0.03}$	$0.85_{\pm0.02}$	$0.87_{\pm0.01}$	$0.89_{\pm0.01}$	$0.91_{\pm0.01}$

Table 6. Ablation Study on the Respective Impacts of Four Patterns on RPP+

| Methods	ML-1M			Games			Lastfm		
	N@1	N@5	N@10	N@1	N@5	N@10	N@1	N@5	N@10
Manual prompts	$0.03_{\pm0.02}$	$0.35_{\pm0.01}$	$0.41_{\pm0.01}$	$0.04_{\pm0.01}$	$0.30_{\pm0.02}$	$0.35_{\pm0.02}$	$0.05_{\pm0.03}$	$0.39_{\pm0.01}$	$0.43_{\pm0.01}$
Enumeration	$0.04_{\pm0.00}$	$0.36_{\pm0.00}$	$0.42_{\pm0.01}$	$0.04_{\pm0.00}$	$0.34_{\pm0.00}$	$0.37_{\pm0.01}$	$0.05_{\pm0.00}$	$0.40_{\pm0.01}$	$0.44_{\pm0.02}$
GRIPS	$0.43_{\pm0.01}$	$0.70_{\pm0.03}$	$0.73_{\pm0.05}$	$0.62_{\pm0.04}$	$0.67_{\pm0.02}$	$0.72_{\pm0.02}$	$0.19_{\pm0.03}$	$0.64_{\pm0.01}$	$0.66_{\pm0.04}$
Role-playing+	$0.75_{\pm0.01}$	$0.82_{\pm0.03}$	$0.84_{\pm0.01}$	$0.74_{\pm0.04}$	$0.79_{\pm0.02}$	$0.82_{\pm0.04}$	$0.86_{\pm0.02}$	$0.89_{\pm0.02}$	$0.90_{\pm0.01}$
History records+	$0.67_{\pm0.08}$	$0.73_{\pm0.06}$	$0.78_{\pm0.03}$	$0.66_{\pm0.00}$	$0.74_{\pm0.01}$	$0.81_{\pm0.01}$	$0.79_{\pm0.04}$	$0.86_{\pm0.01}$	$0.89_{\pm0.03}$
Reasoning guidance+	$0.78_{\pm0.01}$	$0.83_{\pm0.03}$	$0.85_{\pm0.04}$	$0.74_{\pm0.05}$	$0.78_{\pm0.03}$	$0.81_{\pm0.03}$	$0.83_{\pm0.02}$	$0.84_{\pm0.04}$	$0.88_{\pm0.02}$
Output format+	$0.72_{\pm0.01}$	$0.77_{\pm0.02}$	$0.80_{\pm0.03}$	$0.76_{\pm0.04}$	$0.79_{\pm0.03}$	$0.80_{\pm0.02}$	$0.84_{\pm0.05}$	$0.87_{\pm0.04}$	$0.91_{\pm0.03}$
RPP+	$0.82_{\pm0.01}$	$0.85_{\pm0.00}$	$0.87_{\pm0.01}$	$0.83_{\pm0.02}$	$0.85_{\pm0.01}$	$0.86_{\pm0.01}$	$0.93_{\pm0.02}$	$0.94_{\pm0.01}$	$0.95_{\pm0.02}$

4.5 Ablation Study (RQ3)

The ablation study explores the impact of personalizing each pattern (i.e., "role-playing," "history records," "reasoning guidance," and "output format") on facilitating LLMs' recommendation ability. Specifically, we perform experiments on the frozen LLaMa2-7B-chat with four variations of RPP/RPP+, where only one pattern in the prompts is personalized by the corresponding agent. As shown in Tables 5 and 6 and Figure 3, personalizing any of the patterns can enhance the performance of LLMs to rank items, and the full framework of RPP/RPP+ attains the highest performance. Personalizing "reasoning guidance" yields the greatest benefits compared with other patterns, providing interpretability for recommendations and demonstrating the effectiveness of the personalized reasoning process in accurately analyzing user preferences. Besides, "role-playing" and "output format" underscore LLMs' inherent sensitivity to prompts that modifications to the prompt expression can lead to LLMs' better output. The "history records" confirms the impact of users' short or long-term interaction history on recommendations. This emphasizes that decomposing the prompt into four patterns and individually tailoring them can contribute to LLMs' recommendations. Furthermore, as some recommendation tasks focus on user modeling, we define a new pattern called "user profile" and conduct experiments to evaluate the effectiveness of RPP/RPP+ in personalizing this pattern using the ML-1M dataset. We utilize users' profile features, such as gender, age, and occupation, as potential actions for the agent, which can choose whether to include each feature in the prompts. As shown in Table 7, personalizing the user profile leads to performance improvements compared with prompt-based baselines, highlighting the effectiveness of designating "user profile" as a distinct pattern and the advantages of personalizing prompts with RPP/RPP+.

To validate the design of "GRU" in Equation (4), we conduct ablation experiments to replace "GRU" as "mean pooling." As presented in Table 8, the performance of "RPP-pool" is worse than

Table 7. Additional Experiments for the Pattern of "User Profile"

Methods	ML-1M		
	N@1	N@5	N@10
Manual prompts (task-wise)	$0.05_{\pm0.01}$	$0.41_{\pm0.02}$	$0.46_{\pm0.01}$
Enumeration (task-wise)	$0.16_{\pm0.00}$	$0.48_{\pm0.02}$	$0.51_{\pm0.02}$
GRIPS (instance-wise)	$0.63_{\pm0.01}$	$0.72_{\pm0.02}$	$0.77_{\pm0.00}$
User profile (instance-wise)	$0.78_{\pm0.01}\uparrow$	$0.79_{\pm0.03}\uparrow$	$0.82_{\pm0.03}\uparrow$
RPP (instance-wise)	$0.83_{\pm0.02}\uparrow$	$0.83_{\pm0.01}\uparrow$	$0.88_{\pm0.01}\uparrow$
RPP+ (instance-wise)	$0.83_{\pm0.00}\uparrow$	$0.84_{\pm0.01}\uparrow$	$0.89_{\pm0.00}\uparrow$
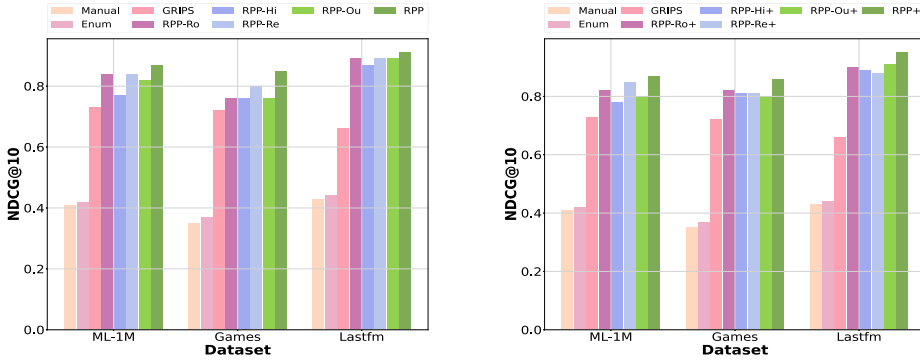
(a) The performance comparison of different patterns for RPP. (b) The performance comparison of different patterns for RPP+.

Fig. 3. The performance comparison between prompt-based methods and RPP/RPP+ with different patterns. "Manual," "Enum," and "GRIPS" represent the baseline prompt-based methods. "RPP-Ro," "RPP-Hi," "RPP-Re," and "RPP-Ou" are the four variations of RPP/RPP+ on "role-playing," "history records," "reasoning guidance," and "output format" patterns, respectively.

Table 8. Ablation Study on the Impact of GRU

Methods	ML-1M			Games			Lastfm		
	N@1	N@5	N@10	N@1	N@5	N@10	N@1	N@5	N@10
Manual prompts	$0.03_{\pm0.02}$	$0.35_{\pm0.01}$	$0.41_{\pm0.01}$	$0.04_{\pm0.01}$	$0.30_{\pm0.02}$	$0.35_{\pm0.02}$	$0.05_{\pm0.03}$	$0.39_{\pm0.01}$	$0.43_{\pm0.01}$
Enumeration	$0.04_{\pm0.00}$	$0.36_{\pm0.00}$	$0.42_{\pm0.01}$	$0.04_{\pm0.00}$	$0.34_{\pm0.01}$	$0.37_{\pm0.01}$	$0.05_{\pm0.00}$	$0.40_{\pm0.01}$	$0.44_{\pm0.02}$
GRIPS	$0.43_{\pm0.01}$	$0.70_{\pm0.03}$	$0.73_{\pm0.05}$	$0.62_{\pm0.04}$	$0.67_{\pm0.02}$	$0.72_{\pm0.02}$	$0.19_{\pm0.03}$	$0.64_{\pm0.01}$	$0.66_{\pm0.04}$
RPP-pool	$0.80_{\pm0.00}$	$0.84_{\pm0.01}$	$0.86_{\pm0.02}$	$0.76_{\pm0.03}$	$0.79_{\pm0.00}$	$0.81_{\pm0.02}$	$0.90_{\pm0.01}$	$0.92_{\pm0.01}$	$0.93_{\pm0.00}$
RPP-Bert	$0.82_{\pm0.01}$	$0.85_{\pm0.00}$	$0.87_{\pm0.01}$	$0.83_{\pm0.02}$	$0.85_{\pm0.01}$	$0.86_{\pm0.01}$	$0.93_{\pm0.02}$	$0.94_{\pm0.01}$	$0.95_{\pm0.02}$

that of "RPP-Bert," which validates the effectiveness of the "GRU" module since it can catch the sequential information of ranking results. Additionally, to evaluate the effectiveness of MARL which can personalize different patterns with different expertise from different agents, we utilize a single agent which is designed as PPO. The results are demonstrated in Figure 6(a), where the PPO-1 uses a single PPO to personalize the entire prompts without dividing them into four patterns, while PPO-4 personalizes the four patterns uniformly with one PPO. The superior performance of RPP compared to PPO-1 and PPO-4 highlights the need for multiple agents to personalize each of the four patterns respectively.
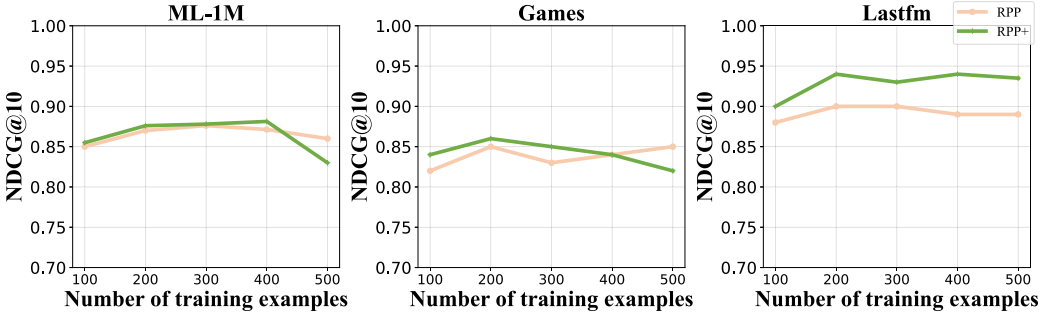
Fig. 4. Sensitivity to the number of training examples, which demonstrates the changes in RPP/RPP+'s performance on three datasets as the number of training examples increases from 100 to 500.
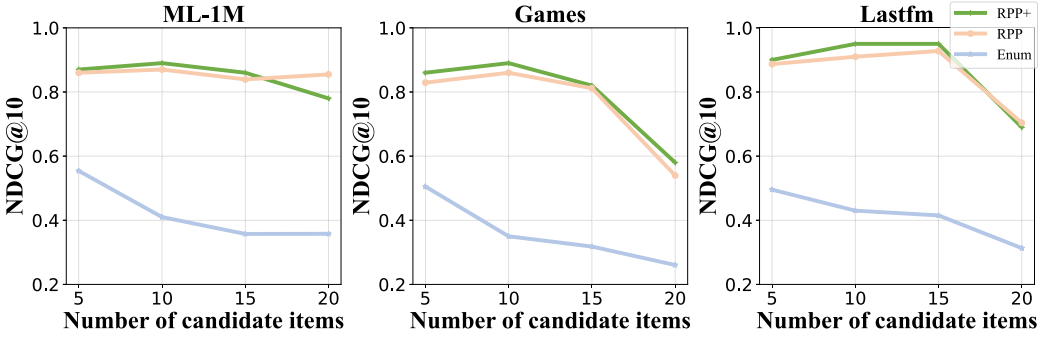

Fig. 5. Sensitivity to the number of candidate items, which indicates the drop in performance of LLM-based methods as the number of candidate items increases.

4.6 Sensitivity Analysis (RQ4)

We investigate the effect of the number of training examples (users) on RPP/RPP+ with LLaMa2-7B-chat in Figure 4. As the number of training examples increases from 100 to 500, the performance of RPP/RPP+ initially improves and subsequently stabilizes. Considering both the performance and resource consumption, we set the number of training examples as 200. Additionally, we explore the sensitivity of RPP/RPP+ to the number of candidate items within the range of {5, 10, 15, 20} in Figure 5. Observing the curves corresponding to the task-wise prompting method (e.g., enumeration) and our framework RPP/RPP+, the ranking performance of LLMs drops as the number of candidate items increases, which can be attributed to the limitation of LLMs in processing long inputs. However, the capability of RPP/RPP+ to enhance the recommendation performance of LLMs is robust since the gap between the curves of RPP/RPP+ and Enumeration remains stable. This observation highlights the effectiveness and stability of personalizing instance-wise prompts rather than solely relying on task-wise prompts.

Furthermore, to evaluate the impacts of different numbers of sentences under each pattern, we conduct experiments for the four patterns with results presented in Figure 6(b). The curves rise initially and then stabilize, demonstrating that when the sentence actions encompass a sufficient range of perspectives for each pattern, the performance of prompt personalization can be sustained at a high level.
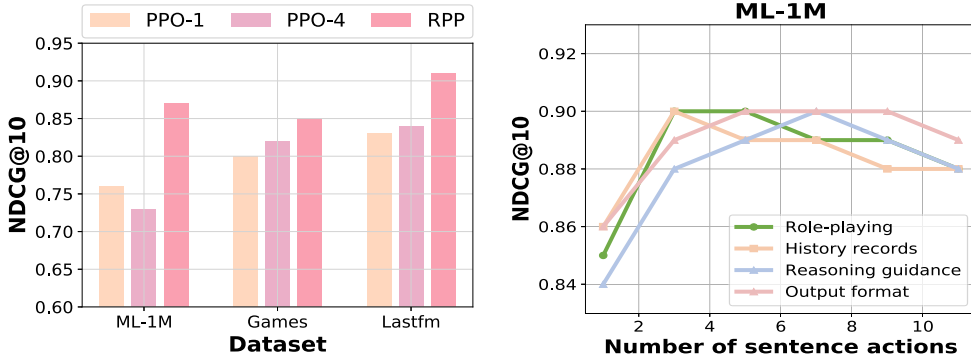
(a) Ablation study for multi-agent reinforcement learning. (b) Sensitivity to the number of sentence actions for each pattern.

Fig. 6. (a) The performance comparison between MARL and a single PPO for personalizing prompts. (b) Demonstrates RPP/RPP+'s sensitivity to the number of actions under each pattern.

4.7 Case Study

Through the case study presented in Figures 7–10, we provide an intuitive demonstration of how variations in these prompts impact the ranking results. For the user in Figure 7, personalizing the "role-playing" pattern from "You are a movie expert" to " You are a movie ranker to match user's interest" results in a higher ranking of the ground truth item that the user interacts with next. This illustrates that assigning LLMs as movie rankers to analyze users' preferences can lead to more precise recommendations for this user. By personalizing the "history record" pattern from five interaction records to two historical records, the ranking results become more precise for the user in Figure 8. This demonstrates that considering a shorter-term interaction history is more effective in capturing his preferences than a long-term one. For the user in Figure 9, personalizing the "reasoning guidance" pattern from "think step by step" to "update the film preferences before ranking" leads to better recommendations, proving that utilizing LLMs' refinement and update mechanism is more reasonable than deducing without specific instructions for him. Moreover, personalizing the "output format" pattern from "giving the output format example" to "answering without any unnecessary steps" makes it easier for LLMs to understand the instructions and recommend for the user in Figure 10. We can observe that due to LLMs' sensitivity to prompts and users' inner differences, personalizing prompts for each user enables LLMs to generate recommendation results that better satisfy the users, with the ground truth prioritized. Furthermore, we visualize the distribution of users across different patterns under our experiments' setting to illustrate user diversity vividly, as depicted in Figures 11–14, underscoring the necessity of personalizing each pattern in the prompts for different users. This further validates the significance of instance-wise prompting for recommendation tasks and the effectiveness of our framework.

4.8 Timing and Computational Complexity Analysis

Compared to directly applying fixed prompts, the streamlined architecture of MARL, with parameters negligible relative to LLMs, incurs minimal additional computational and time costs. When customizing the prompt for each user, compared with heuristical crafting methods like enumeration or token-level optimization approaches like RLPrompt [5] and AutoPrompt [48], the sentence-level optimization can improve searching efficiency and obtain optimal prompts with minimal iterations, thus reducing timing and computational complexity. However, the multi-round iteration with LLMs is still required since achieving the desired prompt in a single step with MARL is challenging.

Input: You are a movie expert. I've watched the following movies in the past in order: '0. Superman', '1. Cocoon'. Now there are 10 candidate movies that I can watch next: '0. The Celluloid Closet', '1. Rudy', '2. Commandments', '3. The Fly', '4. Lolita', '5. Nighthawks', '6. The Show', '7. Red Corner', '8. The Santa Clause', '9. Fatal Beauty'. Please rank these 10 movies according to my watching history. Please only output the ranking results with order numbers. Split your output with line break.

Output: 1. The Fly \n 2. Lolita \n 3. Nighthawks \n 4. Commandments \n 5. The Celluloid Closet \n 6. Rudy \n 7. The Show \n 8. Red Corner \n 9. The Santa Clause \n 10. Fatal Beauty \n

Input: You are a movie ranker to match user's interest. I've watched the following movies in the past in order: '0. About Last Night...', '1. As Good As It Gets'. Now there are 10 candidate movies that I can watch next: '0. The Celluloid Closet', '1. Lolita', '2. Commandments', '3. The Fly', '4. Rudy', '5. Nighthawks', '6. The Show', '7. Red Corner', '8. The Santa Clause', '9. Fatal Beauty'. Please rank these 10 movies according to my watching history. Please only output the ranking results with order numbers. Split your output with line break.

Output: 1. The Santa Clause \n 2. Rudy \n 3. The Fly \n 4. The Celluloid Closet \n 5. The Show \n 6. Commandments \n 7. Nighthawks \n 8. Red Corner \n 9. Lolita \n 10. Fatal Beauty

Fig. 7. The case study on "role-playing" patterns. Variations of prompt pairs are marked green, the candidate items are marked blue, and the ground truth is marked red.

Input: I've watched the following movies in the past in order: 0. Sister Act', '1. Born Yesterday', '2.Grumpy Old Men', '3. Arthur', '4. Terms of Endearment'. Now there are 10 candidate movies that I can watch next: '0. Mrs. Brown (Her Majesty, Mrs. Brown)', '1. Police Academy 4: Citizens on Patrol', '2. Wisdom', '3. Guantanamera', '4. Seven Samurai (The Magnificent Seven) (Shichinin nosamurai)', '5. Pink Flamingos', '6. Wayne's World 2 ', '7. Tommy Boy', '8. Saludos Amigos', '9. Niagara, Niagara'. Please rank these 10 movies according to my watching history. Please only output the ranking results with order numbers. Split your output with line break.

Output: 1. Tommy Boy \n 2. Wayne's World 2 \n 3. Mrs. Brown (Her Majesty, Mrs. Brown) \n 4. Saludos Amigos \n 5. Niagara, Niagara \n 6. Wisdom \n 7. Guantanamera \n 8. Police Academy \n 4: Citizens on Patrol \n 9. Pink Flamingos \n 10. Seven Samurai (The Magnificent Seven) (Shichininno samurai) \n

Input: I've watched the following movies in the past in order:'0. Arthur', '1. Terms of Endearment'. Now there are 10 candidate movies that I can watch next: '0. Mrs. Brown (Her Majesty, Mrs. Brown)', '1. Police Academy 4: Citizens on Patrol', '2. Wisdom', '3. Guantanamera', '4. Seven Samurai (The Magnificent Seven) (Shichinin no samurai)', '5. Pink Flamingos', "6. Wayne's World 2", '7. Tommy Boy', '8. Saludos Amigos', '9. Niagara, Niagara'. Please rank these 10 movies according to my watching history. Please only output the ranking results with order numbers. Split your output with line break.

Output: 1. Seven Samurai (The Magnificent Seven) (Shichinin no samurai) \n 2. Pink Flamingos \n 3. Wayne's World 2 \n 4. Tommy Boy \n 5. Wisdom \n 6. Mrs. Brown (Her Majesty, Mrs. Brown) \n 7. Guantanamera \n 8. Saludos Amigos \n 9. Niagara, Niagara \n 10. Police Academy 4: Citizens on Patrol \n

Fig. 8. The case study on "history records" patterns. Variations of prompt pairs are marked green, the candidate items are marked blue, and the ground truth is marked red.

Formally, let N denote the number of users, L denote the length of a prompt, D denote the possible tokens, and S denote the limited actions. RPP/RPP+ reduces the timing and computational complexity of designing prompts from $O(NLD)$ to $O(NS)$. Our experiments reveal that ChatGPT can complete prompt optimization in approximately three rounds, whereas LLaMa2-7B-chat requires two, seven, and eight iterations on three distinct datasets (i.e., ML-1M, Games, and Lastfm), and Alpaca takes around eight iterations. The timing and computational complexity are acceptable for achieving significant performance improvements. It reveals that RPP and RPP+ can achieve a balance between prompt quality and resource efficiency.
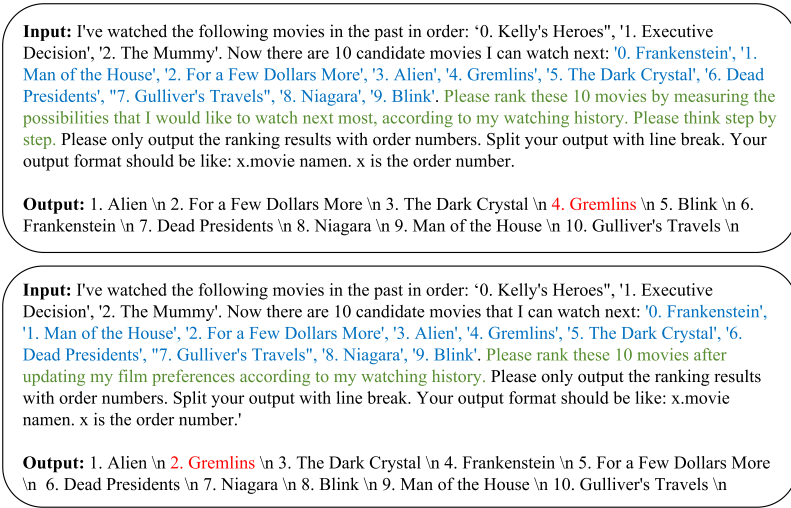
Input: I've watched the following movies in the past in order: '0. Kelly's Heroes", '1. Executive Decision', '2. The Mummy'. Now there are 10 candidate movies I can watch next: '0. Frankenstein', '1. Man of the House', '2. For a Few Dollars More', '3. Alien', '4. Gremlins', '5. The Dark Crystal', '6. Dead Presidents', "7. Gulliver's Travels", '8. Niagara', '9. Blink'. Please rank these 10 movies by measuring the possibilities that I would like to watch next most, according to my watching history. Please think step by step. Please only output the ranking results with order numbers. Split your output with line break. Your output format should be like: x.movie namen. x is the order number.

Output: 1. Alien \n 2. For a Few Dollars More \n 3. The Dark Crystal \n 4. Gremlins \n 5. Blink \n 6. Frankenstein \n 7. Dead Presidents \n 8. Niagara \n 9. Man of the House \n 10. Gulliver's Travels \n

Input: I've watched the following movies in the past in order: '0. Kelly's Heroes", '1. Executive Decision', '2. The Mummy'. Now there are 10 candidate movies that I can watch next: '0. Frankenstein', '1. Man of the House', '2. For a Few Dollars More', '3. Alien', '4. Gremlins', '5. The Dark Crystal', '6. Dead Presidents', "7. Gulliver's Travels", '8. Niagara', '9. Blink'. Please rank these 10 movies after updating my film preferences according to my watching history. Please only output the ranking results with order numbers. Split your output with line break. Your output format should be like: x.movie namen. x is the order number.'

Output: 1. Alien \n 2. Gremlins \n 3. The Dark Crystal \n 4. Frankenstein \n 5. For a Few Dollars More \n 6. Dead Presidents \n 7. Niagara \n 8. Blink \n 9. Man of the House \n 10. Gulliver's Travels \n

Fig. 9. The case study on "reasoning guidance" patterns. Variations of prompt pairs are marked green, the candidate items are marked blue, and the ground truth is marked red.
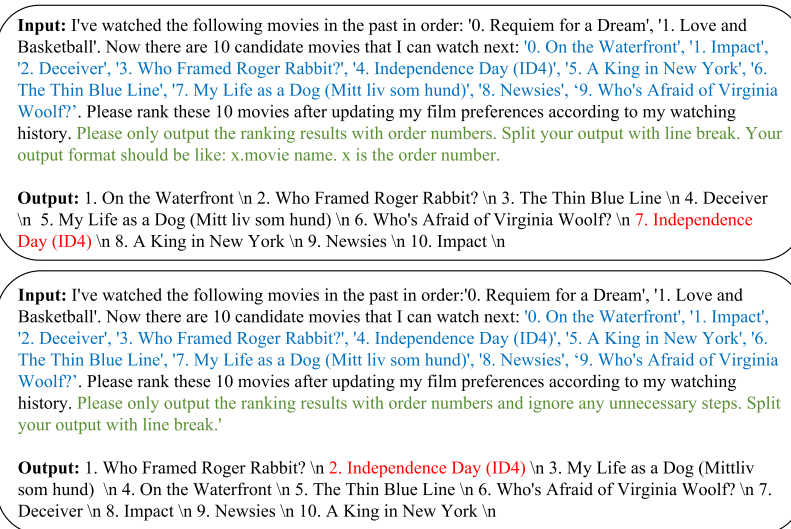
Input: I've watched the following movies in the past in order: '0. Requiem for a Dream', '1. Love and Basketball'. Now there are 10 candidate movies that I can watch next: '0. On the Waterfront', '1. Impact', '2. Deceiver', '3. Who Framed Roger Rabbit?', '4. Independence Day (ID4)', '5. A King in New York', '6. The Thin Blue Line', '7. My Life as a Dog (Mitt liv som hund)', '8. Newsies', '9. Who's Afraid of Virginia Woolf?'. Please rank these 10 movies after updating my film preferences according to my watching history. Please only output the ranking results with order numbers. Split your output with line break. Your output format should be like: x.movie name. x is the order number.

Output: 1. On the Waterfront \n 2. Who Framed Roger Rabbit? \n 3. The Thin Blue Line \n 4. Deceiver \n 5. My Life as a Dog (Mitt liv som hund) \n 6. Who's Afraid of Virginia Woolf? \n 7. Independence Day (ID4) \n 8. A King in New York \n 9. Newsies \n 10. Impact \n

Input: I've watched the following movies in the past in order:'0. Requiem for a Dream', '1. Love and Basketball'. Now there are 10 candidate movies that I can watch next: '0. On the Waterfront', '1. Impact', '2. Deceiver', '3. Who Framed Roger Rabbit?', '4. Independence Day (ID4)', '5. A King in New York', '6. The Thin Blue Line', '7. My Life as a Dog (Mitt liv som hund)', '8. Newsies', '9. Who's Afraid of Virginia Woolf?'. Please rank these 10 movies after updating my film preferences according to my watching history. Please only output the ranking results with order numbers and ignore any unnecessary steps. Split your output with line break.'

Output: 1. Who Framed Roger Rabbit? \n 2. Independence Day (ID4) \n 3. My Life as a Dog (Mittliv som hund) \n 4. On the Waterfront \n 5. The Thin Blue Line \n 6. Who's Afraid of Virginia Woolf? \n 7. Deceiver \n 8. Impact \n 9. Newsies \n 10. A King in New York \n

Fig. 10. The case study on "output format" patterns. Variations of prompt pairs are marked green, the candidate items are marked blue, and the ground truth is marked red.

5 Related Work

Here, we provide a review regarding LLMs for RSs, prompt sensitivity of LLMs, and discrete prompt engineering.

5.1 LLMs for RSs

Recently, LLMs have demonstrated remarkable capabilities in instruction following and intent reasoning, drawing researchers' attention to using LLMs as RSs [8, 36, 57, 58] which can capture
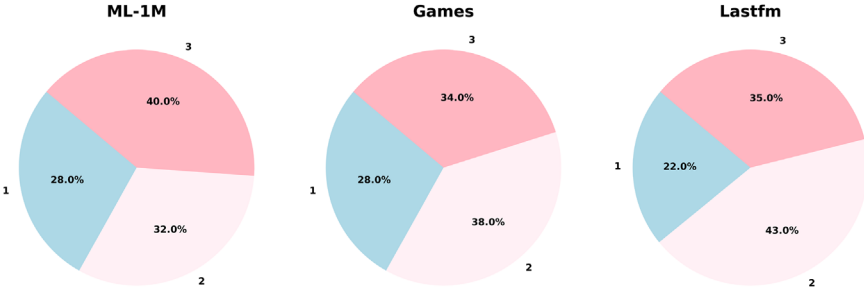
Fig. 11. The user distribution on personalized "role-playing" pattern, with three sectors representing the three expressions for this pattern.
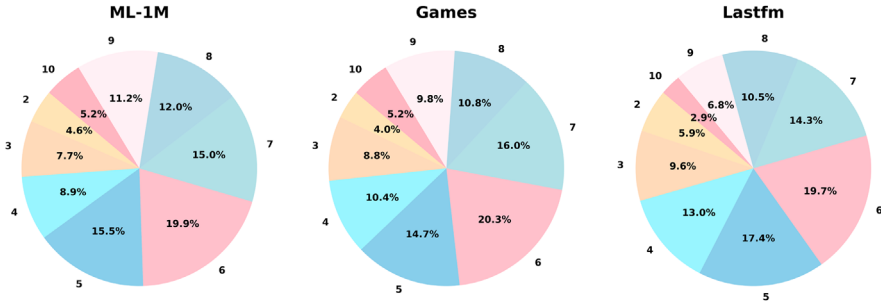

Fig. 12. The user distribution on personalized "history records" pattern, with each sector representing varied interaction history length.
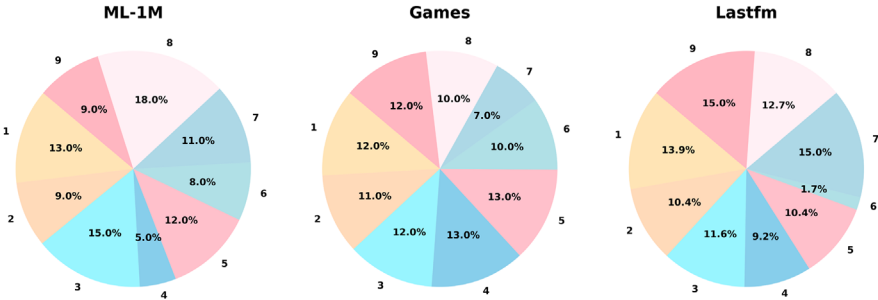

Fig. 13. The user distribution on personalized "reasoning guidance" pattern, with nine sectors representing the nine expressions for this pattern.

user preference and recommend for them. Conceptualizing LLMs as RSs to generate recommendations relies heavily on the quality of the prompts designed. Tailoring prompts of LLMs for the recommendation tasks has been explored by recent research. InstructRec [62] develops a general instruction format that contains users' preferences, intentions, task requirements, and contextual information. Chat-REC [10] recasts user profiles and prior interactions into prompts, enhancing the interactivity and explainability of the recommendation process. LLM4RS [4] designs domain-specific prompt format to analyze ChatGPT's recommendation ability on all three ranking policies, including point-wise, pairwise, and list-wise ranking. Instead of designing a common prompt
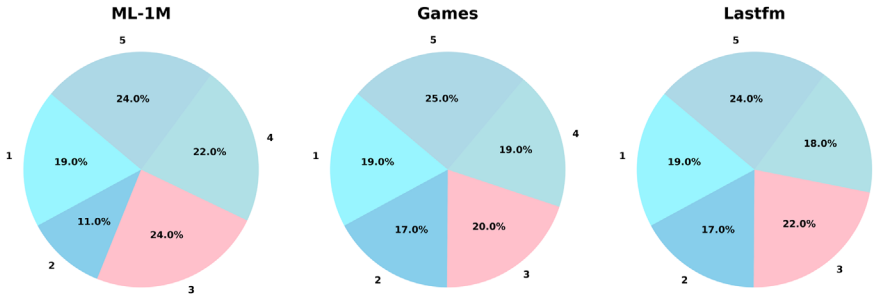
Fig. 14. The user distribution on personalized "output format" pattern, with five sectors representing the five expressions for this pattern.

format for the specific recommendation task, we personalize instance-wise prompt for each user to explore users' different preferences. Though existing studies [51, 55, 61] have explored optimizing prompts for users, they may be limited in their reliance on LLMs to refine or partial optimization from limited perspectives. For example, PepRec [61] refines the initial prompts by incorporating interactions from similar users in the same cluster iteratively. Re2LLM [55] and ISR [51] correct LLMs' recommendation errors through self-reflection and reasoning. Different from them, our method decomposes the completed prompt into multiple key patterns and optimizes each pattern comprehensively with RL.

5.2 Prompt Sensitivity of LLMs

Recent studies [34] have examined the sensitivity of LLMs to prompts that even minor variations in prompts can lead to significantly different outputs, indicating that the one-size-fits-all prompt may not generalize well [26, 37, 50] for different inputs. For example, KATE [31] confirmed that the number and order of in-context examples can influence the output of LLMs. Besides, variations in prompt formats within task descriptions may result in LLMs interpreting input instances differently, consequently influencing the outcomes [66]. These studies underscore the importance of optimizing prompts for specific inputs to enhance the performance of LLMs. Fixed prompt templates that assume uniformity across all samples in a task, regardless of their varying difficulty, may impair LLMs' abilities due to a lack of instance-specific knowledge [18]. To address the difference in inputs, recent research has proposed instance-level prompt optimization [20], which involves rewriting the prompt for each input to better leverage the LLM's capabilities for specific instances. IPL [20] assumed that each learnable prompt token contributes differently to various instances, which can be learned by calculating relevance scores. To meet the specific demands for personalization in recommendation tasks, we enhance LLMs' personalized recommendations by leveraging instance-wise prompts and introducing a method to optimize these prompts using MARL.

5.3 Discrete Prompt Engineering

Several approaches have explored prompt engineering in continuous and discrete space to obtain better answers from LLMs. Since the prompts for recent LLMs (ChatGPT, LLaMa2) are discrete and hard prompts, our work is strongly related to the discrete prompts engineering. Prior works have attempted to construct discrete prompts manually [40] or by heuristic [19], which are limited by human effort or resource consumption. Recent research has explored constructing prompts automatically. For instance, prompt retrieval [44] leverages annotated data and a language model, training a compact retriever to retrieve prompts based on input-output pairs. AutoPrompt [48] utilizes gradient-guided search to identify the optimal tokens within the prompt, although these

prompts are typically not human-interpretable. RLPrompt [5] introduces a framework based on RL to generate prompts iteratively word-by-word from the vast vocabulary. An important distinction between our method and the existing methods is that we decompose the completed prompt into multiple key patterns, optimizing each pattern and concatenating them, which can enhance the efficiency of optimization and ensure the quality of prompts. To the best of our knowledge, our work is the first to explore instance-wise prompting in the domain of RSs.

6 Potential Application and Limitations

In real-world applications, after the initial coarse ranking process, RPP/RPP+ can be utilized in the fine-ranking stage to deliver highly relevant and personalized recommendation results to users based on their preferences and interests. By employing personalized prompts, LLMs can conduct more refined ranking, leading to a more accurate understanding of users' underlying preferences. Additionally, the personalized interaction history length can capture the dynamic changes in users' interests within sequence recommendation scenarios. Furthermore, the personalized reasoning process, as realized by the pattern of "recipe," can improve the interpretability of recommendations.

While the proposed RPP/RPP+ framework presents an efficient solution for personalizing instance-wise prompts for recommendation, it is accompanied by certain limitations. A notable limitation is its partial reliance on manual intervention when designing the action set in the initial setup phase. This suggests a potential area for future research to develop more autonomous and adaptive methods. Another limitation is the efficiency of RPP/RPP+ in narrowing the search space through iterative updates, which necessitates a certain level of computational resources. In the future, we will explore a more efficient method for prompt personalization, preferably without relying on LLMs in the iterations.

7 Conclusion

In this article, we propose instance-wise prompting to personalize prompts for individual users in recommendation tasks. To solve this task, we introduce a new framework, RPP/RPP+, personalizing essential patterns respectively with multi-agent and then concatenating them into a personalized prompt. We optimize the prompts at the sentence level and carefully design the expressions for each pattern, enhancing search efficiency and ensuring prompt quality. Experimental results validate the superiority of RPP/RPP+ over several traditional recommender models, few-shot methods, and prompt-based methods in ranking tasks, demonstrating the potential of personalizing instance-wise prompts.

References

[1] 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 Conference on Recommender Systems (RecSys)*. ACM, New York, NY, 1007–1014.

[2] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. In *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS)*, 1877–1901.

[3] Iván Cantador, Peter Brusilovsky, and Tsvi Kuflik. 2011. Second workshop on information heterogeneity and fusion in recommender systems (HetRec2011). In *Proceedings of the Conference on Recommender Systems (RecSys)*. ACM, New York, NY, 387–388.

[4] Sunhao Dai, Ninglu Shao, Haiyuan Zhao, Weijie Yu, Zihua Si, Chen Xu, Zhongxiang Sun, Xiao Zhang, and Jun Xu. 2023. Uncovering ChatGPT's capabilities in recommender systems. In *Proceedings of the Conference on Recommender Systems (RecSys)*. ACM, New York, NY, 1126–1132.

[5] Mingkai Deng, Jianyu Wang, Cheng-Ping Hsieh, Yihan Wang, Han Guo, Tianmin Shu, Meng Song, Eric P. Xing, and Zhiting Hu. 2022. RLPrompt: Optimizing discrete text prompts with reinforcement learning. In *Proceedings of the*

Conference on Empirical Methods in Natural Language Processing (EMNLP). Association for Computational Linguistics, 3369–3391.

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 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: Human Language Technologies (NAACL-HLT (1))*. Association for Computational Linguistics, 4171–4186.

[7] Chenlu Ding, Jiancan Wu, Yancheng Yuan, Jinda Lu, Kai Zhang, Alex Su, Xiang Wang, and Xiangnan He. 2024. Unified parameter-efficient unlearning for LLMs. arXiv:2412.00383. Retrieved from https://arxiv.org/abs/2412.00383

[8] Chongming Gao, Shiqi Wang, Shijun Li, Jiawei Chen, Xiangnan He, Wenqiang Lei, Biao Li, Yuan Zhang, and Peng Jiang. 2024. CIRS: Bursting filter bubbles by counterfactual interactive recommender system. *ACM Trans. Inf. Syst*. 42, 1 (2024), 14:1–14:27.

[9] Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (ACL/IJCNLP (1))*. Association for Computational Linguistics, 3816–3830.

[10] 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:2303.14524. Retrieved from https://arxiv.org/abs/abs/2303.14524

[11] Shijie Geng, Shuchang Liu, Zuohui Fu, Yingqiang Ge, and Yongfeng Zhang. 2022. Recommendation as language processing (RLP): A unified pretrain, personalized prompt & predict paradigm (P5). In *Proceedings of the Conference on Recommender Systems (RecSys)*. ACM, New York, NY, 299–315.

[12] F. Maxwell Harper and Joseph A. Konstan. 2016. The MovieLens datasets: History and context. *ACM Trans. Interact. Intell. Syst*. 5, 4 (2016), 19:1–19:19.

[13] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yong-Dong Zhang, and Meng Wang. 2020. LightGCN: Simplifying and powering graph convolution network for recommendation. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. ACM, New York, NY, 639–648.

[14] Yupeng Hou, Zhankui He, Julian J. McAuley, and Wayne Xin Zhao. 2023. Learning vector-quantized item representation for transferable sequential recommenders. In *Proceedings of the ACM Web Conference (WWW)*. ACM, New York, NY, 1162–1171.

[15] Yupeng Hou, Shanlei Mu, Wayne Xin Zhao, Yaliang Li, Bolin Ding, and Ji-Rong Wen. 2022. Towards universal sequence representation learning for recommender systems. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, 585–593.

[16] Yupeng Hou, Junjie Zhang, Zihan Lin, Hongyu Lu, Ruobing Xie, Julian J. McAuley, and Wayne Xin Zhao. 2024. Large language models are zero-shot rankers for recommender systems. In *Proceedings of the European Conference on Information Retrieval (ECIR (2))*. Lecture Notes in Computer Science, Vol. 14609, Springer, 364–381.

[17] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *Proceedings of the International Conference on Learning Representations (ICLR)*. Vol. 1, OpenReview.net, 3 pages.

[18] Yuezihan Jiang, Hao Yang, Junyang Lin, Hanyu Zhao, An Yang, Chang Zhou, Hongxia Yang, Zhi Yang, and Bin Cui. 2022. Instance-wise prompt tuning for pretrained language models. arXiv:2206.01958. Retrieved from https://arxiv.org/abs/2206.01958

[19] Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know. *Trans. Assoc. Comput. Linguist*. 8 (2020), 423–438.

[20] Feihu Jin, Jinliang Lu, Jiajun Zhang, and Chengqing Zong. 2023. Instance-aware prompt learning for language understanding and generation. *ACM Trans. Asian Low Resour. Lang. Inf. Process*. 22, 7 (2023), 199:1–199:18.

[21] Hideaki Joko, Shubham Chatterjee, Andrew Ramsay, Arjen P. de Vries, Jeff Dalton, and Faegheh Hasibi. 2024. Doing personal LAPS: LLM-augmented dialogue construction for personalized multi-session conversational search. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. ACM, New York, NY, 796–806.

[22] Wang-Cheng Kang and Julian J. McAuley. 2018. Self-attentive sequential recommendation. In *Proceedings of the IEEE International Conference on Data Mining (ICDM)*. IEEE Computer Society, 197–206.

[23] Yufei Kuang, Jie Wang, Haoyang Liu, Fangzhou Zhu, Xijun Li, Jia Zeng, Hao Jianye, Bin Li, and Feng Wu. 2024. Rethinking branching on exact combinatorial optimization solver: The first deep symbolic discovery framework. In *Proceedings of the 12th International Conference on Learning Representations*.

[24] Xinyi Li, Yongfeng Zhang, and Edward C. Malthouse. 2023. A preliminary study of ChatGPT on news recommendation: Personalization, provider fairness, and fake news. In *Proceedings of INRA@RecSys*. CEUR Workshop Proceedings, Vol. 3561, CEUR-WS.org.

[25] Jiayi Liao, Sihang Li, Zhengyi Yang, Jiancan Wu, Yancheng Yuan, Xiang Wang, and Xiangnan He. 2024. LLaRA: Large language-recommendation assistant. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. ACM, New York, NY, 1785–1795.

[26] Haoyang Liu, Yufei Kuang, Jie Wang, Xijun Li, Yongdong Zhang, and Feng Wu. 2023. Promoting generalization for exact solvers via adversarial instance augmentation. arXiv:2310.14161. Retrieved from https://arxiv.org/abs/2310.14161

[27] Hongyu Liu, Haoyang Liu, Yufei Kuang, Jie Wang, and Bin Li. 2024. Deep symbolic optimization for combinatorial optimization: Accelerating node selection by discovering potential heuristics. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, 2067–2075.

[28] Haoyang Liu, Jie Wang, Zijie Geng, Xijun Li, Yuxuan Zong, Fangzhou Zhu, Jianye Hao, and Feng Wu. 2025. Apollo-MILP: An alternating prediction-correction neural solving framework for mixed-integer linear programming. In *Proceedings of the 13th International Conference on Learning Representations*. Retrieved from https://openreview.net/forum?id=mFY0tPDWK8

[29] Haoyang Liu, Jie Wang, Wanbo Zhang, Zijie Geng, Yufei Kuang, Xijun Li, Bin Li, Yongdong Zhang, and Feng Wu. 2024. MILP-StuDio: MILP instance generation via block structure decomposition. In *Proceedings of the 38th Annual Conference on Neural Information Processing Systems*. Retrieved from https://openreview.net/forum?id=W433RI0VU4

[30] Junling Liu, Chao Liu, Renjie Lv, Kang Zhou, and Yan Zhang. 2023. Is ChatGPT a good recommender? A preliminary study. arXiv:2304.10149. Retrieved from https://arxiv.org/abs/2304.10149

[31] Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. 2022. What makes good in-context examples for GPT-3? In *Proceedings of the DeeLIO@ACL*. Association for Computational Linguistics, 100–114.

[32] Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2022. P-tuning: Prompt tuning can be comparable to fine-tuning across scales and tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL (2))*. Association for Computational Linguistics, 61–68.

[33] Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. 2017. Multi-agent actor-critic for mixed cooperative-competitive environments. In *Proceedings of the Neural Information Processing Systems (NIPS)*, 6379–6390.

[34] Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2022. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL (1))*. Association for Computational Linguistics, 8086–8098.

[35] Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. 2023. Self-refine: Iterative refinement with self-feedback. In *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS)*, 46534–46594.

[36] Wenyu Mao, Shuchang Liu, Haoyang Liu, Haozhe Liu, Xiang Li, and Lanatao Hu. 2025. Distinguished quantized guidance for diffusion-based sequence recommendation. arXiv:2501.17670. Retrieved from https://arxiv.org/abs/2501.17670

[37] Wenyu Mao, Jiancan Wu, Haoyang Liu, Yongduo Sui, and Xiang Wang. 2024. Invariant graph learning meets information bottleneck for out-of-distribution generalization. arXiv:2408.01697. Retrieved from https://arxiv.org/abs/2408.01697

[38] Jianmo Ni, Jiacheng Li, and Julian J. McAuley. 2019. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP/IJCNLP (1))*. Association for Computational Linguistics, 188–197.

[39] Baolin Peng, Xiujun Li, Jianfeng Gao, Jingjing Liu, Yun-Nung Chen, and Kam-Fai Wong. 2018. Adversarial advantage actor-critic model for task-completion dialogue policy learning. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 6149–6153.

[40] Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick S. H. Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander H. Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP/IJCNLP (1))*. Association for Computational Linguistics, 2463–2473.

[41] Archiki Prasad, Peter Hase, Xiang Zhou, and Mohit Bansal. 2023. GrIPS: Gradient-free, edit-based instruction search for prompting large language models. In *Proceedings of the European Chapter of the Association for Computational Linguistics (EACL)*. Association for Computational Linguistics, 3827–3846.

[42] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the Uncertainty in Artificial Intelligence (UAI)*. AUAI Press, 452–461.

[43] Stephen E. Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: BM25 and beyond. *Found. Trends Inf. Retr*. 3, 4 (2009), 333–389.

[44] Ohad Rubin, Jonathan Herzig, and Jonathan Berant. 2022. Learning to retrieve prompts for in-context learning. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*. Association for Computational Linguistics, 2655–2671.

[45] Timo Schick and Hinrich Schütze. 2021. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the European Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, 255–269.

[46] Murray Shanahan, Kyle McDonell, and Laria Reynolds. 2023. Role play with large language models. *Nature* 623, 7987 (2023), 493–498.

[47] Yunfan Shao, Linyang Li, Junqi Dai, and Xipeng Qiu. 2023. Character-LLM: A trainable agent for role-playing. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, 13153–13187.

[48] Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting knowledge from language models with automatically generated prompts. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP (1))*. Association for Computational Linguistics, 4222–4235.

[49] Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: Language agents with verbal reinforcement learning. In *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS)*, Vol. 36, 8634–8652.

[50] Yongduo Sui, Wenyu Mao, Shuyao Wang, Xiang Wang, Jiancan Wu, Xiangnan He, and Tat-Seng Chua. 2024. Enhancing out-of-distribution generalization on graphs via causal attention learning. *ACM Trans. Knowl. Discov. Data* 18, 5 (2024), 127:1–127:24.

[51] Zhu Sun, Hongyang Liu, Xinghua Qu, Kaidong Feng, Yan Wang, and Yew Soon Ong. 2024. Large language models for intent-driven session recommendations. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. ACM, New York, NY, 324–334.

[52] Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinícius Flores Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Z. Leibo, Karl Tuyls, et al. 2018. Value-decomposition networks for cooperative multi-agent learning based on team reward. In *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC/ACM, 2085–2087.

[53] Lei Wang and Ee-Peng Lim. 2023. Zero-shot next-item recommendation using large pretrained language models. arXiv:2304.03153. Retrieved from https://arxiv.org/abs/2304.03153

[54] Xiaobei Wang, Shuchang Liu, Xueliang Wang, Qingpeng Cai, Lantao Hu, Han Li, Peng Jiang, Kun Gai, and Guangming Xie. 2024. Future impact decomposition in request-level recommendations. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*. ACM, New York, NY, 5905–5916.

[55] Ziyan Wang, Yingpeng Du, Zhu Sun, Haoyan Chua, Kaidong Feng, Wenya Wang, and Jie Zhang. 2024. Re2LLM: Reflective reinforcement large language model for session-based recommendation. arXiv:2403.16427. Retrieved from https://arxiv.org/abs/2403.16427

[56] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS)*, Vol. 35, 24824–24837.

[57] Jiancan Wu, Xiangnan He, Xiang Wang, Qifan Wang, Weijian Chen, Jianxun Lian, and Xing Xie. 2022. Graph convolution machine for context-aware recommender system. *Front. Comput. Sci.* 16, 6 (2022), 166614.

[58] Jiancan Wu, Xiang Wang, Xingyu Gao, Jiawei Chen, Hongcheng Fu, and Tianyu Qiu. 2024. On the effectiveness of sampled Softmax loss for item recommendation. *ACM Trans. Inf. Syst.* 42, 4 (2024), 98:1–98:26.

[59] Jiancan Wu, Yi Yang, Yuchun Qian, Yongduo Sui, Xiang Wang, and Xiangnan He. 2023. GIF: A general graph unlearning strategy via influence function. In *Proceedings of the ACM Web Conference (WWW)*. ACM, New York, NY, 651–661.

[60] Likang Wu, Zhi Zheng, Zhaopeng Qiu, Hao Wang, Hongchao Gu, Tingjia Shen, Chuan Qin, Chen Zhu, Hengshu Zhu, Qi Liu, et al. 2023. A survey on large language models for recommendation. arXiv:2305.19860. Retrieved from https://arxiv.org/abs/2305.19860

[61] Yakun Yu, Shiang Qi, Baochun Li, and Di Niu. 2024. PepRec: Progressive enhancement of prompting for recommendation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, 17941–17953.

[62] 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:2305.07001. Retrieved from https://arxiv.org/abs/2305.07001

[63] Tianjun Zhang, Xuezhi Wang, Denny Zhou, Dale Schuurmans, and Joseph E. Gonzalez. 2023. TEMPERA: Test-time prompt editing via reinforcement learning. In *Proceedings of the International Conference on Learning Representations (ICLR)*. OpenReview.net.

[64] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A survey of large language models. arXiv:2303.18223. Retrieved from https://arxiv.org/abs/2303.18223

[65] Yuyue Zhao, Jiancan Wu, Xiang Wang, Wei Tang, Dingxian Wang, and Maarten de Rijke. 2024. Let me do it for you: Towards LLM empowered recommendation via tool learning. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. ACM, New York, NY, 1796–1806.

[66] Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. *Proceedings of the International Conference on Machine Learning (ICML)*. Proceedings of Machine Learning Research, Vol. 139, PMLR, 12697–12706.