

# Revisiting Prompt Engineering: A Comprehensive Evaluation for LLM-based Personalized Recommendation

Genki Kusano  
g-kusano@nec.com  
NEC Corporation  
Kawasaki, Japan

Kosuke Akimoto  
kosuke\_a@nec.com  
NEC Corporation  
Kawasaki, Japan

Kunihiro Takeoka  
k\_takeoka@nec.com  
NEC Corporation  
Kawasaki, Japan

## Abstract

Large language models (LLMs) can perform recommendation tasks by taking prompts written in natural language as input. Compared to traditional methods such as collaborative filtering, LLM-based recommendation offers advantages in handling cold-start, cross-domain, and zero-shot scenarios, as well as supporting flexible input formats and generating explanations of user behavior. In this paper, we focus on a single-user setting, where no information from other users is used. This setting is practical for privacy-sensitive or data-limited applications. In such cases, prompt engineering becomes especially important for controlling the output generated by the LLM. We conduct a large-scale comparison of 23 prompt types across 8 public datasets and 12 LLMs. We use statistical tests and linear mixed-effects models to evaluate both accuracy and inference cost. Our results show that for cost-efficient LLMs, three types of prompts are especially effective: those that rephrase instructions, consider background knowledge, and make the reasoning process easier to follow. For high-performance LLMs, simple prompts often outperform more complex ones while reducing cost. In contrast, commonly used prompting styles in natural language processing, such as step-by-step reasoning, or the use of reasoning models often lead to lower accuracy. Based on these findings, we provide practical suggestions for selecting prompts and LLMs depending on the required balance between accuracy and cost.

This paper has been accepted to ACM RecSys 2025. Please cite it appropriately after September 22, 2025.

## CCS Concepts

- Information systems → Recommender systems.

## Keywords

LLM-based Recommendation, Prompt Evaluation, Statistical Analysis, Reasoning

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

Conference acronym 'XX, Woodstock, NY

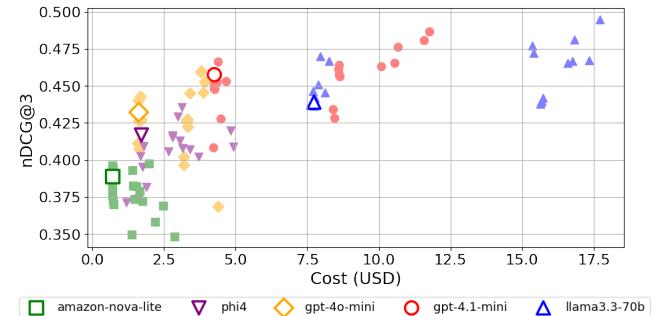
© 2018 Copyright held by the owner/author(s). Publication rights licensed to ACM.  
ACM ISBN 978-1-4503-XXXX-X/2018/06  
<https://doi.org/XXXXXXX.XXXXXXX>

## ACM Reference Format:

Genki Kusano, Kosuke Akimoto, and Kunihiro Takeoka. 2018. Revisiting Prompt Engineering: A Comprehensive Evaluation for LLM-based Personalized Recommendation. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation email (Conference acronym 'XX)*. ACM, New York, NY, USA, 30 pages. <https://doi.org/XXXXXXX.XXXXXXX>

## 1 Introduction

Recommender systems are widely used in services such as online advertising, e-commerce, and video streaming. Traditional approaches, such as collaborative filtering, predict user behavior by learning patterns from many users' histories [7, 18, 25]. These methods work well when there is a large amount of training data. However, their performance decreases in cold-start situations [54, 66, 67] where the system has not seen the target items or users before, and also in cross-domain situations [22, 36, 62] where the system is applied to a different domain from its original training domain, such as recommending books after learning from movie data.



**Figure 1: Recommendation accuracy (nDCG@3) and inference cost (USD for processing 1,600 users from 8 datasets) for 21 prompts, including the baseline, evaluated with five LLMs in Section 4. Hollow and filled markers represent the baseline and 20 prompts from previous studies, respectively.**

A new approach that can address these limitations is the use of large language models (LLMs). When a recommendation task is described in natural language, an LLM can return meaningful results without additional training [11, 13, 29, 30, 32, 56, 57]. These models support a wide range of input formats and generate fluent responses, which was not possible in earlier methods. They can also handle item attributes that are not included in the training data and provide informative explanations for their inferences.

Prompt engineering is also used in recommendation, although it does not always lead to better accuracy. For example, techniques

developed in natural language processing (NLP), such as step-by-step reasoning [23] and rephrasing [12], have been effective in tasks like question answering and summarization. Recommendation tasks, however, require different types of reasoning that emphasize the relationship between users and items. Several studies have examined prompts in both general NLP [42, 47] and recommendation [9, 15, 19, 20, 31, 48, 49, 57, 64]. Research on recommendation has mostly focused on prompt types proposed in the RecSys field. It often includes few techniques from NLP and covers a limited range of datasets or LLMs. As a result, the conclusions from these studies may not generalize well to broader recommendation settings.

To address these gaps, we conduct a broad evaluation of prompts for LLM-based recommendation. Our study compares **23 prompt types, 8 real-world datasets, and 12 LLMs**, which is much larger in scale than previous studies. This paper focuses on a personalized setting, where the LLM uses only one user's interaction history and does not rely on data from others. While incorporating other users may add information, it would also increase the complexity of the experimental design. By limiting the input to a single user, we can more directly assess the effect of each prompt on recommendation performance, without being influenced by other users.

In the experiments, we first evaluated five cost-efficient LLMs (Figure 1), such as gpt-4.1-mini and llama3.3-70b. We used hypothesis testing and linear mixed-effects models to identify which prompts led to better recommendation accuracy. The results showed that three prompt types worked well across different datasets and LLMs. These prompts encouraged rephrasing the input [12], considering background knowledge [63], or clarifying the task goal using a specific thinking process [60]. In contrast, step-by-step reasoning [23], which is often said to be effective in NLP, did not lead to higher accuracy in recommendation.

We then analyzed which combinations of prompts and LLMs were effective in terms of both accuracy and inference cost. In this experiment, we added high-performance LLMs (e.g., gpt-4.1 and claude-3.7-sonnet) and reasoning models (e.g., o3 and o4-mini) to the comparison. In accuracy-focused cases, claude-3.7-sonnet reached the highest accuracy. However, the cost increased sharply when more complex prompts were used, even though their accuracy was similar to that of the simple baseline prompt. For this reason, **if accuracy is the top priority, using the simple baseline prompt with claude-3.7-sonnet is more effective than relying on more complex prompt designs**. On the other hand, **if cost is also a concern, a better option is to use gpt-4.1-mini or llama3.3-70b with one of the three effective prompts** [12, 60, 63]. This combination reached about 90% of the accuracy of claude-3.7-sonnet, at less than one fifth of the cost.

As an additional observation, we found that reasoning models such as o3 showed relatively strong performance. However, their accuracy did not exceed that of claude-3.7-sonnet, and the inference cost was higher. To explore the impact of reasoning, we further evaluated internal reasoning settings by activating the thinking mode in claude-3.7-sonnet and applying the three effective prompts to o3-mini. In some cases, these settings led to lower accuracy than the baseline prompt. These results suggest that **adding more reasoning does not always improve performance in recommendation tasks and may sometimes reduce accuracy**.

This study provides practical guidance on selecting prompts and LLMs for personalized LLM-based recommendation. All code is released on [https://github.com/nec-research-labs/recsys2025\\_reproducibility\\_prompts](https://github.com/nec-research-labs/recsys2025_reproducibility_prompts).

## 2 Related Work

*NLP-Based Recommendation.* Natural language processing (NLP) has contributed to the development of recommender systems. Earlier techniques applied word embeddings such as word2vec to represent items based on user interactions, as seen in Prod2Vec [16] and Item2Vec [3]. Subsequent approaches introduced transformer-based models such as BERT to handle items as tokens, as in BERT4Rec [44]. LLMs have built on this work by providing contextual understanding of both users and items and by generating outputs in natural language. Pretrained language models like T5 or LLaMA have been used to encode user reviews and product descriptions [2, 28, 61].

*Prompt Engineering for Recommendation.* Prompt-based recommendation has received attention for its ability to perform without labeled data or fine-tuning, making it suitable for zero-shot settings. Early studies applied generative models such as ChatGPT to recommendation tasks and explored various prompt styles [15, 31]. Subsequent work proposed methods to improve accuracy, including listwise input formats [9], user history summarization [48, 64], in-context learning with demonstrations [49], emphasis on recent user behavior [20], and dialogue-based modeling [19].

*Evaluation in Recommendation.* Reliable and reproducible evaluation is important for comparing approaches in recommendation. Previous studies have reexamined conventional methods such as matrix factorization [8, 38, 41, 45] and sequential recommendation [4, 10, 40], focusing on their reproducibility. Several studies have examined LLM-based approaches [11, 13, 30, 57], but they used only a small number of prompts or datasets. This paper presents a broader and more systematic comparison across different prompts, datasets, and LLMs.

## 3 Experimental Settings

To examine how prompt design influences recommendation accuracy, we conducted large-scale experiments using eight datasets, two user types, 12 LLMs, and standard ranking metrics.

### 3.1 Datasets

We used eight datasets: the Yelp Dataset<sup>1</sup>, the Microsoft News Recommendation Dataset (MIND) [55], the Food Dataset [35], and five categories from the Amazon Review Dataset [39]: Music, Movies, Groceries, Clothes, and Books.

In each dataset, users with 6 to 11 interactions were labeled as *light*, and those with 31 or more as *heavy*. The most recent item<sup>2</sup> was used as the positive example for inference, and the remaining interactions were used to construct the user prompt. We randomly selected 100 users for each dataset and user type, resulting in 1,600 users in total (8 datasets × 2 types × 100 users).

<sup>1</sup><https://www.kaggle.com/datasets/yelp-dataset/yelp-dataset>

<sup>2</sup>We use the term *item* to refer to entities such as news in MIND, recipes in Food, or businesses in Yelp.

To fairly compare prompt performance, we fixed the same ranking problem across all prompts by using a set of 10 items: one positive and nine negatives. Negative items were randomly selected from those the user had not interacted with. For MIND, these were news articles delivered but not clicked. For the Yelp dataset, we used businesses in the same state as the positive item. For the remaining datasets, we sampled items not included in the user’s history.

### 3.2 LLMs

We evaluated 12 LLMs from OpenAI, Meta, AWS, Microsoft, and Anthropic by fixing the temperature at 0.1. Based on cost and internal behavior, we classified the LLMs into three: low-cost, high-cost, and reasoning models. Table 1 summarizes the LLMs, including their versions, cost classes, and token prices. An LLM is classified as high-cost if its output token price exceeds 2 US dollars. `claude-3.7-sonnet` is used as a standard LLM by default, but treated as a reasoning model when the “thinking mode” is enabled. Reasoning models typically involve additional cost due to internal multi-step inference, and their actual cost is often several times higher than the listed token price.

All OpenAI models were accessed through the official API. Models from Meta, AWS, and Anthropic were accessed via Amazon Bedrock. `phi-4` was run locally, and its cost was estimated based on publicly available API pricing<sup>3</sup>.

**Table 1: LLMs used in this study, including their version, cost classes, and token prices (USD per 1M tokens).**

Provider	Model	Version	Cost Class	Input	Output
OpenAI	<code>gpt-4o-mini</code>	2024-07-18	Low	0.15	0.6
OpenAI	<code>gpt-4.1-mini</code>	2025-04-14	Low	0.4	1.6
OpenAI	<code>gpt-4.1</code>	2025-04-14	High	2.0	8.0
OpenAI	<code>o3-mini</code>	2025-01-31	Reasoning	1.1	4.0
OpenAI	<code>o4-mini</code>	2025-04-16	Reasoning	1.1	4.0
OpenAI	<code>o3</code>	2025-04-16	Reasoning	10	40
Meta	<code>llama3.3-70b</code>	v1(2024-12-19)	Low	0.72	0.72
AWS	<code>amazon-nova-lite</code>	v1(2024-12-04)	Low	0.06	0.24
Microsoft	<code>phi-4</code>	2024-12-12	Low	0.07	0.14
Anthropic	<code>claude-3.5-haiku</code>	20241022-v1	High	0.8	4.0
Anthropic	<code>claude-3.7-sonnet</code>	20250219-v1	High	3.0	15

### 3.3 Prompts

This section describes the prompts compared in this paper. LLMs typically use three main roles in conversation: system, user, and assistant, as seen in OpenAI and Anthropic models. Other LLMs, such as `llama3.3-70b` and `phi-4`, do not follow the same role names but support similar role-based prompting. For consistency, we use system, user, and assistant to describe all prompts in this paper. These LLMs also support multi-turn input, where earlier messages influence the response by providing context. Our experiments evaluate both single-turn and multi-turn prompt formats.

**3.3.1 Standardized Phrases.** To ensure fair evaluation across different prompts, we use a fixed wording pattern for all prompts, as shown in Table 2. The variables used in these phrases are explained as follows: The variable `{item}` is replaced with a domain-specific term: we use “product” for Amazon datasets, “business”

<sup>3</sup><https://deepinfra.com/microsoft/phi-4>. All pricing in Table 1 is as of March 13, 2025.

for Yelp, “news” for MIND, and “recipe” for Food. The variable `{user_item_history}` refers to the target user’s interaction history, which includes the title, category, and description of each item. The variable `{candidate_items}` represents a set of 10 candidate items. Unless otherwise stated, the system role is set as `default_sys_inst`. An example of the **Rephrase** prompt, which is introduced in the next section, is shown in Figure 5<sup>4</sup>.

**Table 2: Standardized phrasing used in prompts**

<code>default_sys_inst</code> :	<i>“You are an AI assistant that helps people find information”</i>
<code>user_info</code> :	<i>“# Requirements: you must rank candidate [item]s that will be provided below to the target user for recommendation. # Observation: {user_item_history}”</i>
<code>candidates_info</code> :	<i>“Based on the above user information, please rank the candidate [item]s that align closely with the user’s preferences. If item IDs [101, 102, 103] are sorted as first 102, second 103, third 101 in order, present your response in the format below: [3,1,2]. In this example, the number of candidate [item]s was 3, but next, 10 [item]s will be provided, so you SHOULD sort 10 [item]s. # Candidates [item]s: {candidate_items}”</i>
<code>task_inst</code> :	<i>“I here repeat candidate instruction. Please rank 10 candidate [item] IDs as follows: If [item] IDs [101, 102, 103] are sorted as first 102, second 103, third 101 in order, present your response in the format below: [3,1,2]. # Additional constraints: - You ONLY rank the given Candidate [item] IDs. - Do not explain the reason and include any other words.”</i>
<code>final_inst</code> :	<i>“Thank you! Based on the above conversation, please provide the final answer. {task_inst}”</i>
<code>preamble</code> :	<i>“Our final goal is to provide an answer to the following problem. Before tackling this issue,”</i>

**3.3.2 Non-Conversational Prompts.** We first introduce prompts that use a single instruction message  $t_{u1}$  from the user role without a conversational format, as shown in Table 3.

**3.3.3 Conversational Prompts.** We next introduce multi-turn conversational prompts. In this format, the user role first sends a message  $t_{u1}$ , and the assistant role replies with  $t_{a1}$ . Then, a second user role message  $t_{u2}$  is used to request the final output.

We begin with prompts where the first user message is set as  $t_{u1} = \text{user\_info} + \text{candidates\_info} + \text{middle\_inst}$ . Unless otherwise stated,  $t_{u2}$  is set to `final_inst`. Table 4 summarizes the variations used for `middle_inst`.

Table 5 summarizes prompts that include a preamble in the form  $t_{u1} = \text{user\_info} + \text{candidates\_info} + \text{preamble} + \text{post\_inst}$ , and presents the variations used for `post_inst`.

Table 6 presents three prompts that control the initial response of the LLM. Each prompt begins with a user message set as  $t_{u1} = \text{user\_info} + \text{candidates\_info}$ , followed by a fixed assistant reply  $t_{a1}$ . After this, we send  $t_{u2} = \text{“OK! Go ahead!”}$  to prompt the LLM to generate a full response  $t_{a2}$ . Finally,  $t_{u3} = \text{final\_inst}$  is used to request the ranking result.

<sup>4</sup>For more details on other prompts, please see Section A in the supplementary material.

```
tsys = (default_sys_inst)

tu1 = # Requirements: you must rank candidate products that will be
provided below to the target user for recommendation. # Observation:
{1: {'title': 'The Bourne Legacy Steelbook', 'categories': 'Blu-ray,
Movies', 'description': "NOTICE: Polish Release, cover may ...", 'rating':
5.0, 'review': 'awesome movie'}, 2 : (omitted) }
(candidate_info)
(preamble), rephrase and expand it to help you do better answering.
Maintain all information in the original question.

ta1 = The task at hand is to evaluate and rank a list of candidate
products based on their alignment with the preferences of a target
user. The user has previously shown a strong interest in action-
packed films, particularly those in the Bourne series, as well as
comedies with high ratings. The ranking should reflect the following
considerations: 1. The presence of action elements, particularly in
the context of the Bourne series. 2. (omitted)
Now, let's proceed to rank the candidate products based on these
criteria. (omitted)
Based on the analysis of these products, the final ranking will be
provided in the specified format. Now, let's rank the products. ####
Ranked List:1. **The Bourne Identity** (Action) - Strong alignment
with the user's interest in the Bourne series.(omitted)10. **Cooking
French, Vol. 2** (Special Interests) - Least relevant to the user's
interests. ### Final Output:[4, 9, 8, 6, 1, 3, 2, 7, 10, 5]

tu2 = (final_inst)

ta2 = [4, 9, 8, 6, 1, 3, 2, 7, 10, 5]
```

**Figure 2: Multi-turn conversation  $[t_{sys}, t_{u1}, t_{a1}, t_{u2}]$  used in the Rephrase prompt as input to the LLM, with output text  $t_{a2}$ .**

### 3.4 Evaluation Metrics and Output Formatting

In the numerical experiments, each prompt produced a ranking of 10 candidate items. We evaluated the ranking performance using two standard metrics: Hit@3 and nDCG@3<sup>5</sup>.

In some cases, LLMs returned only a partial ranking, including just a few top-ranked items. We preserved the order of these items and randomly ranked the rest to complete a full list of 10. If an LLM failed to return any ranking, we retried up to five times. After five failures, we used a fully random ranking.

## 4 Performance Evaluation for Prompts

This section evaluates the effectiveness of the 21 prompt types introduced in Section 3.3 in terms of recommendation accuracy. The goal is to understand which prompts are effective (RQ1), under what conditions (RQ2), and why (RQ3). To make large-scale inference feasible within budget constraints, we conduct experiments using the five low-cost LLMs listed in Table 1.

**RQ1:** Which prompts significantly outperform the baseline?

**RQ2:** Are there any prompts that show stable effectiveness across different LLMs?

**RQ3:** Why do some prompts result in lower accuracy?

<sup>5</sup>All tasks in this paper fix the same setting to re-rank 10 candidate items, so we use  $k = 3$  for evaluating nDCG@ $k$  and Hit@ $k$ . Another reason is that some LLMs do not follow the task instruction to re-rank all items and return only a partial ranking. For gpt-4o-mini in the experiments, the probability of this behavior was usually below 2% when  $k = 3$ , but exceeded 3% in most cases when  $k = 5$  (see Section 4.3 in this paper and Section C in the supplementary material for details).

**Table 3: Non-conversational prompts.**

<b>Baseline</b> [9, 31]: We use this minimum component structure as the baseline: $t_{u1} = \text{user\_info} + \text{candidates\_info} + \text{task\_inst}$ .
<b>Emotion</b> [27]: Add an emotionally expressive text to encourage more careful processing: $t_{u1} = \text{Baseline} + \text{"I want to say this before anything else. These results will have a significant impact on my career. Therefore, please think this seriously."}$
<b>Re-Reading</b> [58]: Encourage the LLM to read the prompt twice: $t_{u1} = \text{Baseline} + \text{"Read the question again:"} + \text{Baseline}$ .
<b>Both-Inst</b> [6]: Split the instructions: $t_{sys} = \text{default\_sys\_inst} + \text{user\_info}$ , and $t_{u1} = \text{candidates\_info} + \text{task\_inst}$ .
<b>Recency-Focused</b> [20]: Highlight the most recent interaction: $t_{u1} = \text{user\_info} + \text{"Note that my most recently [item] is [latest item from user\_item\_history]."} + \text{candidates\_info} + \text{task\_inst}$ .
<b>RolePlay-User</b> [5, 24, 46, 51]: Ask the LLM to act as the target user: $t_{sys} = \text{"You are an AI assistant that pretends to be a person who has interacted with the following items. # Logs:{user\_item\_history}"}$ , $t_{u1} = \text{candidates\_info} + \text{task\_inst}$ . In <code>candidates_info</code> , the first sentence of <code>candidates_info</code> is replaced with: <code>"Please rank the candidate [item]s that align closely with your preferences."</code>
<b>RolePlay-Expert</b> [5, 24, 46, 51]: Replace the system message with: $t_{sys} = \text{"You are an AI expert in [dataset] recommendation."}$ ; $t_{u1} = \text{Baseline}$ . In this paper, we set <code>{dataset}</code> to “business” for Yelp, “news” for MIND, “recipe” for Food, and the dataset name itself for others (e.g., the Music category of the Amazon dataset is “music”).
<b>RolePlay-Frederick</b> [6]: Add a name to the system role: $t_{sys} = \text{"You are Frederick."} + \text{default\_sys\_inst}$ ; $t_{u1} = \text{Baseline}$ .
<b>Summarize-Item</b> [1, 64]: The format is the same as <b>Baseline</b> , except that item descriptions in <code>{user_item_history}</code> and <code>{candidate_items}</code> are summarized in advance using the following prompt: <code>"We have the following item details in the [dataset] domain: # Item Information: [item_info] Please summarize this item information, ensuring to include key features and attributes to enhance clarity and understanding within 10–200 words."</code> This helps reduce prompt length when item descriptions are long or noisy.

**Table 4: middle\_inst used in conversational prompts.**

<b>Explain</b> [33, 53]: Explain the reason of the item order. $\text{middle\_inst} = \text{"Please also include the reason for arranging them in that order."}$
<b>Mock</b> [6]: Confirm understanding using $\text{middle\_inst} = \text{"Do you understand?"}$ and set the next reply as $t_{a1} = \text{"Yes! I understand. I am ready to answer your question."}$

### 4.1 RQ1: Model-Wise Effectiveness

We investigate whether each prompt leads to higher recommendation accuracy compared to the **Baseline**, analyzing the results separately for each LLM. This analysis helps identify which prompts are effective for specific LLMs.

**4.1.1 Evaluation Method.** For each LLM, we conducted a hypothesis test to determine whether a prompt performed better than the **Baseline** using a one-sided Wilcoxon signed-rank test<sup>6</sup>. We

<sup>6</sup>The Wilcoxon test is a non-parametric statistical method for comparing paired values. For each user  $u$ , we calculate the difference  $e_u = s_u^p - s_u^{\text{Baseline}}$ , where  $s_u^p$  is the score for prompt  $p$ . Users with  $e_u = 0$  are excluded. We rank the absolute values  $|e_u|$  and compute the test statistic from the sum of ranks where  $e_u > 0$ . The one-sided  $p$ -value is calculated using a normal approximation.

**Table 5: post\_inst used in conversational prompts.**

<b>Step-Back</b> [63]: Consider background knowledge. $\text{post\_inst} = \text{"please consider the principles and theories behind this question."}$
<b>ReAct</b> [60]: Follow explicitly specified steps for reasoning. $\text{post\_inst} = \text{"please follow this format to proceed step by step with *Observation*, *Thought*, and *Action*: Observation: Observe the user's history and preferences. Thought: Infer the user's tastes or tendencies from the observation. Action: Choose one candidate item and examine its characteristics. Observation: Observe the characteristics of that item. Thought: Consider whether the item matches the user's preferences. (Repeat for multiple items if necessary) Finally, provide your *Answer*."}$
<b>Rephrase</b> [12]: Rewrite the input for better understanding. $\text{post\_inst} = \text{"rephrase and expand it to help you do better answering. Maintain all information in the original question."}$
<b>Echo</b> [37]: Repeat the input before answering. $\text{post\_inst} = \text{"let's repeat the question"}$ . We then fix $t_{a2}$ as $t_{a1}$ .
<b>Summarize-User</b> [48, 64, 65]: Describe the user based on their behavior. $\text{post\_inst} = \text{"it is important to express who the target user is. Please describe the user in 100–500 words in a way that contributes to the final objective."}$
<b>Generate-Item</b> [14, 21, 26]: Describe items the user might like. $\text{post\_inst} = \text{"please describe the [item] features that the user might like."}$
<b>Reuse-Item</b> (Original): Use the same instruction as <b>Generate-Item</b> , but remove the most recent item from $\{\text{user\_item\_history}\}$ . We fix $t_{a1}$ as the text of the most recent item.

**Table 6: Fixed initial responses ( $t_{a1}$ ).**

<b>Step-by-Step</b> [23]: Begin with a short instruction for step-by-step thinking. $t_{a1} = \text{"Let's think step by step."}$
<b>Deep-Breath</b> [59]: Insert a calming phrase to promote careful reasoning. $t_{a1} = \text{"take a deep breath and work on this problem step by step."}$
<b>Plan-Solve</b> [50]: Structure the response in three stages: understanding, planning, and solving. $t_{a1} = \text{"Let's first understand the problem and devise a plan to solve the problem. Then, let's carry out the plan and solve the problem step by step."}$

also calculated the relative improvement of each prompt using  $r^p = (m^p - m^{\text{Baseline}})/m^{\text{Baseline}}$ , where  $m^p$  is the average score for prompt  $p$ . These evaluations used all 1,600 users, as well as specific subsets, such as individual datasets (e.g., 200 users from the Movies dataset) or specific user types (e.g., heavy users,  $n = 800$ ).

Table 7 shows the average nDCG@3 scores for each prompt and dataset using gpt-4.1-mini. The last five columns report the overall scores for all users evaluated with the other low-cost LLMs. Relative improvements are summarized in Table 8<sup>7</sup>.

**4.1.2 Discussion.** We summarize key findings from the statistical analysis across five cost-efficient LLMs.

Stable prompts across datasets and LLMs. Some prompts that significantly improved performance over the **Baseline** in gpt-4.1-mini (marked with \* in Table 7) also maintained their effectiveness across different datasets and user types. Importantly, none of them showed a clear decrease in performance. Similar trends were observed with other LLMs such as llama3.3-70b and gpt-40-mini. In particular,

<sup>7</sup>For results with other LLMs, datasets, and Hit@3, please refer to Section B in the supplementary material.

the prompts **Rephrase**, **Step-Back**, **Explain**, **Recency-Focused**, and **Summarize-User** showed higher accuracy from Table 8.

Smaller effects in low-performing LLMs. In results from phi-4 and amazon-nova-lite, no prompt showed significant improvement over the **Baseline**. These models also showed lower accuracy compared to other LLMs. One possible reason is that they may have limited ability to understand detailed prompt instructions. As a result, changes in prompt design had only a small effect.

Performance variation depending on LLMs. Some prompts showed different results depending on the LLM. For example, **Re-Reading** and **Echo** improved accuracy in llama3.3-70b, but reduced it in gpt-4.1-mini and gpt-40-mini. These prompts involve repeated content, which may confuse some models or make it harder to focus on relevant information. Interestingly, **ReAct** showed the highest accuracy in both gpt-4.1-mini and llama3.3-70b, but performed the worst in gpt-40-mini.

Limited effects of NLP-style prompts. Prompts commonly used in NLP, such as **RolePlay**, **Mock**, **Plan-Solve**, **Deep-Breath**, **Emotion**, and **Step-by-Step**, did not improve recommendation accuracy. This suggests that such techniques may be less suitable for ranking tasks, where capturing user preferences is essential.

## 4.2 RQ2: Model-Agnostic Effectiveness

We use a linear mixed-effects model (LMM)<sup>8</sup> to examine how each prompt is effective while controlling for factors such as the LLM, user, and evaluation metric. This method is similar to linear regression but also captures variation across users and LLMs. The model helps identify prompts that perform well under various conditions.

While the previous analysis in Section 4.1 showed that performance varied depending on the LLM, this analysis aims to find prompts that are effective regardless of the LLM or dataset. LMM is suitable for this purpose because it adjusts for differences across users, LLMs, and metrics when evaluating each prompt.

**4.2.1 Evaluation Method.** To evaluate prompt performance across different LLMs and users, we used the following LMM:

$$\text{acc}_i = \beta_0 + \sum_{j=1}^P \beta_j \cdot I(\text{prompt}_i = j) + \gamma_{\text{user}(i)} + \gamma_{\text{llm}(i)} + \gamma_{\text{metric}(i)} + \epsilon_i,$$

where  $\text{acc}_i$  is the accuracy for observation  $i$ ,  $I(\cdot)$  is an indicator function,  $\beta_j$  is the fixed effect of prompt  $j$ , and  $\gamma$  are random effects. We tested each  $\beta_j$  using a Wald test. If  $\beta_j > 0$  and  $p < 0.05$ , we regard the prompt as significantly better than the **Baseline**, which serves as the reference.

**4.2.2 Discussion.** According to Table 9, **ReAct**, **Rephrase**, and **Step-Back** performed significantly better across all datasets and LLMs. These prompts also achieved higher accuracy in gpt-4.1-mini and llama3.3-70b (see Table 7). Since these two LLMs showed relatively strong performance among the low-cost models, their results may have exerted a stronger influence on the LMM estimates.

Prompts such as **Re-Reading**, **RolePlay-User**, **Summarize-Item**, and **Echo** showed lower accuracy in the LMM analysis. As shown in Table 7, **RolePlay-User** and **Summarize-Item** did not

<sup>8</sup>[https://www.statsmodels.org/stable/mixed\\_linear.html](https://www.statsmodels.org/stable/mixed_linear.html)

**Table 7: nDCG@3 for gpt-4.1-mini and other LLMs. The best prompt in each column is underlined. Prompts significantly better or worse than Baseline ( $p < 0.05$ ) are marked with \* or  $\nabla$ , respectively.**

	Music	Movie	Groceries	Clothes	Book	Yelp	News	Food	Light	Heavy	4.1-mini	llama3.3	4o-mini	phi-4	nova-lite
Baseline	0.610	0.516	0.443	0.408	0.585	0.512	0.262	0.328	0.460	0.456	0.458	0.439	0.432	0.417	0.389
Emotion	0.608	<u>0.491</u> $\nabla$	0.438	0.411	0.582	0.503	0.271	0.327	0.461	0.447	0.454	0.442	0.440	0.417	<u>0.376</u> $\nabla$
Re-Reading	<u>0.567</u> $\nabla$	0.487	0.420	0.400	<u>0.553</u> $\nabla$	<u>0.421</u> $\nabla$	0.275	0.304	0.444	<u>0.412</u> $\nabla$	<u>0.428</u> $\nabla$	<u>0.472</u> $*$	<u>0.402</u> $\nabla$	0.409	0.393
Both-Inst	0.609	0.513	0.443	0.398	0.566	<u>0.544</u> $*$	0.226 $\nabla$	0.316	0.459	0.444	0.452	0.437	0.430	0.403	0.396
Recency-Focused	0.601	0.519	<u>0.466</u> $\nabla$	0.420	0.591	0.526	0.245	<u>0.361</u> $\nabla$	0.478	0.455	0.466	<u>0.470</u> $*$	0.443	0.409	0.394
RolePlay-User	<u>0.540</u> $\nabla$	0.484	<u>0.372</u> $\nabla$	0.325 $\nabla$	<u>0.533</u> $\nabla$	0.500	0.195 $\nabla$	0.317	0.420 $\nabla$	0.397 $\nabla$	<u>0.408</u> $\nabla$	0.447	0.411 $\nabla$	<u>0.375</u> $\nabla$	0.391
RolePlay-Expert	0.592	0.507	0.441	0.380 $\nabla$	0.587	0.515	0.260	0.305	0.450	0.446	0.448 $\nabla$	0.440	0.431	0.415	0.383
RolePlay-Frederick	0.588 $\nabla$	0.505	0.441	0.403	0.569	0.511	0.261	0.303	0.451	0.445 $\nabla$	0.448 $\nabla$	0.437	0.426	0.418	0.379
Summarize-Item	<u>0.536</u> $\nabla$	0.495	<u>0.401</u> $\nabla$	0.421	<u>0.536</u> $\nabla$	0.481	0.234	0.318	0.431 $\nabla$	0.424 $\nabla$	0.428 $\nabla$	0.446	0.427	<u>0.371</u> $\nabla$	0.378
Step-Back	0.642	<u>0.561</u> $*$	0.469	0.437	0.594	0.521	0.246	0.340	0.480 $*$	0.473	0.476 $*$	<u>0.481</u> $*$	<u>0.460</u> $*$	<u>0.436</u> $\nabla$	0.348 $\nabla$
ReAct	0.632	<u>0.566</u> $*$	<u>0.511</u> $*$	<u>0.448</u> $\nabla$	0.608	<u>0.548</u> $*$	0.248	0.334	0.479 $*$	<u>0.494</u> $*$	<u>0.487</u> $*$	<u>0.495</u> $*$	<u>0.369</u> $\nabla$	0.413	0.369
Rephrase	0.626	<u>0.564</u> $*$	0.461	0.437	<u>0.627</u> $*$	0.517	0.253	0.359	<u>0.485</u> $*$	0.476 $*$	0.481 $*$	0.467 $*$	0.445 $*$	0.429	0.398
Echo	0.594	0.525	0.426	0.383	0.546 $\nabla$	<u>0.449</u> $\nabla$	0.270	0.282	0.433 $\nabla$	0.436 $\nabla$	<u>0.434</u> $\nabla$	<u>0.477</u> $*$	<u>0.397</u> $\nabla$	0.420	<u>0.349</u> $\nabla$
Summarize-User	<u>0.646</u> $*$	0.523	0.472	0.406	0.580	0.506	0.266	0.324	0.480 $*$	0.452	0.466	0.467 $*$	0.453 $*$	0.417	0.372 $\nabla$
Generate-Item	0.612	0.515	0.457	0.419	0.581	0.532	0.268	0.306 $\nabla$	0.467	0.455	0.461	0.438	0.445 $*$	0.406	0.383
Reuse-Item	0.606	<u>0.551</u> $*$	0.415	0.391	0.578	0.527	<u>0.229</u> $\nabla$	0.329	0.456	0.451	0.453	<u>0.467</u> $*$	0.427	<u>0.382</u> $\nabla$	0.370 $\nabla$
Explain	0.616	0.514	<u>0.476</u> $*$	0.412	0.571	0.530	0.261	0.326	0.466	0.461	0.463	<u>0.465</u> $*$	<u>0.460</u> $*$	0.415	0.379
Mock	0.614	<u>0.495</u> $\nabla$	0.439	0.415	0.580	0.521	<u>0.235</u> $\nabla$	0.320	0.455	0.450	0.452	0.451 $*$	<u>0.408</u> $\nabla$	<u>0.395</u> $\nabla$	0.373 $\nabla$
Step-by-Step	0.608	0.511	<u>0.424</u> $\nabla$	0.428	0.580	0.509	0.277	0.326	0.464	0.452	0.458	0.439	0.428	0.408	0.382
Deep-Breath	0.615	0.517	0.436	<u>0.428</u> $*$	0.591	0.525	<u>0.286</u> $*$	0.313	0.463	0.465	<u>0.464</u> $\nabla$	0.440	0.426	0.407	<u>0.373</u> $\nabla$
Plan-Solve	0.602	0.501	0.443	0.427	0.573	0.526	0.259	0.318	0.462	0.450	0.456	0.442	0.422 $\nabla$	0.402	<u>0.358</u> $\nabla$

**Table 8: Average relative improvements in nDCG@3 and Hit@3. Prompts with significant changes are marked with \* (better) or  $\nabla$  (worse), as in Table 7. “Avg” shows the mean across all LLMs, with parentheses showing the total count of significant differences.**

	4.1-mini	llama3.3	4o-mini	phi-4	nova-lite	Avg
Rephrase	4.5*	5.9*	2.4	3.0	1.7	3.5(5)
Step-Back	4.4*	8.9*	5.6*	3.6	<u>-11.1</u> $\nabla$	2.3(4)
Explain	1.0	6.1*	5.3*	-0.2	-2.3	2.0(4)
Summarize-User	1.2	5.9*	4.3*	0.5	<u>-5.2</u> $\nabla$	1.3(2)
Recency-Focused	0.9	6.0*	1.4	-1.2	1.4	1.7(2)
ReAct	5.8*	12.0*	<u>-13.8</u> $\nabla$	-0.2	-5.3	-0.3(1)
Generate-Item	0.4	-0.2	2.3	-2.3	-0.9	-0.2(1)
Both-Inst	-1.3	-0.6	-1.4	-3.3	1.7	-1.0
Step-by-Step	-0.3	-0.1	-1.7	-1.6	-1.9	-1.1
RolePlay-Expert	-1.6	0.4	-0.9	-0.6	-1.0	-0.8(-1)
Emotion	-1.0	1.1	1.5	0.5	<u>-3.5</u> $\nabla$	-0.3(-2)
Re-Reading	-6.1 $\nabla$	7.6*	<u>-7.6</u> $\nabla$	-1.4	1.9	-1.1(-2)
Reuse-Item	-1.3	5.6*	-1.7	<u>-7.7</u> $\nabla$	-5.0 $\nabla$	-2.0(-2)
Deep-Breath	1.3	0.1	-1.8	-2.2	-4.0 $\nabla$	-1.3(-2)
RolePlay-Frederick	-2.0	-0.4	-1.9	0.5	-2.2	-1.2(-2)
Plan-Solve	-0.6	0.9	-2.3	-3.0	<u>-8.3</u> $\nabla$	-2.7(-3)
Summarize-Item	-5.8 $\nabla$	1.1	-1.8	<u>-9.6</u> $\nabla$	-2.7	-3.8(-4)
Echo	-4.9 $\nabla$	8.5*	<u>-8.1</u> $\nabla$	1.6	<u>-9.0</u> $\nabla$	-2.4(-4)
Mock	-1.1	2.2	-6.5 $\nabla$	<u>-4.6</u> $\nabla$	-4.1 $\nabla$	-2.8(-5)
RolePlay-User	-10.9 $\nabla$	1.5	-5.9 $\nabla$	<u>-9.7</u> $\nabla$	1.1	-4.8(-6)

improve performance in any LLM, while **Re-Reading** and **Echo** were effective only in llama3.3-70b and showed negative effects in others.

### 4.3 RQ3: Error Analysis

This section examines why some prompts led to lower accuracy. As described in Section 3.4, if an LLM failed to return any ranking and all five retries also failed, we completed it by random ranking. The left values in Table 10 show these failure rates.

**Table 9: Prompts that showed statistically significant positive or negative effects by LMEM compared to the Baseline.**

Dataset	Positive	Negative
Music		Summarize-Item, RolePlay-User, Re-Reading
Movie	ReAct, Rephrase	
Groceries	ReAct	RolePlay-User
Clothes		RolePlay-User
Book	Rephrase	RolePlay-User, Summarize-Item
Yelp		Re-Reading, Echo
News		RolePlay-User
Food		Echo
Light	Rephrase	RolePlay-User, Summarize-Item, Echo
Heavy	ReAct	Re-Reading, RolePlay-User, Summarize-Item
ALL	ReAct, Rephrase, Step-Back	Re-Reading, RolePlay-User, Summarize-Item, Echo

For gpt-4.1-mini and llama3.3-70b, almost all outputs followed the expected format. For gpt-4o-mini, **ReAct** showed a failure rate of 1.4%. In phi-4, all prompts except **Summarize-Item** had failure rates above 10%, mostly due to token length limits<sup>9</sup>. In amazon-nova-lite, both **ReAct** and **Step-Back** exceeded 10%. For LLMs other than phi-4 and amazon-nova-lite, many failures included responses such as “*the user is interested in the candidate items XX, XX, XX.*”, where the LLM ignored the instruction in task\_inst to return a full ranking<sup>9</sup>. In contrast, failures in phi-4 and amazon-nova-lite were often caused by prompts that were too long for the LLMs to process<sup>9</sup>. Since **Summarize-Item** compresses item descriptions and reduces token length, it resulted in the lowest failure rate in phi-4.

We also examined cases where the LLM returned only a few items, although a full ranking was still constructed by randomly

<sup>9</sup>Detailed generated outputs, token lengths per dataset, and rates for outputs with only one or five or fewer items are provided in Section C of the supplementary material.

**Table 10: Percentages of outputs with failures (left) and those containing three or fewer ranked items (right).**

	4.1-mini	llama3.3	4o-mini	phi-4	nova-lite
Baseline	0/0	0/0	0.4/1.7	<b>16.6/12.5</b>	0.1/0
Emotion	0/0	0/0	0/1.0	<b>17.0/12.6</b>	0/0
Re-Reading	0/0	0/0	0.4/2.1	<b>46.1/26.1</b>	0/0
Both-Inst	0/0	0/0	0.4/1.0	<b>16.4/12.5</b>	0.1/0
Recency-Focused	0/0	0/0	0.8/2.1	<b>18.0/12.6</b>	0.1/0
RolePlay-User	0/0	0/0	0/0.1	<b>16.2/12.5</b>	0/0
RolePlay-Expert	0/0	0/0	0.4/1.3	<b>16.6/12.5</b>	0/0
RolePlay-Frederick	0/0	0/0	0.5/1.6	<b>16.2/12.8</b>	0.1/0
Summarize-Item	0/0	0/0	0/0.3	7.9/0	0/0
Step-Back	0/0	0/0	0/1.7	<b>35.5/13.6</b>	<b>39.6/0.5</b>
ReAct	0.1/0	0/0	<b>1.4/21.5</b>	<b>26.4/15.1</b>	<b>19.1/2.9</b>
Rephrase	0/0	0/0	0.1/1.8	<b>25.0/14.1</b>	5.1/0
Echo	0/0	0/0	0.5/2.2	<b>44.9/25.0</b>	0/0
Summarize-User	0/0	0/0	0.1/0.8	<b>18.8/13.0</b>	1.7/0
Generate-Item	0/0	0/0	0.2/1.0	<b>24.3/12.5</b>	0.2/0
Reuse-Item	0/0	0/0	0/2.4	<b>19.5/12.6</b>	0/0
Explain	0/0	0/0	0/0.2	<b>23.1/12.6</b>	0.1/0
Mock	0/0	0/0	0.1/1.5	<b>16.9/12.5</b>	0/0
Step-by-Step	0/0	0/0	0/1.7	<b>36.0/13.5</b>	0.6/0
Deep-Breath	0/0	0/0	0/1.7	<b>39.1/13.3</b>	0.4/0
Plan-Solve	0/0	0/0	0/2.4	<b>50.9/14.1</b>	8.4/1.9

appending the unmentioned ones. Specifically, we calculated the rate at which the output contained three or fewer items. As shown in the right values of Table 10<sup>9</sup>, **ReAct** in gpt-4o-mini showed a partial output rate of over 20%. This was the only prompt-model combination in gpt-4o-mini with such a high rate, and this behavior contributed to the lower accuracy observed in Table 7. These results suggest that failing to follow instructions and returning incomplete rankings is a key factor in reduced accuracy.

#### 4.4 Key Findings of This Section

We summarize the findings from this section. In Section 4.1, we compared recommendation accuracy across different prompts for each LLM and dataset. Prompts such as **Rephrase**, **Step-Back**, **Explain**, **Recency-Focused**, and **Summarize-User** improved accuracy under many conditions. Section 4.2 extended this analysis using a linear mixed-effects model, which adjusted for differences across users, evaluation metrics, and LLMs. The results confirmed that **ReAct**, **Step-Back**, and **Rephrase** performed well across diverse settings, which are similar to the effective prompts found in Section 4.1. In contrast, as shown in Section 4.3, prompts that did not improve accuracy often failed to follow the output format or return complete rankings.

Based on these findings, we conclude that in cost-efficient LLMs such as gpt-4.1-mini and llama3.3-70b, prompts that clarify the task context and goal, such as **ReAct**, **Step-Back**, and **Rephrase**, are especially effective for recommendation.

#### 5 Toward Improving Accuracy

In Section 4, we identified several prompts that improved recommendation accuracy. In this section, we explore whether additional improvement is possible by combining prompts, applying meta-level techniques, or using more advanced LLMs.

#### 5.1 Prompt Combinations

We focus on **ReAct**, **Step-Back**, and **Rephrase**, which were found to be effective in Section 4, and create combinations by using them in sequence. **ReAct** and **Step-Back** support structured reasoning, while **Rephrase** makes the instruction text easier to understand. Based on this, we design two combinations: **Rephrase → ReAct** and **Rephrase → Step-Back**<sup>10</sup>.

We also apply meta-prompts that refine outputs based on the LLM’s initial reasoning (Table 11). Let  $t_{u1}$  be the base prompt (e.g., **ReAct**),  $t_{a1}$  the initial output, and  $t_{u3} = \text{final\_inst}$ . These methods use intermediate reasoning explicitly to improve the final response. We apply **Self-Refine** and **Self-Consistency** to **Rephrase**, **ReAct**, and **Step-Back**, resulting in six meta-prompt variants.

**Table 11: Meta-level prompting**

**Self-Refine** [34]: After generating  $t_{a1}$ , the model is asked to review its own answer using  $t_{u2} = \text{"Thank you! As an expert, what do you think about the above answer? Please provide feedback so that more accurate predictions can be made in the future."}$ .

**Self-Consistency** [52]: The prompt  $t_{u1}$  is executed multiple times to produce  $\{t_{a1,1}, t_{a1,2}, \dots\}$ . These answers are then summarized using  $t_{u2} = \text{"We asked the monitors and got the following answers: # Collected answers: \{t_{a1,1}, t_{a1,2}, \dots\}"}$ .

Since these prompts require additional steps or repeated inference, their inference cost can be several times higher. To keep the comparison practical, we reduce the number of users from 100 to 10 per dataset and user type, resulting in 160 users in total<sup>11</sup>.

**Table 12: nDCG@3 across datasets, where SR and SC refer to Self-Refine and Self-Consistency.**

	4.1-mini	llama3.3	4o-mini	phi-4	nova-lite
ReAct	0.547*	0.533*	0.407	0.407	0.367
SC (ReAct)	0.535*	0.537*	0.389	0.402	0.441
SC (Step-Back)	0.543*	0.518*	0.454	0.427	0.369
Step-Back	0.516	0.543*	0.444	0.439	0.359
SC (Rephrase)	0.522*	0.536*	0.468	0.460	0.397
SR (ReAct)	0.540*	0.516*	0.378	0.369 <sup>v</sup>	0.407
SR (Rephrase)	0.514	0.533*	0.462	0.471	0.455
SR (Step-Back)	0.520	0.527*	0.458	0.392 <sup>v</sup>	0.390
Rephrase→ReAct	0.534*	0.506	0.376	0.472	0.402
Rephrase→Step-Back	0.514	0.501	0.470	0.481	0.352
Baseline	0.475	0.465	0.440	0.460	0.412

According to Table 12, few prompt combinations or meta-level methods outperformed the original prompts. In gpt-4.1-mini, applying **Self-Consistency** to **Step-Back** slightly improved accuracy, but similar effects were not observed in other cases. In amazon-nova-lite, applying **Self-Refine** to **Rephrase** led to a large improvement in accuracy. However, the variance of the outputs was high, and the  $p$ -value was not significant. Furthermore, the resulting accuracy did not exceed that of other LLMs. These results suggest that in cost-efficient LLMs such as gpt-4.1-mini

<sup>10</sup>Full prompt wordings are provided in Section A of the supplementary material.

<sup>11</sup>This setting balances computational cost and statistical reliability. Clear differences were already observed with this sample size in Table 12.

and llama3.3-70b, **Rephrase**, **ReAct**, and **Step-Back** are already effective without additional techniques. Since additional instructions increased inference cost but did not meaningfully improve accuracy, this conclusion seems reasonable.

In summary, using the original forms of **Rephrase**, **ReAct**, and **Step-Back** remains the most practical choice, as they are simple, effective, and cost-efficient.

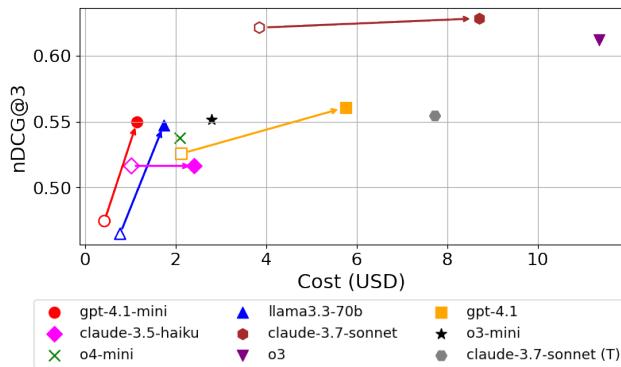
## 5.2 Evaluation Using High-Performance LLMs

In the previous experiments, we used cost-efficient LLMs such as gpt-4.1-mini to compare all prompts. As another way to improve recommendation accuracy, we consider using more powerful LLMs. In this section, we examine whether the prompts **Rephrase**, **ReAct**, and **Step-Back**, which showed strong performance, are still effective with stronger models. For reasoning models, we evaluated only the **Baseline**, since their internal thinking processes may conflict with custom instructions, and we observed that o3-mini actually did.

Table 13 shows the recommendation accuracy and inference cost for each model and prompt under the same settings as in Section 12. Figure 1 presents a two-dimensional plot of these results.

**Table 13: nDCG@3 and inference cost (USD) for each LLM. claude-3.7-sonnet (T) refers to the version with thinking mode enabled and is treated as a reasoning model.**

	Baseline	Rephrase	Step-Back	ReAct
llama3.3-70b	0.465 / 0.78	0.548* / 1.74	0.543* / 1.69	0.533* / 1.78
gpt-4.1-mini	0.475 / 0.43	0.550* / 1.15	0.516 / 1.08	0.547* / 1.18
claude-3.5-haiku	0.516 / 1.03	0.501 / 2.28	0.506 / 2.30	0.516 / 2.41
gpt-4.1	0.526 / 2.13	0.515 / 4.89	0.527 / 4.87	0.561 / 5.76
claude-3.7-sonnet	0.621 / 3.85	0.621 / 9.23	0.628 / 8.71	0.609 / 10.69
o3-mini	0.552 / 2.79	0.537 / 5.00	0.542 / 4.82	(failed)
o4-mini	0.537 / 2.09	-	-	-
o3	0.612 / 11.36	-	-	-
claude-3.7-sonnet (T)	0.555 / 7.72	-	-	-



**Figure 3: nDCG@3 and inference cost. For non-reasoning models, hollow and filled markers represent the Baseline and the highest-scoring prompt in Table 13, respectively.**

Table 13 shows that the highest accuracy was achieved by the combination of claude-3.7-sonnet and **Step-Back**, which resulted in an nDCG@3 score of 0.628. However, other prompts using

the same LLM, including **Baseline**, also had scores around 0.62, suggesting no clear advantage. The next highest score was 0.612 for o3 with **Baseline**. Several other combinations had scores near 0.55, such as gpt-4.1 with **ReAct**, claude-3.7-sonnet (T) with **Baseline**, and o3-mini with **Baseline**.

Accuracy decreasing from overthinking. Using the thinking mode (T) in claude-3.7-sonnet did not improve accuracy and even resulted in lower performance than the **Baseline**. In o3-mini, **Rephrase** and **Step-Back** also failed to improve performance, and every attempt to use **ReAct** was blocked due to policy violations, even after five retries. A similar issue has been reported in the OpenAI developer community<sup>12</sup>. Such failures may be caused by internal instructions that restrict the use of certain prompts. These observations suggest that when reasoning steps become complicated, the LLM may become confused, which is referred to as the overthinking problem [17, 43]. To avoid these, we recommend checking the behavior of prompts in advance before deploying them in recommendation tasks.

Best Option for Accuracy Among the LLM and prompt combinations with nDCG@3 scores above 0.6, claude-3.7-sonnet with **Step-Back** achieved the highest accuracy. However, the difference compared to **Baseline** was minimal. o3 also reached a comparable level of accuracy, but in terms of cost, using claude-3.7-sonnet with **Baseline** was several times cheaper. Considering the risk that unnecessary modifications may lead to reduce accuracy, we conclude that this simple configuration is the most reasonable choice.

Best option for cost efficiency. For second-tier accuracy around 0.55, using gpt-4.1-mini or llama3.3-70b with one of **Rephrase**, **ReAct**, or **Step-Back** was cost-efficient. For example, gpt-4.1-mini with **Rephrase** achieved accuracy comparable to claude-3.7-sonnet (T) with **Baseline**, but at only one-fifth of the cost.

## 6 Practical Guideline

This section explains a practical guideline to apply the findings of this study to newly developed or reader-targeted LLMs, prompt methods, or datasets. As a starting point, we suggest testing four prompts that showed strong results in our experiments: **Baseline**, **Rephrase**, **Step-Back**, and **ReAct**. At the same time, readers should record both ranking metrics such as nDCG@k and Hit@k, and the inference cost. After collecting the results, it is important to check the inference logs. In particular, readers should look for incomplete rankings or responses that do not follow the instruction, as described in Section 4.3. By comparing the four prompts, readers can select the one that fits their needs, whether they focus on accuracy, cost, or a balance between the two. We hope this guideline supports future work in LLM-based recommender systems.

## 7 Conclusion

This study examined the effect of prompt design on personalized recommendation using LLMs. We compared 23 prompts across 8 datasets and 12 LLMs in a systematic evaluation. We also tested combinations of prompts and meta-level techniques.

<sup>12</sup><https://community.openai.com/t/your-prompt-was-flagged-as-potentially-violating-our-usage-policy/1060163/27>

We found that prompts that restructure user context or guide the model through structured reasoning, such as **Rephrase**, **Step-Back**, and **ReAct**, were effective with cost-efficient LLMs like gpt-4.1-mini and llama3.3-70b. In contrast, general-purpose prompting techniques often used in NLP, such as **RolePlay**, **Emotion**, and **Step-by-Step**, did not always improve recommendation accuracy and reduced performance in some cases. For high-performance models like claudie-3.7-sonnet, the simple **Baseline** prompt achieved high accuracy without increasing inference cost. These results provide practical guidelines for designing prompts and selecting LLMs for real-world recommendation tasks.

## References

- [1] Arkadeep Acharya, Brijraj Singh, and Naoyuki Onoe. 2023. LLM Based Generation of Item-Description for Recommendation System. In *RecSys*. ACM, 1204–1207.
- [2] 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 *RecSys*. ACM, 1007–1014.
- [3] Oren Barkan and Noam Koenigstein. 2016. Item2vec: Neural Item Embedding for Collaborative Filtering. In *RecSys Posters (CEUR Workshop Proceedings, Vol. 1688)*. CEUR-WS.org.
- [4] Filippo Betello, Antonio Purificato, Federico Siciliano, Giovanni Trappolini, Andrei Bacciu, Nicola Tonello, and Fabrizio Silvestri. 2025. A Reproducible Analysis of Sequential Recommender Systems. *IEEE Access* 13 (2025), 5762–5772.
- [5] Jiangjie Chen, Xintao Wang, Rui Xu, Siyu Yuan, Yikai Zhang, Wei Shi, Jian Xie, Shuang Li, Ruihan Yang, Tinghui Zhu, Aili Chen, Nianqi Li, Lida Chen, Caiyu Hu, Siye Wu, Scott Ren, Ziqian Fu, and Yanghua Xiao. 2024. From Persona to Personalization: A Survey on Role-Playing Language Agents. *TMLR* (2024), 2835–8856.
- [6] Benjamin Clavié, Alexandru Ciceu, Frederick Naylor, Guillaume Soulié, and Thomas Brightwell. 2023. Large Language Models in the Workplace: A Case Study on Prompt Engineering for Job Type Classification. In *NLDB (Lecture Notes in Computer Science, Vol. 13913)*. Springer, 3–17.
- [7] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep Neural Networks for YouTube Recommendations. In *RecSys*. ACM, 191–198.
- [8] Maurizio Ferrari Dacrema, Paolo Cremonesi, and Dietmar Jannach. 2019. Are we really making much progress? A worrying analysis of recent neural recommendation approaches. In *RecSys*. ACM, 101–109.
- [9] 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 *RecSys*. ACM, 1126–1132.
- [10] Alexander Dallmann, Daniel Zoller, and Andreas Hotho. 2021. A Case Study on Sampling Strategies for Evaluating Neural Sequential Item Recommendation Models. In *RecSys*. ACM, 505–514.
- [11] Yashar Deldjoo, Zhankui He, Julian J. McAuley, Anton Korikov, Scott Sanner, Arnaud Ramisa, René Vidal, Maheswaran Sathiamoorthy, Atoosa Kasirzadeh, and Silvia Milano. 2024. A Review of Modern Recommender Systems Using Generative Models (Gen-RecSys). In *KDD*. ACM, 6448–6458.
- [12] Yih Deng, Weitong Zhang, Zixiang Chen, and Quanquan Gu. 2023. Rephrase and Respond: Let Large Language Models Ask Better Questions for Themselves. *CoRR* abs/2311.04205 (2023).
- [13] Wenqi Fan, Zihua Zhao, Jiatong Li, Yunqing Liu, Xiaowei Mei, Yiqi Wang, Jiliang Tang, and Qing Li. 2023. Recommender Systems in the Era of Large Language Models (LLMs). *TKDE* 01 (5555), 1–20.
- [14] Yi Fang, Wenjie Wang, Yang Zhang, Fengbin Zhu, Qifan Wang, Fuli Feng, and Xiangnan He. 2025. Reason4Rec: Large Language Models for Recommendation with Deliberative User Preference Alignment. *CoRR* abs/2502.02061 (2025).
- [15] Yunfan Gao, Tao Sheng, Youlin Xiang, Yun Xiong, Haofen Wang, and Jiawei Zhang. 2023. Chat-REC: Towards Interactive and Explainable LLMs-Augmented Recommender System. *CoRR* abs/2303.14524 (2023).
- [16] Mihajlo Grbovic, Vladan Radosavljevic, Nemanja Djuric, Narayan Bhamidipati, Jaikit Savla, Varun Bhagwan, and Doug Sharp. 2015. E-commerce in Your Inbox: Product Recommendations at Scale. In *KDD*. ACM, 1809–1818.
- [17] Danny Halawi, Jean-Stanislas Denain, and Jacob Steinhardt. 2024. Overthinking the Truth: Understanding how Language Models Process False Demonstrations. In *ICLR*. OpenReview.net.
- [18] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural Collaborative Filtering. In *WWW*. ACM, 173–182.
- [19] Zhankui He, Zhouhang Xie, Rahul Jha, Harald Steck, Dawen Liang, Yesu Feng, Bodhisattwa Prasad Majumder, Nathan Kallus, and Julian J. McAuley. 2023. Large Language Models as Zero-Shot Conversational Recommenders. In *CIKM*. ACM, 720–730.
- [20] 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 *ECIR* (2) (*Lecture Notes in Computer Science, Vol. 14609*). Springer, 364–381.
- [21] Jianchao Ji, Zelong Li, Shuyuan Xu, Wenyue Hua, Yingqiang Ge, Juntao Tan, and Yongfeng Zhang. 2024. GenRec: Large Language Model for Generative Recommendation. In *ECIR* (3) (*Lecture Notes in Computer Science, Vol. 14610*). Springer, 494–502.
- [22] Muhammad Murad Khan, Roliana Ibrahim, and Imran Ghani. 2017. Cross Domain Recommender Systems: A Systematic Literature Review. *ACM Comput. Surv.* 50, 3 (2017), 36:1–36:34.
- [23] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large Language Models are Zero-Shot Reasoners. In *NeurIPS*.
- [24] Aobo Kong, Shiwuan Zhao, Hao Chen, Qicheng Li, Yong Qin, Ruiqi Sun, Xin Zhou, Enzhi Wang, and Xiaohang Dong. 2024. Better Zero-Shot Reasoning with Role-Play Prompting. In *NAACL-HLT*. ACL, 4099–4113.

- [25] Yehuda Koren, Robert M. Bell, and Chris Volinsky. 2009. Matrix Factorization Techniques for Recommender Systems. *Computer* 42, 8 (2009), 30–37.
- [26] Genki Kusano. 2024. Data Augmentation using Reverse Prompt for Cost-Efficient Cold-Start Recommendation. In *RecSys*. ACM, 861–865.
- [27] Cheng Li, Jindong Wang, Yixuan Zhang, Kaijie Zhu, Wenxin Hou, Jianxun Lian, Fang Luo, Qiang Yang, and Xing Xie. 2023. Large Language Models Understand and Can Be Enhanced by Emotional Stimuli. *CoRR* abs/2307.11760 (2023).
- [28] Lei Li, Yongfeng Zhang, and Li Chen. 2023. Prompt Distillation for Efficient LLM-based Recommendation. In *CIKM*. ACM, 1348–1357.
- [29] Lei Li, Yongfeng Zhang, Dugang Liu, and Li Chen. 2024. Large Language Models for Generative Recommendation: A Survey and Visionary Discussions. In *LREC/COLING*. ELRA and ICCL, 10146–10159.
- [30] Jianghao Lin, Xinyi Dai, Yunji Xi, Weiven Liu, Bo Chen, Xiangyang Li, Chenxu Zhu, Hufeng Guo, Yong Yu, Ruiming Tang, and Weinan Zhang. 2024. How Can Recommender Systems Benefit from Large Language Models: A Survey. *ACM Trans. Inf. Syst.* (2024). Just Accepted.
- [31] Junling Liu, Chao Liu, Renjie Lv, Kang Zhou, and Yan Zhang. 2023. Is ChatGPT a Good Recommender? A Preliminary Study. *CoRR* abs/2304.10149 (2023).
- [32] Jiahao Liu, Xueshuo Yan, Dongsheng Li, Guangping Zhang, Hansu Gu, Peng Zhang, Tun Lu, Li Shang, and Ning Gu. 2025. Improving LLM-powered Recommendations with Personalized Information. *CoRR* abs/2502.13845 (2025).
- [33] Yucong Luo, Mingyue Cheng, Hao Zhang, Junyu Lu, and Enhong Chen. 2024. Unlocking the Potential of Large Language Models for Explainable Recommendations. In *DASFAA* (5) (*Lecture Notes in Computer Science*, Vol. 14854). Springer, 286–303.
- [34] Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegrefe, Uri Alon, Nouha Dziri, Shrimai Prabhuramoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. 2023. Self-Refine: Iterative Refinement with Self-Feedback. In *NeurIPS*.
- [35] Bodhisattwa Prasad Majumder, Shuyang Li, Jianmo Ni, and Julian J. McAuley. 2019. Generating Personalized Recipes from Historical User Preferences. In *EMNLP/IJCNLP* (1). ACL, 5975–5981.
- [36] Tong Man, Huawei Shen, Xiaolong Jin, and Xueqi Cheng. 2017. Cross-Domain Recommendation: An Embedding and Mapping Approach. In *IJCAI*. ijcai.org, 2464–2470.
- [37] Raja Sekhar Reddy Mekala, Yasaman Razeghi, and Sameer Singh. 2024. EchoPrompt: Instructing the Model to Rephrase Queries for Improved In-context Learning. In *NAACL (Short Papers)*. ACL, 399–432.
- [38] Aleksandr Milogradskii, Oleg Lashinini, Alexander P, Marina Ananyeva, and Sergey Kolesnikov. 2024. Revisiting BPR: A Replicability Study of a Common Recommender System Baseline. In *RecSys*. ACM, 267–277.
- [39] Jianmo Ni, Jiacheng Li, and Julian J. McAuley. 2019. Justifying Recommendations using Distantly-Labeled Reviews and Fine-Grained Aspects. In *EMNLP/IJCNLP* (1). ACL, 188–197.
- [40] Aleksandr V. Petrov and Craig Macdonald. 2022. A Systematic Review and Replicability Study of BERT4Rec for Sequential Recommendation. In *RecSys*. ACM, 436–447.
- [41] Steffen Rendle, Walid Krichene, Li Zhang, and John R. Anderson. 2020. Neural Collaborative Filtering vs. Matrix Factorization Revisited. In *RecSys*. ACM, 240–248.
- [42] Pranab Sahoo, Ayush Kumar Singh, Sriparna Saha, Vinija Jain, Samrat Mondal, and Aman Chatha. 2024. A Systematic Survey of Prompt Engineering in Large Language Models: Techniques and Applications. *CoRR* abs/2402.07927 (2024).
- [43] Yang Sui, Yu-Neng Chuang, Guanchu Wang, Jiamu Zhang, Tianyi Zhang, Jiayi Yuan, Hongyi Liu, Andrew Wen, Shaochen Zhong, Hanjie Chen, and Xia Ben Hu. 2025. Stop Overthinking: A Survey on Efficient Reasoning for Large Language Models. *CoRR* abs/2503.16419 (2025).
- [44] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019. BERT4Rec: Sequential Recommendation with Bidirectional Encoder Representations from Transformer. In *CIKM*. ACM, 1441–1450.
- [45] Zhu Sun, Di Yu, Hui Fang, Jie Yang, Xinghua Qu, Jie Zhang, and Cong Geng. 2020. Are We Evaluating Rigorously? Benchmarking Recommendation for Reproducible Evaluation and Fair Comparison. In *RecSys*. ACM, 23–32.
- [46] Yu-Min Tseng, Yu-Chao Huang, Teng-Yun Hsiao, Wei-Lin Chen, Chao-Wei Huang, Yu Meng, and Yun-Nung Chen. 2024. Two Tales of Persona in LLMs: A Survey of Role-Playing and Personalization. In *EMNLP (Findings)*. ACL, 16612–16631.
- [47] Shubham Vatsal and Harsh Dubey. 2024. A Survey of Prompt Engineering Methods in Large Language Models for Different NLP Tasks. *CoRR* abs/2407.12994 (2024).
- [48] Lei Wang and Ee-Peng Lim. 2023. Zero-Shot Next-Item Recommendation using Large Pretrained Language Models. *CoRR* abs/2304.03153 (2023).
- [49] Lei Wang and Ee-Peng Lim. 2024. The Whole is Better than the Sum: Using Aggregated Demonstrations in In-Context Learning for Sequential Recommendation. In *NAACL-HLT*. ACL, 876–895.
- [50] Lei Wang, Wanyu Xu, Yihuai Lan, Zhiqiang Hu, Yunshi Lan, Roy Ka-Wei Lee, and Ee-Peng Lim. 2023. Plan-and-Solve Prompting: Improving Zero-Shot Chain-of-Thought Reasoning by Large Language Models. In *ACL* (1). ACL, 2609–2634.
- [51] Noah Wang, Zhongyuan Peng, Haoran Que, Jiaheng Liu, Wangchunshu Zhou, Yuhua Wu, Hongcheng Guo, Ruitong Gan, Zehao Ni, Jian Yang, Man Zhang, Zhaoxiang Zhang, Wanli Ouyang, Ke Xu, Wenhao Huang, Jie Fu, and Junran Peng. 2024. RoleLLM: Benchmarking, Eliciting, and Enhancing Role-Playing Abilities of Large Language Models. In *ACL (Findings)*. ACL, 14743–14777.
- [52] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksa Chowdhery, and Denny Zhou. 2023. Self-Consistency Improves Chain of Thought Reasoning in Language Models. In *ICLR*. OpenReview.net.
- [53] 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 *NeurIPS*.
- [54] Yinwei Wei, Xiang Wang, Qi Li, Liqiang Nie, Yan Li, Xuanping Li, and Tat-Seng Chua. 2021. Contrastive Learning for Cold-Start Recommendation. In *ACM Multimedia*. ACM, 5382–5390.
- [55] Fangzhao Wu, Ying Qiao, Jiun-Hung Chen, Chuhan Wu, Tao Qi, Jianxun Lian, Danyang Liu, Xing Xie, Jianfeng Gao, Winnie Wu, and Ming Zhou. 2020. MIND: A Large-scale Dataset for News Recommendation. In *ACL*. ACL, 3597–3606.
- [56] Likang Wu, Zhi Zheng, Zhaopeng Qiu, Hao Wang, Hongchao Gu, Tingjia Shen, Chuan Qin, Chen Zhu, Hengshu Zhu, Qi Liu, Hui Xiong, and Enhong Chen. 2024. A survey on large language models for recommendation. *World Wide Web (WWW)* 27, 5 (2024), 60.
- [57] Lanling Xu, Junjie Zhang, Bingqian Li, Jinpeng Wang, Sheng Chen, Wayne Xin Zhao, and Ji-Rong Wen. 2025. Tapping the Potential of Large Language Models as Recommender Systems: A Comprehensive Framework and Empirical Analysis. *ACM Transactions on Knowledge Discovery from Data* (2025). Just Accepted.
- [58] Xiaohan Xu, Chongyang Tao, Tao Shen, Can Xu, Hongbo Xu, Guodong Long, Jian-Guang Lou, and Shuai Ma. 2024. Re-Reading Improves Reasoning in Large Language Models. In *EMNLP*. ACL, 15549–15575.
- [59] Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V. Le, Denny Zhou, and Xinyun Chen. 2024. Large Language Models as Optimizers. In *ICLR*. OpenReview.net.
- [60] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R. Narasimhan, and Yuan Cao. 2023. ReAct: Synergizing Reasoning and Acting in Language Models. In *ICLR*. OpenReview.net.
- [61] Zhenrui Yue, Sara Rabhi, Gabriel de Souza Pereira Moreira, Dong Wang, and Even Oldridge. 2023. LlamaRec: Two-Stage Recommendation using Large Language Models for Ranking. *CoRR* abs/2311.02089 (2023).
- [62] Tianzi Zang, Yanmin Zhu, Haobing Liu, Ruohan Zhang, and Jiadi Yu. 2023. A Survey on Cross-domain Recommendation: Taxonomies, Methods, and Future Directions. *ACM Trans. Inf. Syst.* 41, 2 (2023), 42:1–42:39.
- [63] Huaixiu Steven Zheng, Swaroop Mishra, Xinyun Chen, Heng-Tze Cheng, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2024. Take a Step Back: Evoking Reasoning via Abstraction in Large Language Models. In *ICLR*. OpenReview.net.
- [64] Zhi Zheng, Wenshuo Chao, Zhaopeng Qiu, Hengshu Zhu, and Hui Xiong. 2024. Harnessing Large Language Models for Text-Rich Sequential Recommendation. In *WWW*. ACM, 3207–3216.
- [65] Joyce Zhou, Yijia Dai, and Thorsten Joachims. 2024. Language-Based User Profiles for Recommendation. *CoRR* abs/2402.15623 (2024).
- [66] Zhihui Zhou, Lilin Zhang, and Ning Yang. 2023. Contrastive Collaborative Filtering for Cold-Start Item Recommendation. In *WWW*. ACM, 928–937.
- [67] Ziwei Zhu, Shahin Sefati, Parsa Saadatpanah, and James Caverlee. 2020. Recommendation for New Users and New Items via Randomized Training and Mixture-of-Experts Transformation. In *SIGIR*. ACM, 1121–1130.

## A Detailed Prompts

We show the prompts used in the main paper together with the intermediate output text from gpt-4o-mini. We replace long texts where the LLM examines many items and its inner reasoning steps with the blue marker “(omitted)”.

- **Baseline:** Figure 4
- **Rephrase:** Figure 5
- **ReAct:** Figure 6
- **Step-Back:** Figure 7
- **Step-by-Step:** Figure 8
- **Pretend-User:** Figure 9
- **Summarize-Item:** Figure 10
- **Rephrase → ReAct:** Figure 11
- **SelfRefine (Step-Back):** Figure 12

## B Accuracies in Section 4.1

Table 7 in the main paper showed the nDCG@3 accuracy for all datasets using gpt-4.1-mini. In this section, we show accuracy tables for the other LLMs.

- 11lama3.3-70b: Table 14
- gpt-4o-mini: Table 15
- phi4: Table 16
- amazon-nova-lite: Table 17

We also present the prompts and LLMs used with each dataset.

- **Music:** Table 18
- **Movie:** Table 19
- **Groceries:** Table 20
- **Clothes:** Table 21
- **Books:** Table 22
- **Yelp:** Table 23
- **News:** Table 24
- **Food:** Table 25
- **Light:** Table 26
- **Heavy:** Table 27

Table 28 and Table 29 show the Hit@3 results corresponding to Table 7 and Table 13 in the main paper.

## C Error Analysis in Section 4.3

Table 10 in the main paper shows the percentage of cases where the LLMs generated three or fewer ranked items, as well as the percentage of failed cases. Table 30 shows the percentage of cases where the LLMs generated five or fewer and nine or fewer items.

Table 31 and Table 32 show the maximum, minimum, average, and standard deviation of token counts for the **Baseline** and **Summarize-Item** prompts across all datasets.

Figure 13 shows the internal reasoning process in a failed case using the **ReAct** prompt with gpt-4o-mini. In this case, the LLM responded with “*Since none of the candidate products align with the user’s preferences*”.

## D Inference Cost in the main paper

Table 33, Table 34 and Table 35 summarize the inference costs in Section 4, 5.1, and 5.2 of the main paper.

```
tsys = You are an AI assistant that helps people find information.
-----
tu1 = # Requirements:
you must rank candidate products that will be provided below to the target user for recommendation.

# Observation:
1: {'title': 'Bourne Ultimatum', 'categories': 'Studio Specials, Universal Studios Home Entertainment, All Universal Studios Titles', 'description': '', 'rating': 5.0, 'review': 'awesome movie'},
(omitted)
5: {'title': 'The Bourne Legacy', 'categories': 'Fully Loaded DVDs, DTS', 'description': 'The Bourne Legacy takes the action-packed Bourne series to an explosive new level. On the verge of having their conspiracy exposed, members of the government's intelligence community will stop at nothing to erase all evidence of their top secret programs - even the agents involved. Aaron Cross (Jeremy Renner) must use his genetically-engineered skills to survive the ultimate game of cat-and-mouse and finish what Jason Bourne started. Also starring Academy Award winner Rachel Weisz and Academy Award nominee Edward Norton, critics are calling this a "thrilling, edge-of-your-seat heart-pounder" (Meg Porter Berns, WSVN-TV (FOX)).', 'rating': 5.0, 'review': 'awesome movie'}}
```

Based on the above user information, please rank the candidate products that align closely with the user's preferences. If item IDs [101, 102, 103] are sorted as first 102, second 103, third 101 in order, present your response in the format below: [3,1,2]. In this example, the number of candidate products was 3, but next, 10 products will be provided, so you SHOULD sort 10 products.

```
# Candidates products:
1: {'title': 'Chicago', 'categories': 'Studio Specials, Miramax Home Entertainment, All Titles', 'description': 'Earning 13 Academy Award nominations including Best Picture, Best Director, Best Actress, and Best Supporting Actress, Chicago stars Catherine Zeta-Jones, Renee Zellweger and Richard Gere. Directed by Rob Marshall, it's a dazzling spectacle that was cheered by both audiences and critics nationwide. PG-13. 107 minutes. DVD.'},
(omitted)
10: {'title': 'Love Streams', 'categories': 'Criterion Collection, All', 'description': 'A troubled divorcee and a moody writer are apart and then together, sharing a special relation.'}}
```

I here repeat candidate instruction. Please rank 10 candidate product IDs as follows: If product IDs [101, 102, 103] are sorted as first 102, second 103, third 101 in order, present your response in the format below: [3,1,2].

```
# Additional constraints:
- You ONLY rank the given Candidate product IDs.
- Do not explain the reason and include any other words.
```

**Figure 4: Baseline**

```
tsys = (default_system_inst)
```

```
-----  
tu1 = (user_info + candidate_info)
```

Our final goal is to provide an answer to the following problem. Before tackling this issue, rephrase and expand it to help you do better answering. Maintain all information in the original question.

t<sub>a1</sub> = The task at hand is to evaluate and rank a list of candidate products based on their alignment with the preferences of a target user. The user has previously shown a strong interest in action-packed films, particularly those in the Bourne series, as well as comedies with high ratings. The goal is to identify which of the new candidate products best match these interests and rank them accordingly.

The candidate products provided include a mix of genres, including action, comedy, and drama. Each product has a title, category, and description, which will help determine how closely they align with the user's preferences.

The ranking should reflect the following considerations:

1. The presence of action elements, particularly in the context of the Bourne series.
2. High ratings and positive reviews, as the user has previously favored products with a perfect rating of 5.0.
3. The relevance of the genre to the user's interests, with a focus on action and comedy.

After analyzing the candidate products, I will provide a ranked list indicating the order of preference for the user.

Now, let's proceed to rank the candidate products based on these criteria.

Here are the candidate products to be ranked:

1. Seven Year Itch VHS (Comedy)  
*(omitted)*
10. Love Streams (Drama)

Based on the analysis of these products, the final ranking will be provided in the specified format.

Now, let's rank the products.

### Ranked List:

1. \*\*The Bourne Identity\*\* (Action) - Strong alignment with the user's interest in the Bourne series.  
*(omitted)*
10. \*\*Cooking French, Vol. 2\*\* (Special Interests) - Least relevant to the user's interests.

### Final Output:

```
[4, 9, 8, 6, 1, 3, 2, 7, 10, 5]
```

```
-----  
tu2 = (final_inst)
```

**Figure 5: Rephrase**

```
tsys = (default_system_inst)
-----
tu1 = (user_info + candidate_info)

Our final goal is to provide an answer to the following problem. Before tackling this issue, please follow this format to proceed step by step with *Observation*, *Thought*, and *Action*:
- Observation: Observe the user's history and preferences.
- Thought: Infer the user's tastes or tendencies from the observation.
- Action: Choose one candidate item and examine its characteristics.
- Observation: Observe the characteristics of that item.
- Thought: Consider whether the item matches the user's preferences.
(Repeat for multiple items if necessary)
Finally, provide your *Answer*.
-----
ta1 = ### Observation:
The user has shown a strong preference for action-packed movies, particularly those in the Bourne series, as well as high-rated comedies like "Tyler Perry's A Madea Christmas." All items in the user's history have a perfect rating of 5.0 and are well-reviewed, indicating a preference for quality content.

### Thought:
The user likely enjoys thrilling action films and comedies with strong narratives and engaging characters. Given this, I will prioritize candidate products that are action-oriented or highly rated comedies.

### Action:
I will examine the candidate items one by one to determine their relevance to the user's preferences.

1. **Seven Year Itch VHS**: Comedy, but lacks a strong description or notable acclaim.
(omitted)
10. **Love Streams**: Drama, not aligned with the user's preferences.

### Observation:
- **The Bourne Identity**: Strongly matches user preferences as it is an action film from a series they enjoy.
- **Underworld: Awakening**: Action-oriented, but may not be as appealing as the Bourne series.
- **Jack Reacher Combo**: Another action film that fits the user's taste for thrillers.

### Thought:
The top candidates based on the user's preferences are "The Bourne Identity," "Jack Reacher Combo," and "Underworld: Awakening." "The Bourne Identity" is the most aligned with the user's established preferences, followed by "Jack Reacher Combo" and then "Underworld: Awakening."

### Answer:
[4, 9, 8]
-----
tu2 = (final_inst)
```

**Figure 6: ReAct**

```
tsys = (default_system_inst)
-----
tu1 = (user_info + candidate_info)

Our final goal is to provide an answer to the following problem. Before tackling this issue, please consider the principles and theories behind this question.
-----
ta1 = To rank the candidate products based on the user's preferences, we should consider the following factors:
1. **Genre and Themes**: The user has shown a preference for action and thriller movies, particularly those in the Bourne series, which are characterized by intense action and espionage themes.
2. **Ratings**: Products with higher ratings (5.0 in this case) are more favorable.
3. **Familiarity with Series**: The user has already shown interest in the Bourne series, so products related to that series should be prioritized.
4. **Overall Appeal**: Products that are critically acclaimed or have a strong following may also be ranked higher.

Now, let's analyze the candidate products:
1. **The Bourne Identity** (4) - This is directly related to the Bourne series and fits the user's established interest in action and espionage. (omitted)
10. **Love Streams** (10) - A drama that does not fit the user's established preferences for action and thrillers.

Based on this analysis, the ranked list of candidate products is:
[4, 8, 9, 2, 3, 6, 7, 1, 5, 10]
-----
tu2 = (final_inst)
```

**Figure 7: Step-Back**

```
tsys = (default_system_inst)
-----
tu1 = (user_info + candidate_info)
-----
ta1 = Let's think step by step.
-----
tu2 = OK! Go ahead.
-----
ta2 = To rank the candidate products, we will consider the following factors based on the user's previous preferences:
1. **Genre**: The user seems to prefer action and thriller movies, particularly those related to the Bourne series.
2. **Rating**: Higher-rated products are more likely to be recommended.
3. **Description**: Products with engaging descriptions that align with the user's interests will be prioritized.

Now, let's analyze the candidate products:
1. **The Bourne Identity** (Product 4): This is directly related to the Bourne series, which the user has shown a strong preference for. It has a compelling description and fits the action/thriller genre. (omitted)
10. **Love Streams** (Product 10): This is a drama that may not resonate with the user's action/thriller interests.

Based on this analysis, the ranked list of candidate products is:
[4, 8, 9, 2, 3, 6, 1, 7, 5, 10]
-----
tu3 = (final_inst)
```

**Figure 8: Step-by-Step**

```
tsys = You are an AI assistant that pretends to be a person who has interacted with the following items.
# Logs: (user_item_history)
-----
tu1 = Please rank the candidate products that align closely with your preferences. If item IDs [101, 102, 103] are sorted as first 102, second 103, third 101 in order, present your response in the format below: [3,1,2]. In this example, the number of candidate products was 3, but next, 10 products will be provided, so you SHOULD sort 10 products.
# Candidates products: (candidate_items)
(final_inst)
```

**Figure 9: Pretend-User**

```
tsys = (default_system_inst)
-----
tu1 = # Requirements:
you must rank candidate products that will be provided below to the target user for recommendation.

# Observation:
{1: {'summary': '**Item Summary: "Bourne Ultimatum"**\n\n- **Title:** Bourne Ultimatum\n- **Categories:** This item falls under several categories, including Studio Specials, Universal Studios Home Entertainment, and All Universal Studios Titles. This classification highlights its association with a prominent studio known for quality entertainment.\n- **Description:** Currently, there is no detailed description provided for the item.\n\n"Bourne Ultimatum" is a notable film that is part of the acclaimed Bourne series, recognized for its thrilling action and intricate plot. As part of Universal Studios' collection, it reflects the studio's commitment to delivering high-quality cinematic experiences.', 'rating': 5.0, 'review': 'awesome movie'},
 (committed)
5: {'summary': '**Title:** The Bourne Legacy \n**Categories:** Fully Loaded DVDs, DTS \n**Description:** "The Bourne Legacy" elevates the action-packed Bourne series as it follows Aaron Cross (Jeremy Renner), a genetically-engineered agent, who must navigate a dangerous game of cat-and-mouse to survive after the government's intelligence community seeks to eliminate all evidence of their covert programs. With a star-studded cast including Academy Award winners Rachel Weisz and Edward Norton, the film is praised as a "thrilling, edge-of-your-seat heart-pounder" by critics. This installment promises explosive action and suspense, making it a must-watch for fans of the franchise.', 'rating': 5.0,
 'review': 'awesome movie'}}}

(candidate_info)
(final_inst)
```

**Figure 10: Summarize-Item**

```
tsys = (default_system_inst)
-----
tu1 = (user_info + candidate_info)

Our final goal is to provide an answer to the ranking item problem. Before tackling this issue, rephrase and expand it to help you do better answering. Maintain all information in the original question.
-----
ta1 = # Task Overview:
You are tasked with ranking a set of candidate products based on their alignment with user preferences. The user has previously shown a strong interest in action-packed films, particularly those in the Bourne series, as well as comedies that are well-reviewed. The goal is to recommend the top products that best match these interests.

# User Preferences:
The user has expressed a preference for movies with high ratings and engaging content. The previous products they liked include:
1. **Bourne Ultimatum** - Action-packed with a perfect rating of 5.0.
(omitted)
5. **The Bourne Legacy** - A thrilling addition to the Bourne franchise, rated 5.0.

# Candidate Products:
You will evaluate the following 10 candidate products to determine which ones best align with the user's preferences for high-rated action and comedy films:
1. **Seven Year Itch VHS** - Comedy genre.
(omitted)
10. **Love Streams** - A drama film.

# Objective:
Your task is to rank these 10 candidate products based on how closely they align with the user's preferences for action and comedy films, particularly those that are highly rated. The ranking should be presented in the format of item IDs sorted by their relevance to the user's interests.

Please provide the ranked list of candidate products.
-----
tu2 = Our final goal is to provide an answer to the ranking item problem. Before tackling this issue, please follow this format to proceed step by step with *Observation*, *Thought*, and *Action*:
- Observation: Observe the user's history and preferences.
- Thought: Infer the user's tastes or tendencies from the observation.
- Action: Choose one candidate item and examine its characteristics.
- Observation: Observe the characteristics of that item.
- Thought: Consider whether the item matches the user's preferences.
(Repeat for multiple items if necessary)
Finally, provide your *Answer*.
-----
ta2 = ### Step-by-Step Process

**Observation:** The user has a strong preference for action-packed films, particularly those in the Bourne series, as well as comedies that are well-reviewed. All previously liked products have a perfect rating of 5.0.

**Thought:** The user enjoys high-quality films that are either thrilling or humorous. The Bourne series is a clear favorite, indicating a strong preference for action and espionage themes. Comedies that are well-received also appeal to the user.

---
(omitted)
---
(omitted)
---
(omitted)
---
(omitted)
---
(omitted)

**Final Evaluation of All Candidates:**
1. **The Bourne Identity** - Strong match for action preference.
(omitted)
5. **Chicago** - Critically acclaimed but not action or comedy.

### Answer:
Based on the evaluations, the ranked list of candidate products is: *[4, 9, 8, 3, 2]*.
-----
tu3 = (final_inst)
```

Figure 11: Rephrase → ReAct

```
tsys = (default_system_inst)
```

```
-----
```

```
tu1 = (user_info + candidate_info)
```

Our final goal is to provide an answer to the ranking item problem. Before tackling this issue, please consider the principles and theories behind the first question.

```
-----
```

```
ta1 = To rank the candidate products based on the user's preferences, we will focus on the following criteria:
```

1. **Genre and Themes**: The user has shown a preference for action and thriller movies, particularly those in the Bourne series, which are characterized by intense action and espionage themes.
2. **Ratings**: Higher-rated products (5.0 ratings) are prioritized.
3. **Relevance**: Products that are part of the same franchise or have similar themes to the Bourne series will be ranked higher.

Given these criteria, let's analyze the candidate products:

1. **The Bourne Identity** (Product 4): This is directly related to the Bourne series, which the user has shown a strong preference for. It has a similar action and espionage theme.  
*(omitted)*
10. **Love Streams** (Product 10): This is a drama and does not fit the user's action-oriented preferences.

Based on this analysis, the ranked list of candidate products is as follows:

```
[4, 8, 9, 2, 3, 6, 7, 1, 5, 10]
```

```
-----
```

```
tu2 = Thank you! As an expert, what do you think about the above answers? Please provide feedback so that more accurate predictions can be made in the future.
```

```
-----
```

```
ta2 = The ranking provided is a solid attempt to align the candidate products with the user's preferences based on the information available. However, there are several areas for improvement to enhance the accuracy of future predictions:
```

1. **User Preference Analysis**: While the analysis focused on action and thriller genres, it could benefit from a deeper understanding of the user's overall preferences. For instance, if the user enjoys a mix of genres, including comedy or drama, those products could be ranked higher.  
*(omitted)*
7. **Feedback Loop**: Implementing a feedback mechanism where users can indicate their satisfaction with recommendations could help refine future predictions. This would allow for continuous learning and improvement based on user responses.

By incorporating these considerations, future rankings can become more nuanced and tailored to the user's preferences, leading to more accurate and satisfying recommendations.

```
-----
```

```
tu3 = (final_inst)
```

**Figure 12: SelfRefine (Step-Back)**

**Table 14: llama3.3-70b**

	Music	Movie	Groceries	Clothes	Book	Yelp	News	Food	Heavy	Light
Baseline	0.605	0.506	0.478	0.408	0.574	0.475	0.157	0.312	0.440	0.439
Emotion	0.607	0.514	0.484	0.406	0.575	0.484	0.177	0.290	0.439	0.445
Re-Reading	0.631	0.526	0.554*	0.441	0.577	0.507*	0.233*	0.308	0.471*	0.474*
Both-Inst	0.607	0.502	0.495	0.404	0.561	0.507*	0.169	0.248 <sup>▽</sup>	0.437	0.436
Recency-Focused	0.627	0.523	0.495	0.473*	0.601	0.468	0.248*	0.324	0.467*	0.473*
RolePlay-User	0.606	0.512	0.482	0.446*	0.560	0.481	0.169	0.315	0.461*	0.432
RolePlay-Expert	0.597	0.516	0.490	0.400	0.561 <sup>▽</sup>	0.490	0.154	0.313	0.439	0.441
RolePlay-Frederick	0.605	0.505	0.480	0.385 <sup>▽</sup>	0.566	0.472	0.160	0.319	0.435	0.438
Summarize-Item	0.586	0.495	0.451	0.441	0.573	0.453	0.284*	0.281	0.459*	0.432
Step-Back	0.622	0.578*	0.473	0.458	0.621*	0.553*	0.208*	0.338	0.480*	0.483*
ReAct	0.645	0.549*	0.480	0.453	0.636*	0.566*	0.246*	0.385*	0.507*	0.483*
Rephrase	0.631	0.569*	0.460	0.441	0.602	0.530*	0.185	0.323	0.492*	0.443
Echo	0.630	0.547*	0.540*	0.458*	0.591	0.501	0.255*	0.295	0.481*	0.473*
Summarize-User	0.624	0.536*	0.530*	0.440	0.597	0.504*	0.164	0.341	0.467*	0.467*
Generate-Item	0.604	0.506	0.481	0.406	0.574	0.475	0.163	0.295	0.438	0.438
Reuse-Item	0.603	0.530	0.523*	0.423	0.592	0.519*	0.227*	0.318	0.451	0.483*
Explain	0.605	0.541*	0.520*	0.450*	0.582	0.547*	0.145	0.332	0.458*	0.473*
Mock	0.595	0.531*	0.495	0.411	0.556	0.481	0.220*	0.320	0.442	0.460*
Step-by-Step	0.599	0.525	0.484	0.394	0.567	0.481	0.169	0.294	0.439	0.440
Deep-Breath	0.595	0.527*	0.479	0.395	0.566	0.484	0.179	0.295	0.437	0.442
Plan-Solve	0.614	0.499	0.473	0.390	0.579	0.468	0.197*	0.316	0.434	0.450

**Table 15: gpt-4o-mini**

	Music	Movie	Groceries	Clothes	Book	Yelp	News	Food	Heavy	Light
Baseline	0.541	0.502	0.440	0.393	0.518	0.501	0.200	0.364	0.444	0.421
Emotion	0.561	0.527	0.445	0.400	0.543	0.479	0.204	0.360	0.450	0.430
Re-Reading	0.456 <sup>▽</sup>	0.483	0.471	0.377	0.436 <sup>▽</sup>	0.439 <sup>▽</sup>	0.219	0.335	0.431	0.374 <sup>▽</sup>
Both-Inst	0.586*	0.484	0.437	0.397	0.520	0.485	0.194	0.341	0.432 <sup>▽</sup>	0.429
Recency-Focused	0.565	0.497	0.466	0.398	0.532	0.480	0.230	0.375	0.449	0.437
RolePlay-User	0.591*	0.492	0.358 <sup>▽</sup>	0.344 <sup>▽</sup>	0.545	0.493	0.116 <sup>▽</sup>	0.348	0.426 <sup>▽</sup>	0.396 <sup>▽</sup>
RolePlay-Expert	0.566	0.503	0.441	0.383	0.513	0.505	0.191	0.344	0.443	0.418
RolePlay-Frederick	0.556	0.487	0.445	0.376	0.517	0.481	0.205	0.345	0.430 <sup>▽</sup>	0.423
Summarize-Item	0.546	0.465 <sup>▽</sup>	0.407	0.392	0.558	0.485	0.200	0.367	0.441	0.414
Step-Back	0.581	0.570*	0.457	0.423	0.545	0.455 <sup>▽</sup>	0.270*	0.375	0.465*	0.454*
ReAct	0.394 <sup>▽</sup>	0.443 <sup>▽</sup>	0.423	0.355	0.300 <sup>▽</sup>	0.339 <sup>▽</sup>	0.449*	0.245 <sup>▽</sup>	0.378 <sup>▽</sup>	0.359 <sup>▽</sup>
Rephrase	0.563	0.555*	0.433	0.408	0.521	0.521	0.219	0.345	0.452	0.439*
Echo	0.463 <sup>▽</sup>	0.458 <sup>▽</sup>	0.485*	0.357	0.415 <sup>▽</sup>	0.458 <sup>▽</sup>	0.194	0.344	0.424 <sup>▽</sup>	0.370 <sup>▽</sup>
Summarize-User	0.579	0.546*	0.475*	0.399	0.574*	0.464	0.258*	0.325 <sup>▽</sup>	0.456	0.449*
Generate-Item	0.584*	0.558*	0.428	0.396	0.519	0.501	0.195	0.379	0.454	0.436
Reuse-Item	0.524	0.528	0.460	0.394	0.499	0.429 <sup>▽</sup>	0.247*	0.338	0.427 <sup>▽</sup>	0.428
Explain	0.606*	0.541*	0.450	0.414	0.584*	0.502	0.222	0.361	0.467*	0.453*
Mock	0.509	0.509	0.414 <sup>▽</sup>	0.341 <sup>▽</sup>	0.509	0.444 <sup>▽</sup>	0.181 <sup>▽</sup>	0.353	0.413 <sup>▽</sup>	0.402 <sup>▽</sup>
Step-by-Step	0.541	0.508	0.454	0.396	0.507	0.468 <sup>▽</sup>	0.211	0.337	0.433 <sup>▽</sup>	0.422
Deep-Breath	0.532	0.493	0.454	0.398	0.495	0.480	0.213	0.343	0.431 <sup>▽</sup>	0.421
Plan-Solve	0.504 <sup>▽</sup>	0.516	0.439	0.387	0.489	0.484	0.191	0.368	0.424 <sup>▽</sup>	0.421

**Table 16: phi4**

	Music	Movie	Groceries	Clothes	Book	Yelp	News	Food	Heavy	Light
Baseline	0.591	0.462	0.464	<u>0.471</u>	0.595	0.331	0.145	0.278	0.423	0.411
Emotion	0.573	0.446	0.483	0.451	0.580	0.352	0.131	<u>0.319*</u>	0.428	0.406
Re-Reading	0.540 <sup>▽</sup>	0.457	0.478	0.469	0.533 <sup>▽</sup>	0.306	0.244*	0.244	0.432	0.386 <sup>▽</sup>
Both-Inst	0.553 <sup>▽</sup>	0.455	0.462	0.435 <sup>▽</sup>	0.580	0.360	0.083 <sup>▽</sup>	0.293	0.417	0.388 <sup>▽</sup>
Recency-Focused	0.562	0.455	0.457	0.429	<u>0.598</u>	0.352	0.131	0.292	0.424	0.395
RolePlay-User	0.529 <sup>▽</sup>	0.423 <sup>▽</sup>	0.395 <sup>▽</sup>	0.399 <sup>▽</sup>	0.573	0.320	0.042 <sup>▽</sup>	0.316	0.391 <sup>▽</sup>	0.358 <sup>▽</sup>
RolePlay-Expert	0.581	0.461	0.469	0.451	0.592	0.322	0.132	<u>0.310*</u>	0.423	0.406
RolePlay-Frederick	0.568	0.475	0.470	0.440	0.561 <sup>▽</sup>	0.382*	0.152	0.295	0.435	0.401
Summarize-Item	0.434 <sup>▽</sup>	0.418	0.395 <sup>▽</sup>	0.422	0.511 <sup>▽</sup>	0.372	0.118	0.302	0.377 <sup>▽</sup>	0.366 <sup>▽</sup>
Step-Back	0.530 <sup>▽</sup>	0.467	0.398 <sup>▽</sup>	0.373 <sup>▽</sup>	0.493 <sup>▽</sup>	0.330	0.605*	0.290	0.447	<u>0.424</u>
ReAct	0.566	<u>0.497</u>	<u>0.483</u>	0.411 <sup>▽</sup>	0.510 <sup>▽</sup>	0.278 <sup>▽</sup>	0.286*	0.274	0.426	0.400
Rephrase	0.554	0.445	0.451	0.432	0.533 <sup>▽</sup>	0.348	0.370*	0.301	0.454*	0.404
Echo	0.552	<u>0.510*</u>	0.472	0.459	0.523 <sup>▽</sup>	0.351	0.211*	0.281	<u>0.457*</u>	0.383 <sup>▽</sup>
Summarize-User	<u>0.597</u>	0.478	0.447	<u>0.422</u> <sup>▽</sup>	0.572	0.372*	0.173	0.272	0.450*	0.383 <sup>▽</sup>
Generate-Item	0.534 <sup>▽</sup>	0.474	0.475	0.433 <sup>▽</sup>	0.551 <sup>▽</sup>	0.373*	0.140	0.268	0.436	0.376 <sup>▽</sup>
Reuse-Item	0.554	0.416	0.414	0.409 <sup>▽</sup>	0.586	0.343	0.098 <sup>▽</sup>	0.233	0.403	0.360 <sup>▽</sup>
Explain	0.560	0.488	0.474	0.402 <sup>▽</sup>	0.581	<u>0.372*</u>	0.165	0.278	0.446*	0.384 <sup>▽</sup>
Mock	0.568	0.436	0.450	0.402 <sup>▽</sup>	0.584	0.332	0.092 <sup>▽</sup>	0.299	0.405	0.386 <sup>▽</sup>
Step-by-Step	0.489 <sup>▽</sup>	0.470	0.425	0.395 <sup>▽</sup>	0.468 <sup>▽</sup>	0.308	0.440*	0.270	0.420	0.397
Deep-Breath	0.481 <sup>▽</sup>	0.460	0.399 <sup>▽</sup>	0.371 <sup>▽</sup>	0.511 <sup>▽</sup>	0.315	0.473*	0.246	0.400	0.415
Plan-Solve	0.478 <sup>▽</sup>	0.425	0.389 <sup>▽</sup>	0.358 <sup>▽</sup>	0.446 <sup>▽</sup>	0.288	<u>0.606*</u>	0.230	0.406	0.398

**Table 17: amazon-nova-lite**

	Music	Movie	Groceries	Clothes	Book	Yelp	News	Food	Heavy	Light
Baseline	0.441	0.401	0.414	0.367	0.470	0.447	0.267	0.305	0.360	0.418
Emotion	0.425	0.418	0.382	0.385	0.461	0.404 <sup>▽</sup>	0.265	0.266	0.347	0.404
Re-Reading	<u>0.509*</u>	0.381	<u>0.421</u>	0.378	0.494	0.389 <sup>▽</sup>	0.316*	0.258	0.379	0.408
Both-Inst	0.426	0.413	0.379	<u>0.412*</u>	0.475	0.433	0.383*	0.249 <sup>▽</sup>	0.364	0.429
Recency-Focused	0.481	0.415	0.412	<u>0.433*</u>	0.450	0.402 <sup>▽</sup>	0.272	0.289	0.357	<u>0.432</u>
RolePlay-User	0.428	0.431	0.331 <sup>▽</sup>	0.369	0.426	0.425	0.439*	0.280	0.372	0.411
RolePlay-Expert	0.478	<u>0.440*</u>	0.387	0.392	0.442	0.397 <sup>▽</sup>	0.254	0.273	0.358	0.408
RolePlay-Frederick	0.457	0.393	0.380	<u>0.404*</u>	0.427 <sup>▽</sup>	0.437	0.289	0.241 <sup>▽</sup>	0.354	0.403 <sup>▽</sup>
Summarize-Item	0.443	<u>0.447</u>	0.356	0.373	0.428	0.404	0.262	0.310	0.355	0.400
Step-Back	0.378	0.384	0.325 <sup>▽</sup>	0.302 <sup>▽</sup>	0.345 <sup>▽</sup>	0.323 <sup>▽</sup>	<u>0.517*</u>	0.213 <sup>▽</sup>	0.344	0.353 <sup>▽</sup>
ReAct	0.396	0.363	0.375	0.326	0.469	0.330 <sup>▽</sup>	0.426*	0.268	0.384	0.355 <sup>▽</sup>
Rephrase	<u>0.497*</u>	<u>0.457*</u>	0.380	0.356	<u>0.502</u>	0.427	0.249	0.313	<u>0.391*</u>	0.405
Echo	0.446	0.341 <sup>▽</sup>	0.350 <sup>▽</sup>	0.330	0.442	0.324 <sup>▽</sup>	0.324*	0.240 <sup>▽</sup>	0.331 <sup>▽</sup>	0.368 <sup>▽</sup>
Summarize-User	0.441	0.383	0.400	0.351	0.436	0.394 <sup>▽</sup>	0.301	0.270	0.358	0.386 <sup>▽</sup>
Generate-Item	0.424	0.404	0.396	0.379	0.482	0.438	0.281	0.257 <sup>▽</sup>	0.360	0.405
Reuse-Item	0.423	0.389	0.386	0.372	0.383 <sup>▽</sup>	0.386 <sup>▽</sup>	0.409*	0.212 <sup>▽</sup>	0.337 <sup>▽</sup>	0.404
Explain	0.436	0.398	0.370 <sup>▽</sup>	0.366	0.478	0.439	0.262	0.285	0.366	0.392 <sup>▽</sup>
Mock	0.443	0.386	0.387	0.360	0.443	0.372 <sup>▽</sup>	0.365*	0.228 <sup>▽</sup>	0.353	0.393 <sup>▽</sup>
Step-by-Step	0.471	0.416	0.362 <sup>▽</sup>	0.366	0.451	0.420	0.291	0.281	0.352	0.413
Deep-Breath	0.432	0.372	0.393	0.344	0.466	0.421	0.269	0.290	0.350	0.397 <sup>▽</sup>
Plan-Solve	<u>0.492*</u>	0.343 <sup>▽</sup>	0.274 <sup>▽</sup>	0.267 <sup>▽</sup>	0.405 <sup>▽</sup>	0.362 <sup>▽</sup>	0.409*	<u>0.314</u>	0.374	0.342 <sup>▽</sup>

**Table 18: Music**

	gpt-4.1-mini	llama3.3-70b	gpt-4o-mini	phi4	amazon-nova-lite
Baseline	0.610	0.605	0.541	0.591	0.441
Emotion	0.608	0.607	0.561	0.573	0.425
Re-Reading	0.567 <sup>▽</sup>	0.631	0.456 <sup>▽</sup>	0.540 <sup>▽</sup>	0.509*
Both-Inst	0.609	0.607	0.586*	0.553 <sup>▽</sup>	0.426
Recency-Focused	0.601	0.627	0.565	0.562	0.481
RolePlay-User	0.540 <sup>▽</sup>	0.606	0.591*	0.529 <sup>▽</sup>	0.428
RolePlay-Expert	0.592	0.597	0.566	0.581	0.478
RolePlay-Frederick	0.588 <sup>▽</sup>	0.605	0.556	0.568	0.457
Summarize-Item	0.536 <sup>▽</sup>	0.586	0.546	0.434 <sup>▽</sup>	0.443
Step-Back	0.642	0.622	0.581	0.530 <sup>▽</sup>	0.378
ReAct	0.632	0.645	0.394 <sup>▽</sup>	0.566	0.396
Rephrase	0.626	0.631	0.563	0.554	0.497*
Echo	0.594	0.630	0.463 <sup>▽</sup>	0.552	0.446
Summarize-User	0.646*	0.624	0.579	0.597	0.441
Generate-Item	0.612	0.604	0.584*	0.534 <sup>▽</sup>	0.424
Reuse-Item	0.606	0.603	0.524	0.554	0.423
Explain	0.616	0.605	0.606*	0.560	0.436
Mock	0.614	0.595	0.509	0.568	0.443
Step-by-Step	0.608	0.599	0.541	0.489 <sup>▽</sup>	0.471
Deep-Breath	0.615	0.595	0.532	0.481 <sup>▽</sup>	0.432
Plan-Solve	0.602	0.614	0.504 <sup>▽</sup>	0.478 <sup>▽</sup>	0.492*

**Table 19: Movies**

	gpt-4.1-mini	llama3.3-70b	gpt-4o-mini	phi4	amazon-nova-lite
Baseline	0.516	0.506	0.502	0.462	0.401
Emotion	0.491 <sup>▽</sup>	0.514	0.527	0.446	0.418
Re-Reading	0.487	0.526	0.483	0.457	0.381
Both-Inst	0.513	0.502	0.484	0.455	0.413
Recency-Focused	0.519	0.523	0.497	0.455	0.415
RolePlay-User	0.484	0.512	0.492	0.423 <sup>▽</sup>	0.431
RolePlay-Expert	0.507	0.516	0.503	0.461	0.440*
RolePlay-Frederick	0.505	0.505	0.487	0.475	0.393
Summarize-Item	0.495	0.495	0.465 <sup>▽</sup>	0.418	0.447
Step-Back	0.561*	0.578*	0.570*	0.467	0.384
ReAct	0.566*	0.549*	0.443 <sup>▽</sup>	0.497	0.363
Rephrase	0.564*	0.569*	0.555*	0.445	0.457*
Echo	0.525	0.547*	0.458 <sup>▽</sup>	0.510*	0.341 <sup>▽</sup>
Summarize-User	0.523	0.536*	0.546*	0.478	0.383
Generate-Item	0.515	0.506	0.558*	0.474	0.404
Reuse-Item	0.551*	0.530	0.528	0.416	0.389
Explain	0.514	0.541*	0.541*	0.488	0.398
Mock	0.495	0.531*	0.509	0.436	0.386
Step-by-Step	0.511	0.525	0.508	0.470	0.416
Deep-Breath	0.517	0.527*	0.493	0.460	0.372
Plan-Solve	0.501	0.499	0.516	0.425	0.343 <sup>▽</sup>

**Table 20: Groceries**

	gpt-4.1-mini	llama3.3-70b	gpt-4o-mini	phi4	amazon-nova-lite
Baseline	0.443	0.478	0.440	0.464	0.414
Emotion	0.438	0.484	0.445	0.483	0.382
Re-Reading	0.420	0.554*	0.471	0.478	0.421
Both-Inst	0.443	0.495	0.437	0.462	0.379
Recency-Focused	0.466	0.495	0.466	0.457	0.412
RolePlay-User	0.372 <sup>▽</sup>	0.482	0.358 <sup>▽</sup>	0.395 <sup>▽</sup>	0.331 <sup>▽</sup>
RolePlay-Expert	0.441	0.490	0.441	0.469	0.387
RolePlay-Frederick	0.441	0.480	0.445	0.470	0.380
Summarize-Item	0.401	0.451	0.407	0.395 <sup>▽</sup>	0.356
Step-Back	0.469	0.473	0.457	0.398 <sup>▽</sup>	0.325 <sup>▽</sup>
ReAct	0.511*	0.480	0.423	0.483	0.375
Rephrase	0.461	0.460	0.433	0.451	0.380
Echo	0.426	0.540*	0.485*	0.472	0.350 <sup>▽</sup>
Summarize-User	0.472	0.530*	0.475*	0.447	0.400
Generate-Item	0.457	0.481	0.428	0.475	0.396
Reuse-Item	0.415	0.523*	0.460	0.414	0.386
Explain	0.476*	0.520*	0.450	0.474	0.370 <sup>▽</sup>
Mock	0.439	0.495	0.414 <sup>▽</sup>	0.450	0.387
Step-by-Step	0.424	0.484	0.454	0.425	0.362 <sup>▽</sup>
Deep-Breath	0.436	0.479	0.454	0.399 <sup>▽</sup>	0.393
Plan-Solve	0.443	0.473	0.439	0.389 <sup>▽</sup>	0.274 <sup>▽</sup>

**Table 21: Clothes**

	gpt-4.1-mini	llama3.3-70b	gpt-4o-mini	phi4	amazon-nova-lite
Baseline	0.408	0.408	0.393	0.471	0.367
Emotion	0.411	0.406	0.400	0.451	0.385
Re-Reading	0.400	0.441	0.377	0.469	0.378
Both-Inst	0.398	0.404	0.397	0.435 <sup>▽</sup>	0.412*
Recency-Focused	0.420	0.473*	0.398	0.429	0.433*
RolePlay-User	0.325 <sup>▽</sup>	0.446*	0.344 <sup>▽</sup>	0.399 <sup>▽</sup>	0.369
RolePlay-Expert	0.380 <sup>▽</sup>	0.400	0.383	0.451	0.392
RolePlay-Frederick	0.403	0.385 <sup>▽</sup>	0.376	0.440	0.404*
Summarize-Item	0.421	0.441	0.392	0.422	0.373
Step-Back	0.437	0.458	0.423	0.373 <sup>▽</sup>	0.302 <sup>▽</sup>
ReAct	0.448	0.453	0.355	0.411 <sup>▽</sup>	0.326
Rephrase	0.437	0.441	0.408	0.432	0.356
Echo	0.383	0.458*	0.357	0.459	0.330
Summarize-User	0.406	0.440	0.399	0.422 <sup>▽</sup>	0.351
Generate-Item	0.419	0.406	0.396	0.433 <sup>▽</sup>	0.379
Reuse-Item	0.391	0.423	0.394	0.409 <sup>▽</sup>	0.372
Explain	0.412	0.450*	0.414	0.402 <sup>▽</sup>	0.366
Mock	0.415	0.411	0.341 <sup>▽</sup>	0.402 <sup>▽</sup>	0.360
Step-by-Step	0.428	0.394	0.396	0.395 <sup>▽</sup>	0.366
Deep-Breath	0.428*	0.395	0.398	0.371 <sup>▽</sup>	0.344
Plan-Solve	0.427	0.390	0.387	0.358 <sup>▽</sup>	0.267 <sup>▽</sup>

**Table 22: Books**

	gpt-4.1-mini	llama3.3-70b	gpt-4o-mini	phi4	amazon-nova-lite
Baseline	0.585	0.574	0.518	0.595	0.470
Emotion	0.582	0.575	0.543	0.580	0.461
Re-Reading	0.553 <sup>▽</sup>	0.577	0.436 <sup>▽</sup>	0.533 <sup>▽</sup>	0.494
Both-Inst	0.566	0.561	0.520	0.580	0.475
Recency-Focused	0.591	0.601	0.532	0.598	0.450
RolePlay-User	0.533 <sup>▽</sup>	0.560	0.545	0.573	0.426
RolePlay-Expert	0.587	0.561 <sup>▽</sup>	0.513	0.592	0.442
RolePlay-Frederick	0.569	0.566	0.517	0.561 <sup>▽</sup>	0.427 <sup>▽</sup>
Summarize-Item	0.536 <sup>▽</sup>	0.573	0.558	0.511 <sup>▽</sup>	0.428
Step-Back	0.594	0.621*	0.545	0.493 <sup>▽</sup>	0.345 <sup>▽</sup>
ReAct	0.608	0.636*	0.300 <sup>▽</sup>	0.510 <sup>▽</sup>	0.469
Rephrase	0.627*	0.602	0.521	0.533 <sup>▽</sup>	0.502
Echo	0.546 <sup>▽</sup>	0.591	0.415 <sup>▽</sup>	0.523 <sup>▽</sup>	0.442
Summarize-User	0.580	0.597	0.574*	0.572	0.436
Generate-Item	0.581	0.574	0.519	0.551 <sup>▽</sup>	0.482
Reuse-Item	0.578	0.592	0.499	0.586	0.383 <sup>▽</sup>
Explain	0.571	0.582	0.584*	0.581	0.478
Mock	0.580	0.556	0.509	0.584	0.443
Step-by-Step	0.580	0.567	0.507	0.468 <sup>▽</sup>	0.451
Deep-Breath	0.591	0.566	0.495	0.511 <sup>▽</sup>	0.466
Plan-Solve	0.573	0.579	0.489	0.446 <sup>▽</sup>	0.405 <sup>▽</sup>

**Table 23: Yelp**

	gpt-4.1-mini	llama3.3-70b	gpt-4o-mini	phi4	amazon-nova-lite
Baseline	0.512	0.475	0.501	0.331	0.447
Emotion	0.503	0.484	0.479	0.352	0.404 <sup>▽</sup>
Re-Reading	0.421 <sup>▽</sup>	0.507*	0.439 <sup>▽</sup>	0.306	0.389 <sup>▽</sup>
Both-Inst	0.544*	0.507*	0.485	0.360	0.433
Recency-Focused	0.526	0.468	0.480	0.352	0.402 <sup>▽</sup>
RolePlay-User	0.500	0.481	0.493	0.320	0.425
RolePlay-Expert	0.515	0.490	0.505	0.322	0.397 <sup>▽</sup>
RolePlay-Frederick	0.511	0.472	0.481	0.382*	0.437
Summarize-Item	0.481	0.453	0.485	0.372	0.404
Step-Back	0.521	0.553*	0.455 <sup>▽</sup>	0.330	0.323 <sup>▽</sup>
ReAct	0.548*	0.566*	0.339 <sup>▽</sup>	0.278 <sup>▽</sup>	0.330 <sup>▽</sup>
Rephrase	0.517	0.530*	0.521	0.348	0.427
Echo	0.449 <sup>▽</sup>	0.501	0.458 <sup>▽</sup>	0.351	0.324 <sup>▽</sup>
Summarize-User	0.506	0.504*	0.464	0.372*	0.394 <sup>▽</sup>
Generate-Item	0.532	0.475	0.501	0.373*	0.438
Reuse-Item	0.527	0.519*	0.429 <sup>▽</sup>	0.343	0.386 <sup>▽</sup>
Explain	0.530	0.547*	0.502	0.372*	0.439
Mock	0.521	0.481	0.444 <sup>▽</sup>	0.332	0.372 <sup>▽</sup>
Step-by-Step	0.509	0.481	0.468 <sup>▽</sup>	0.308	0.420
Deep-Breath	0.525	0.484	0.480	0.315	0.421
Plan-Solve	0.526	0.468	0.484	0.288 <sup>▽</sup>	0.362 <sup>▽</sup>

**Table 24: News**

	gpt-4.1-mini	llama3.3-70b	gpt-4o-mini	phi4	amazon-nova-lite
Baseline	0.262	0.157	0.200	0.145	0.267
Emotion	0.271	0.177	0.204	0.131	0.265
Re-Reading	0.275	0.233*	0.219	0.244*	0.316*
Both-Inst	0.226 <sup>▽</sup>	0.169	0.194	0.083 <sup>▽</sup>	0.383*
Recency-Focused	0.245	0.248*	0.230	0.131	0.272
RolePlay-User	0.195 <sup>▽</sup>	0.169	0.116 <sup>▽</sup>	0.042 <sup>▽</sup>	0.439*
RolePlay-Expert	0.260	0.154	0.191	0.132	0.254
RolePlay-Frederick	0.261	0.160	0.205	0.152	0.289
Summarize-Item	0.234	0.284*	0.200	0.118	0.262
Step-Back	0.246	0.208*	0.270*	0.605*	0.517*
ReAct	0.248	0.246*	0.449*	0.286*	0.426*
Rephrase	0.253	0.185	0.219	0.370*	0.249
Echo	0.270	0.255*	0.194	0.211*	0.324*
Summarize-User	0.266	0.164	0.258*	0.173	0.301
Generate-Item	0.268	0.163	0.195	0.140	0.281
Reuse-Item	0.229 <sup>▽</sup>	0.227*	0.247*	0.098 <sup>▽</sup>	0.409*
Explain	0.261	0.145	0.222	0.165	0.262
Mock	0.235 <sup>▽</sup>	0.220*	0.181 <sup>▽</sup>	0.092 <sup>▽</sup>	0.365*
Step-by-Step	0.277	0.169	0.211	0.440*	0.291
Deep-Breath	0.286*	0.179	0.213	0.473*	0.269
Plan-Solve	0.259	0.197*	0.191	0.606*	0.409*

**Table 25: Food**

	gpt-4.1-mini	llama3.3-70b	gpt-4o-mini	phi4	amazon-nova-lite
Baseline	0.328	0.312	0.364	0.278	0.305
Emotion	0.327	0.290	0.360	0.319*	0.266
Re-Reading	0.304	0.308	0.335	0.244	0.258
Both-Inst	0.316	0.248 <sup>▽</sup>	0.341	0.293	0.249 <sup>▽</sup>
Recency-Focused	0.361	0.324	0.375	0.292	0.289
RolePlay-User	0.317	0.315	0.348	0.316	0.280
RolePlay-Expert	0.305	0.313	0.344	0.310*	0.273
RolePlay-Frederick	0.303	0.319	0.345	0.295	0.241 <sup>▽</sup>
Summarize-Item	0.318	0.281	0.367	0.302	0.310
Step-Back	0.340	0.338	0.375	0.290	0.213 <sup>▽</sup>
ReAct	0.334	0.385*	0.245 <sup>▽</sup>	0.274	0.268
Rephrase	0.359	0.323	0.345	0.301	0.313
Echo	0.282	0.295	0.344	0.281	0.240 <sup>▽</sup>
Summarize-User	0.324	0.341	0.325 <sup>▽</sup>	0.272	0.270
Generate-Item	0.306 <sup>▽</sup>	0.295	0.379	0.268	0.257 <sup>▽</sup>
Reuse-Item	0.329	0.318	0.338	0.233	0.212 <sup>▽</sup>
Explain	0.326	0.332	0.361	0.278	0.285
Mock	0.320	0.320	0.353	0.299	0.228 <sup>▽</sup>
Step-by-Step	0.326	0.294	0.337	0.270	0.281
Deep-Breath	0.313	0.295	0.343	0.246	0.290
Plan-Solve	0.318	0.316	0.368	0.230	0.314

**Table 26: Light**

	gpt-4.1-mini	llama3.3-70b	gpt-4o-mini	phi4	amazon-nova-lite
Baseline	0.460	0.440	0.444	0.423	0.360
Emotion	0.461	0.439	0.450	0.428	0.347
Re-Reading	0.444	0.471*	0.431	0.432	0.379
Both-Inst	0.459	0.437	0.432 <sup>▽</sup>	0.417	0.364
Recency-Focused	0.478	0.467*	0.449	0.424	0.357
RolePlay-User	0.420 <sup>▽</sup>	0.461*	0.426 <sup>▽</sup>	0.391 <sup>▽</sup>	0.372
RolePlay-Expert	0.450	0.439	0.443	0.423	0.358
RolePlay-Frederick	0.451	0.435	0.430 <sup>▽</sup>	0.435	0.354
Summarize-Item	0.431 <sup>▽</sup>	0.459*	0.441	0.377 <sup>▽</sup>	0.355
Step-Back	0.480*	0.480*	0.465*	0.447	0.344
ReAct	0.479*	0.507*	0.378 <sup>▽</sup>	0.426	0.384
Rephrase	0.485*	0.492*	0.452	0.454*	0.391*
Echo	0.433 <sup>▽</sup>	0.481*	0.424 <sup>▽</sup>	0.457*	0.331 <sup>▽</sup>
Summarize-User	0.480*	0.467*	0.456	0.450*	0.358
Generate-Item	0.467	0.438	0.454	0.436	0.360
Reuse-Item	0.456	0.451	0.427 <sup>▽</sup>	0.403	0.337 <sup>▽</sup>
Explain	0.466	0.458*	0.467*	0.446*	0.366
Mock	0.455	0.442	0.413 <sup>▽</sup>	0.405	0.353
Step-by-Step	0.464	0.439	0.433 <sup>▽</sup>	0.420	0.352
Deep-Breath	0.463	0.437	0.431 <sup>▽</sup>	0.400	0.350
Plan-Solve	0.462	0.434	0.424 <sup>▽</sup>	0.406	0.374

**Table 27: Heavy**

	gpt-4.1-mini	llama3.3-70b	gpt-4o-mini	phi4	amazon-nova-lite
Baseline	0.456	0.439	0.421	0.411	0.418
Emotion	0.447	0.445	0.430	0.406	0.404
Re-Reading	0.412 <sup>▽</sup>	0.474*	0.374 <sup>▽</sup>	0.386 <sup>▽</sup>	0.408
Both-Inst	0.444	0.436	0.429	0.388 <sup>▽</sup>	0.429
Recency-Focused	0.455	0.473*	0.437	0.395	0.432
RolePlay-User	0.397 <sup>▽</sup>	0.432	0.396 <sup>▽</sup>	0.358 <sup>▽</sup>	0.411
RolePlay-Expert	0.446	0.441	0.418	0.406	0.408
RolePlay-Frederick	0.445 <sup>▽</sup>	0.438	0.423	0.401	0.403 <sup>▽</sup>
Summarize-Item	0.424 <sup>▽</sup>	0.432	0.414	0.366 <sup>▽</sup>	0.400
Step-Back	0.473	0.483*	0.454*	0.424	0.353 <sup>▽</sup>
ReAct	0.494*	0.483*	0.359 <sup>▽</sup>	0.400	0.355 <sup>▽</sup>
Rephrase	0.476*	0.443	0.439*	0.404	0.405
Echo	0.436 <sup>▽</sup>	0.473*	0.370 <sup>▽</sup>	0.383 <sup>▽</sup>	0.368 <sup>▽</sup>
Summarize-User	0.452	0.467*	0.449*	0.383 <sup>▽</sup>	0.386 <sup>▽</sup>
Generate-Item	0.455	0.438	0.436	0.376 <sup>▽</sup>	0.405
Reuse-Item	0.451	0.483*	0.428	0.360 <sup>▽</sup>	0.404
Explain	0.461	0.473*	0.453*	0.384 <sup>▽</sup>	0.392 <sup>▽</sup>
Mock	0.450	0.460*	0.402 <sup>▽</sup>	0.386 <sup>▽</sup>	0.393 <sup>▽</sup>
Step-by-Step	0.452	0.440	0.422	0.397	0.413
Deep-Breath	0.465	0.442	0.421	0.415	0.397 <sup>▽</sup>
Plan-Solve	0.450	0.450	0.421	0.398	0.342 <sup>▽</sup>

**Table 28: Hit@3 for Table 7 in the main paper.**

	Music	Movie	Groceries	Clothes	Book	Yelp	News	Food	Light	Heavy	4.1-mini	llama3.3	4o-mini	phi4	nova-lite
Baseline	0.720	0.595	0.525	0.520	0.675	0.680	0.380	0.445	0.568	0.568	0.568	0.543	0.537	0.514	0.497
Emotion	0.715	0.565 <sup>v</sup>	0.535	0.505	0.675	0.665	0.390	0.440	0.569	0.554	0.561	0.551	0.543	0.519	0.480 <sup>v</sup>
Re-Reading	0.690	0.605	0.520	0.500	0.630 <sup>v</sup>	0.545 <sup>v</sup>	0.390	0.400	0.555	0.515 <sup>v</sup>	0.535 <sup>v</sup>	0.586*	0.493 <sup>v</sup>	0.509	0.511
Both-Inst	0.715	0.595	0.550	0.505	0.645	0.710	0.340 <sup>v</sup>	0.425	0.570	0.551	0.561	0.540	0.524	0.497	0.505
Recency-Focused	0.695	0.590	0.555	0.525	0.665	0.680	0.360	0.470	0.580	0.555	0.568	0.570*	0.539	0.511	0.504
RolePlay-User	0.630 <sup>v</sup>	0.570	0.460 <sup>v</sup>	0.385 <sup>v</sup>	0.605 <sup>v</sup>	0.665	0.290 <sup>v</sup>	0.435	0.515 <sup>v</sup>	0.495 <sup>v</sup>	0.505 <sup>v</sup>	0.551	0.500 <sup>v</sup>	0.467 <sup>v</sup>	0.506
RolePlay-Expert	0.710	0.590	0.555	0.485	0.685	0.685	0.375	0.405	0.564	0.559	0.561	0.547	0.529	0.510	0.496
RolePlay-Frederick	0.700	0.600	0.545	0.505	0.655	0.670	0.380	0.405	0.569	0.546 <sup>v</sup>	0.557	0.542	0.524 <sup>v</sup>	0.517	0.489
Summarize-Item	0.635 <sup>v</sup>	0.595	0.505	0.535	0.650	0.645	0.340	0.405	0.537 <sup>v</sup>	0.540 <sup>v</sup>	0.539 <sup>v</sup>	0.547	0.524	0.471 <sup>v</sup>	0.486
Step-Back	0.755	0.665*	0.600*	0.560	0.695	0.690	0.350	0.440	0.596*	0.593*	0.594*	0.588*	0.564*	0.528	0.439 <sup>v</sup>
ReAct	0.730	0.680*	0.630*	0.550	0.705	0.690	0.355	0.445	0.586	0.610*	0.598*	0.604*	0.468 <sup>v</sup>	0.517	0.471 <sup>v</sup>
Rephrase	0.730	0.660*	0.565	0.550	0.705	0.695	0.360	0.455	0.589	0.591	0.590*	0.573*	0.547	0.529	0.504
Echo	0.695	0.640	0.505	0.490	0.640	0.585 <sup>v</sup>	0.390	0.380 <sup>v</sup>	0.536 <sup>v</sup>	0.545	0.541 <sup>v</sup>	0.589*	0.494 <sup>v</sup>	0.527	0.458 <sup>v</sup>
Summarize-User	0.750	0.625	0.585	0.505	0.650	0.675	0.375	0.410	0.584	0.560	0.572	0.573*	0.558*	0.520	0.468 <sup>v</sup>
Generate-Item	0.720	0.600	0.555	0.525	0.670	0.675	0.385	0.410 <sup>v</sup>	0.575	0.560	0.568	0.542	0.546	0.504	0.497
Reuse-Item	0.700	0.630	0.520	0.485	0.665	0.680	0.345	0.440	0.570	0.546	0.558	0.569*	0.525	0.478 <sup>v</sup>	0.472 <sup>v</sup>
Explain	0.720	0.605	0.575	0.520	0.660	0.695	0.375	0.430	0.574	0.571	0.573	0.577*	0.559*	0.514	0.487
Mock	0.725	0.575	0.540	0.530	0.685	0.665	0.355	0.420	0.564	0.560	0.562	0.552	0.497 <sup>v</sup>	0.494 <sup>v</sup>	0.477 <sup>v</sup>
Step-by-Step	0.715	0.590	0.510	0.545	0.670	0.670	0.380	0.435	0.573	0.556	0.564	0.542	0.525	0.508	0.487
Deep-Breath	0.730	0.600	0.530	0.545	0.690	0.685	0.400	0.420	0.575	0.575	0.575	0.544	0.526	0.504	0.478 <sup>v</sup>
Plan-Solve	0.705	0.585	0.530	0.535	0.660	0.680	0.380	0.425	0.568	0.557	0.562	0.550	0.525	0.501	0.454 <sup>v</sup>

**Table 29: Hit@3 for Table 13 in the main paper.**

	Baseline	Rephrase	Step-Back	ReAct
llama3.3-70b	0.537	0.650*	0.656*	0.644*
gpt-4.1-mini	0.581	0.662*	0.619	0.650*
claude-3.5-haiku	0.606	0.613	0.600	0.625
gpt-4.1	0.619	0.606	0.606	0.644
claude-3.7-sonnet	0.738	0.719	0.744	0.706
o3-mini	0.675	0.631	0.637	
o4-mini	0.656			
claude-3.7-sonnet (T)	0.637			
o3	0.725			

**Table 30: Percentages of outputs with containing five or fewer ranked item (left) and those containing nine or fewer ranked items (right).**

	gpt-4.1-mini	llama3.3-70b	gpt-4o-mini	phi4	amazon-nova-lite
Baseline	0/0.1	0/0.1	4.9/5.9	<b>12.9/13.5</b>	0.6/0.8
Emotion	0/0.1	0/0.1	4.1/4.9	<b>12.8/13.0</b>	0.2/0.4
Re-Reading	0/0.1	0/0.1	<b>15.6/18.0</b>	<b>27.8/28.3</b>	0/0.1
Both-Inst	0/0.1	0/0.1	2.8/3.6	<b>13.4/13.5</b>	0.6/0.8
Recency-Focused	0/0.1	0/0.1	6.2/6.9	<b>12.9/12.9</b>	0.5/0.6
RolePlay-User	0/0.1	0/0.1	0.4/0.8	<b>12.6/12.9</b>	0.1/0.2
RolePlay-Expert	0/0.1	0/0.1	4.5/5.4	<b>12.5/12.9</b>	1.9/2.0
RolePlay-Frederick	0/0.1	0/0.1	4.6/5.4	<b>13.4/13.6</b>	0.8/0.9
Summarize-Item	0/0.1	0/0.1	1.6/2.5	0/1.1	1.0/1.1
Step-Back	0/0.2	0/0.1	3.9/7.9	<b>14.5/15.6</b>	1.8/2.8
ReAct	0/0.9	0/0.1	<b>45.8/62.7</b>	<b>35.0/40.4</b>	<b>12.4/16.7</b>
Rephrase	0/0.9	0/0.1	4.6/5.7	<b>16.0/16.5</b>	1.5/1.8
Echo	0/0.1	0/0.1	<b>12.6/15.9</b>	<b>25.0/25.0</b>	0.1/0.2
Summarize-User	0/2.6	0/0.1	3.8/7.3	<b>13.3/13.8</b>	0.1/0.7
Generate-Item	0/0.1	0/0.1	3.5/5.3	<b>12.6/13.3</b>	1.0/1.1
Reuse-Item	0/0.1	0/0.1	<b>11.1/12.7</b>	<b>14.3/15.3</b>	<b>12.6/12.9</b>
Explain	0/0.2	0/0.1	1.4/6.2	<b>12.7/12.9</b>	0.5/0.9
Mock	0/0.1	0/0.1	<b>10.2/11.9</b>	<b>12.6/13.5</b>	1.6/1.9
Step-by-Step	0/0.1	0/0.1	5.2/6.5	<b>17.5/20.0</b>	0.5/0.6
Deep-Breath	0/0.1	0/0.1	5.6/6.8	<b>15.5/18.0</b>	0.1/0.2
Plan-Solve	0/0.1	0/0.1	7.4/9.3	<b>16.5/18.3</b>	6.2/8.1

**Table 31: Token Number Statistics (Baseline)**

	Light				Heavy			
	max	min	mean	std	max	min	mean	std
Yelp	9662	2959	6218.2	1199.7	48660	13363	24749.6	4674.3
News	1966	1227	1537.3	151.3	5516	2932	3890.8	495.5
Food	6218	3991	4859.1	353.2	18552	12373	15463.6	1117.0
Movie	3573	1121	2146.0	487.3	13282	3173	6793.2	1964.9
Music	3884	1038	1943.1	465.5	12315	2767	6579.7	2180.9
Groceries	4427	1668	2921.1	533.8	15994	5453	9414.0	2211.5
Clothes	4050	1340	2571.5	486.4	14379	3905	7168.7	1794.8
Book	4316	1004	1896.2	527.5	12014	3007	6477.8	1764.1

**Table 32: Token Number Statistics (Summarize-Item)**

	Light				Heavy			
	max	min	mean	std	max	min	mean	std
Yelp	7735	3464	4685.1	703.0	41732	12718	17863.0	3784.7
News	2670	1894	2285.6	176.9	7277	4776	5974.3	650.0
Food	5461	3716	4198.0	271.3	16199	10683	13388.6	1029.1
Movie	4807	2299	3050.8	435.3	16693	6427	9431.4	2106.2
Music	4936	2317	3048.5	452.6	16820	6124	9952.4	2374.8
Groceries	4160	2426	3064.5	398.0	16654	6605	9706.0	2059.7
Clothes	4328	2125	2751.8	384.5	14796	5884	8092.4	1588.4
Book	5287	2369	3104.1	517.2	16754	6367	9881.5	1995.0

```
ta1 = ### Observation:  
The user has a history of reviewing ballet and dance-related workout DVDs. They express a strong preference for challenging workouts that provide effective results, particularly in toning and defining muscles. The user has rated several products, with the highest rating being 5.0 for a challenging ballet workout, while others received lower ratings due to being too easy or not effective enough.  
  
### Thought:  
The user seems to prefer products that are intense and provide a good workout, particularly in ballet and dance fitness. They have explicitly mentioned the need for challenging workouts that yield visible results. Therefore, any candidate products that align with these preferences, especially in the fitness category, would likely be more appealing to the user.  
  
### Action:  
I will examine the candidate products to identify any that align with the user's preferences for challenging workouts, particularly in the exercise and fitness category.  
  
1. **Emmett Otter's Jug-Band Christmas** - Not related to fitness.  
(omitted)  
10. **Royal Paintbox** - Not related to fitness.  
  
### Observation:  
Among the candidate products, only **Tracy Anderson: Precision Toning** is related to fitness. However, based on the user's previous experiences with Tracy Anderson's workouts, they found them too easy. The other products do not align with the user's interests in ballet or challenging fitness routines.  
  
### Thought:  
Since there is only one candidate product related to fitness, and it does not align with the user's preference for challenging workouts, I must conclude that there are no suitable recommendations from the provided candidates.  
  
### Answer:  
Since none of the candidate products align with the user's preferences for challenging ballet or dance workouts, the final answer is: **[]** (no suitable recommendations).
```

**Figure 13: Incorrect output generated by the ReAct prompt**

**Table 33: Inference Cost for 1,600 users in Section 4**

	gpt-4.1-mini	llama3.3-70b	gpt-4o-mini	phi4	amazon-nova-lite
Baseline	4.25	7.73	1.61	1.71	0.70
Emotion	4.27	7.76	1.61	1.71	0.70
Re-Reading	8.44	15.40	3.19	4.93	1.40
Both-Inst	4.25	7.73	1.61	1.67	0.70
Recency-Focused	4.38	7.97	1.68	1.79	0.72
RolePlay-User	4.23	7.71	1.59	1.64	0.69
RolePlay-Expert	4.25	7.73	1.61	1.70	0.70
RolePlay-Frederick	4.26	7.73	1.61	1.71	0.70
Summarize-Item	4.48	8.11	1.69	1.20	0.70
Step-Back	10.66	16.82	3.77	3.13	2.87
ReAct	11.76	17.70	4.39	3.05	2.47
Rephrase	11.59	17.35	3.87	2.98	1.99
Echo	8.40	15.34	3.19	4.83	1.39
Summarize-User	10.55	16.74	3.92	2.81	1.75
Generate-Item	8.58	15.62	3.41	2.67	1.41
Reuse-Item	4.67	8.26	1.71	1.88	0.75
Explain	10.09	16.60	3.81	2.81	1.65
Mock	4.33	7.88	1.63	1.74	0.71
Step-by-Step	8.60	15.67	3.31	3.14	1.54
Deep-Breath	8.61	15.68	3.28	3.41	1.47
Plan-Solve	8.63	15.73	3.34	3.71	2.19
sum	149.28	257.27	55.83	54.21	27.21

**Table 34: Inference Cost for 160 users in Section 5.1**

	gpt-4.1-mini	llama3.3-70b	gpt-4o-mini	phi4	amazon-nova-lite
Baseline	0.43	0.78	0.33	0.17	0.14
Rephrase	1.15	1.74	0.38	0.28	0.20
Step-Back	1.08	1.69	0.38	0.32	0.30
ReAct	1.18	1.78	0.44	0.30	0.26
Rephrase→ReAct	1.87	2.88	0.75	0.46	0.38
Rephrase→Step-Back	1.72	2.82	0.66	0.44	0.36
SelfRefine (Rephrase)	1.73	2.77	0.62	0.41	0.32
SelfRefine (Step-Back)	1.73	2.68	0.60	0.48	0.38
SelfRefine (ReAct)	1.83	2.79	0.69	0.51	0.33
SelfConsistency (Rephrase)	1.75	2.74	0.69	0.42	0.33
SelfConsistency (Step-Back)	1.76	2.63	0.60	0.46	0.26
SelfConsistency (ReAct)	1.95	2.77	0.70	0.43	0.30
sum	18.17	28.06	6.84	4.69	3.54

**Table 35: Inference Cost for 160 users in Section 5.2**

	Baseline	ReAct	Rephrase	Step-Back	sum
phi4	0.17	0.30	0.28	0.32	1.07
amazon-nova-lite	0.14	0.26	0.20	0.30	0.90
gpt-4o-mini	0.33	0.44	0.38	0.38	1.53
llama3.3-70b	0.78	1.78	1.74	1.69	5.98
gpt-4.1-mini	0.43	1.18	1.15	1.08	3.83
claude-3.5-haiku	1.03	2.41	2.28	2.30	8.02
gpt-4.1	2.13	5.76	4.89	4.87	17.65
claude-3.7-sonnet	3.85	10.69	9.23	8.71	32.48
o3-mini	2.79	-	5.00	4.82	12.62
o4-mini	2.09	-	-	-	2.09
claude-3.7-sonnet (T)	7.72	-	-	-	7.72
o3	11.36	-	-	-	11.36