Large Language Models for Intent-Driven Session Recommendations

Zhu Sun
A*STAR Centre for Frontier AI Research; Singapore University of Technology and Design
Singapore, Singapore

Hongyang Liu*
Macquarie University
Sydney, Australia

Xinghua Qu†
Shanda Group AI Lab; Tianqiao and Chrissy Chen Institute
Singapore, Singapore

Kaidong Feng
Yanshan University
Qinghuangdao, China

Yan Wang
Macquarie University
Sydney, Australia

Yew Soon Ong
A*STAR Centre for Frontier AI Research; Nanyang Technological University
Singapore, Singapore

ABSTRACT

The goal of intent-aware session recommendation (ISR) approaches is to capture user intents within a session for accurate next-item prediction. However, the capability of these approaches is limited by assuming all sessions have a uniform and fixed number of intents. In reality, user sessions can vary, where the number of intentions may differ from one to another. Moreover, they can only learn user intents in the latent space, which further restricts the model's transparency. To ease these issues, we propose a simple yet effective paradigm for ISR motivated by the advanced reasoning capability of large language models (LLMs). Specifically, we first create an initial prompt to instruct LLMs to predict the next item by inferring varying user intents reflected in a session. Then, we propose an effective optimization mechanism to automatically optimize prompts with an iterative self-reflection. Finally, we leverage the robust generalizability of LLMs across diverse domains to efficiently select the optimal prompt for ISR. As such, the proposed paradigm effectively guides LLMs to identify varying user intents at a semantic level, thus delivering more accurate and comprehensible recommendations. Extensive experiments on three real-world datasets verify the superiority of our proposed method.

CCS CONCEPTS

• **Information systems** → **Recommender systems**; • **Computing methodologies** → *Neural networks.*

KEYWORDS

Session Recommendations, User Intents, Large Language Models

*Co-first authors
†Corresponding author: teddy.qu@shanda.com

SIGIR '24, July 14–18, 2024, Washington, DC, USA

ACM ISBN 979-8-4007-0431-4/24/07
https://doi.org/10.1145/3626772.3657688

ACM Reference Format:
Zhu Sun, Hongyang Liu, Xinghua Qu, Kaidong Feng, Yan Wang, and Yew Soon Ong. 2024. Large Language Models for Intent-Driven Session Recommendations. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '24), July 14–18, 2024, Washington, DC, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3626772.3657688

1 INTRODUCTION

Session-based recommendation (SR) [6, 16, 22, 65] aims to predict next items to be interacted based on short anonymous behavior sessions. Typically, different sessions may unveil diverse user intents [57]. Figure 1 shows two real sessions in Amazon Electronic dataset [37], where the first one in (a) reflects a single main purpose, i.e., shopping for laptop and accessories, while the second one in (b) is associated with two major intents, i.e., shopping for laptop protectors and camera accessories, respectively. As most public datasets do not include explicit intents of a session [57], intent-aware session recommendation (ISR) has emerged to capture the latent user intents in a session, thus enhancing the accuracy of SR.

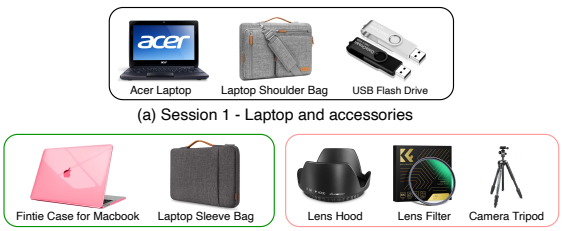

Figure 1: Examples of user sessions with various intents.

Early studies in ISR [29, 35, 56, 59] primarily constrain sessions to a single purpose or goal, such as in Figure 1(a). However, this simple assumption doesn't always hold in the real world, where a session may involve diverse items for various purposes, as in Figure 1(b). Consequently, various approaches have been developed to model multiple intentions in a session for more accurate prediction, e.g., IDSR [10], MCPRN [57], and NirGNN [26]. Despite the success, they suffer from two major limitations: (1) reliance on an unrealistic assumption that all sessions possess a consistent and

fixed number of intentions, treating it as a hyper-parameter; and (2) solely learning latent intentions within the embedding space, thereby impeding the transparency. Such limitations thus hinder these approaches for more accurate and comprehensible ISR.

Recently, the rise of large language models (LLMs) has opened up unprecedented opportunities in recommendation [4, 11, 18, 46], with SR not being an exception. Typically, LLMs are employed in two distinct ways, namely in-context learning (ICL) [24, 55] and parameter-efficient fine-tuning [3, 71]. However, LLMs cannot fully realize their potential through simple ICL, e.g., [55]. While fine-tuning LLMs holds promise, it grapples with challenges stemming from computational demands and the availability of open-source LLMs. Therefore, we propose a simple yet effective paradigm (abbreviated as LLM4ISR) to exploit the power of LLMs for more effective ISR from the perspective of delicate prompt designing.

Specifically, we first create an initial prompt via the Prompt Initialization module that guides LLMs to understand semantic user intents in a session, and predict the next item accordingly. Inspired by [41] in natural language processing (NLP), the Prompt Optimization module seeks to automatically optimize the initial prompt with an iterative self-reflection. The LLM is required to offer reasoning rooted in the identified errors to improve the initial prompt. Such improved prompts are then efficiently assessed with UCB bandits [1] to shortlist promising prompt candidates for an iterative optimization. Lastly, the Prompt Selection module prioritizes the selection of the optimal prompt by harnessing the robust generalizability of LLMs across diverse domains to maximize accuracy improvements. As such, our LLM4ISR can efficiently direct LLMs to infer and comprehend dynamic user intents at a semantic level, resulting in more accurate and understandable SR.

Our main contributions lie three-fold. **(1)** We introduce a simple yet powerful paradigm -LLM4ISR- to exploit the capability of LLMs for enhanced ISR through effective prompt designing. **(2)** Armed with prompt initialization, optimization, and selection modules, LLM4ISR empowers LLMs to comprehend varying user intents in a session semantically, for more accurate and comprehensible SR. **(3)** Experiments on three datasets showcase LLM4ISR significantly outperforms baselines with an average lift of 57.37% and 61.03% on HR and NDCG, respectively. Meanwhile, several insightful observations are gained, for example, (a) LLM4ISR yields promising accuracy with only a small number of training samples; (b) LLM4ISR exhibits advanced generalizability and excels in cross-domain scenarios; (c) LLM4ISR showcases superior strength on sparser datasets with shorter sessions; however, it might exhibit increased hallucination tendencies with sparser datasets; (d) the performance of LLM4ISR shows a positive correlation with the quality of the initial prompt, and lower-quality initial prompts tend to yield more significant improvements; and (e) a streamlined description and subtask division can enhance the quality of initial prompts.

2 RELATED WORK

Session-based Recommendation. Early works employ *conventional methods*, such as frequent sequential patterns [2, 67], session-level item-similarity (e.g., SKNN [25, 36]) and Markov chain (e.g., FPMC [45]). Later, *recurrent neural networks* (RNN) have been applied to handle longer sequences assuming adjacent items in a session are sequentially dependent. GRU4Rec [20] is the representative model, and has been further extended using, e.g., data augmentation [53, 60], new losses [19], parallel architecture with item features [21], and cross-session information transfer [44]. Subsequently, the *attention mechanism* has been adopted to relax this assumption by emphasizing more informative items in sessions [47], such as NARM [29] and STAMP [35][1]. To model the high-order transition among items, *graph neural network* (GNN) based methods have been recently designed to learn more accurate item embeddings from session graphs, such as SR-GNN [62], FGNN [43], GC-SAN [64], GSL4Rec [61], GNG-ODE [14], KMVG [7], and ADRL [8]. Besides, many studies propose to leverage both intra- and inter-session information, e.g., GCE-GNN [59], HG-GNN [39], DGNN [13], SPARE [40], CGSR [69], and HADCG [48]. Other works (e.g., S^2-DHCN [63] and CoHHN [73]) use hypergraphs for enhanced item representations in SR.

Intent-aware Session Recommendation. An essential line of research learns the intents hidden in the session for accurate SR. Early studies assume items inside a session are associated with one (implicit) purpose. In particular, NARM [29] and STAMP [35] use the attention mechanism to learn users' main intention in a session. SR-GNN [62], GC-SAN [64], GCE-GNN [59], TAGNN [70], LESSR [9] and MSGAT [42] model each session as graph-structured data and apply GNN to learn users' main intent. Nonetheless, a session may encompass items with varying intentions. Thus, solely modeling the main intent could lead to information loss, potentially hurting the performance of SR.

Hence, many studies endeavor to learn multiple intents for more effective SR. Specifically, NirGNN [26] learns dual intents via attention mechanisms and data distribution in session graphs. MCPRN [57] designs mixture-channel purpose routing networks to detect the purposes of each item in a session. IDSR [10] projects the item representation into multiple spaces indicating various intentions. HIDE [32] splits an item embedding into multiple chunks to represent various intentions. MIHSG [13] and Atten-Mixer [72] learn multi-granularity consecutive user intents for more accurate session representations. STAGE [31] and ISCON [38] capture the impact of multiple intrinsic intents for better SR. DAGNN [66] extracts session demands over the item category space to capture semantically correlated categories. However, they make an impractical assumption that all sessions have a uniform and fixed number of intentions. Moreover, most of them can only learn latent intents, thus restricting the transparency of SR. In contrast, we aim to leverage the advanced reasoning capabilities of LLMs to uncover varying numbers of semantic intents within a session for more accurate and comprehensible SR.

LLM-based Session Recommendation. LLMs have achieved remarkable achievements for general recommendation [4, 11, 18, 46]. To the best of our knowledge, NIR [55] is the only one that adopts zero-shot prompting for SR, and most methods target sequential recommendation [17]. Among them, some leverage the in-context learning (ICL) capability of LLMs, for instance, Hou et al. [24] use LLMs as rankers by designing sequential, recency-based and ICL

[1]Our study focuses on session-based recommendation. Therefore, some popular sequential-based recommendation approaches, e.g., Bert4Rec [49] and SASRec [27] are out of our scope. Please refer to [68] for the detailed difference between session- and sequential-based recommendation.

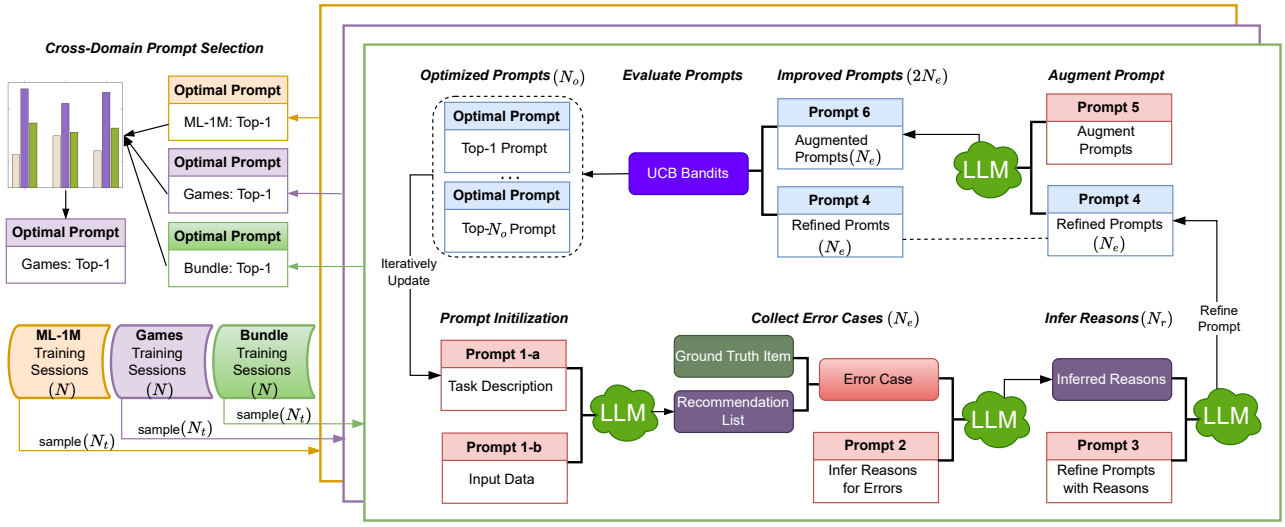

Figure 2: The overall architecture of LLM4ISR, composed of Prompt Initialization, Optimization, and Selection modules.

prompting. Others align LLMs for recommendation via parameter-efficient fine-tuning. To be specific, BIGRec [3] employ LLMs in an all-rank scenario by grounding LLMs to the recommendation and actual item spaces. TransRec [33] identifies fundamental steps of LLM-based recommendation to bridge the item and language spaces. GPT4Rec [30] generates multiple queries given item titles in a user's history and searches these queries to retrieve items for recommendation. LlamaRec [71] uses an ID-based sequential recommender to generate candidates and then designs a verbalizer to transform LLM output as the probability distribution for ranking.

However, the potential of LLMs cannot be exploited solely through simple ICL (e.g., [55]). While fine-tuning LLMs for recommendation shows promising results, it is constrained by the computational demands and availability of open-source LLMs. Instead, we introduce a new paradigm for ISR by automatically optimizing prompts, which efficiently guides LLMs to comprehend the varying semantic user intents in a session, for enhanced accuracy and transparency.

Efficient Prompt Designing for LLMs. Designing prompts efficiently constitutes a crucial aspect of LLM research. Most studies aim to enhance prompts by either differentiable tuning of soft prompts [28] or direct training of the prompt generator [58]. Another set of research focuses on improving prompts through discrete manipulations guided by reinforcement learning [12], often relying on auxiliary reward models. Within the discrete manipulation domain, some works use LLM-based feedback [15] but rely on task-specific local search over the prompt space without clear semantic direction. Our approach, inspired by [41], is versatile and applicable to various recommendation tasks beyond ISR, introducing meaningful semantic improvements to prompts.

3 THE PROPOSED METHOD

We now introduce LLM4ISR, a simple yet powerful framework inspired by the work [41] in the area of NLP. It is specifically designed to efficiently guide LLMs in comprehending varying user intents at a semantic level, to enhance the accuracy and transparency of SR.

Framework Overview. Figure 2 illustrates the overall architecture of our proposed framework, which is mainly composed of three modules. Firstly, Prompt Initialization is tasked with generating an initial prompt that directs LLMs in dynamically comprehending semantic user intents at the session level. Subsequently, Prompt Optimization aims to evaluate, refine, augment, and optimize the initial prompt through an iterative self-reflection (i.e., inferring reasons from the collected error cases). Lastly, Prompt Selection is designed to properly select the optimal prompt by exploiting the robust generalizability of LLMs across diverse domains, thus maximizing the accuracy enhancements of SR.

Prompt 1-a: ***Task Description***

Based on the user's current session interactions, you need to answer the following subtasks step by step:

1. *Discover combinations of items within the session, where the size of combinations can be one or more.*
2. *Based on the items within each combination, infer the user's interactive intent for each combination.*
3. *Select the intent from the inferred ones that best represent the user's current preferences.*
4. *Based on the selected intent, please rerank the items in the candidate set according to the possibility of potential user interactions and show me your ranking results with the item index.*

Note that the order of all items in the candidate set must be provided, and the items for ranking must be within the candidate set.

3.1 Prompt Initialization

Given a session, we first create an initial prompt for the task description. It seeks to guide LLMs in understanding the varying user intents at the semantic level in a session, which empowers LLMs to make more accurate and comprehensible SR. The task description is described in Prompt 1-a which divides the SR task into four subtasks by using the planning strategy [74]. Then, Prompt 1-a is used to guide ChatGPT[2] to predict the next item based on the historical (training) user sessions fed by Prompt 1-b. For ease of understanding, we take one training session as an example, that is,

[2]Without a further statement, it is based on GPT-3.5-turbo.

the current session interactions: [1."Zenana Women's Cami Sets", 2."Monster Tattoos", 3."I Love You This Much Funny T-rex Adult T-shirt", 4."Breaking Bad Men's Logo T-Shirt", 5."Sofia the First Sofia's Transforming Dress", 6."Lewis N. Clark 2-Pack Neon Leather Luggage Tag", 7."Russell Athletic Women's Stretch Capri", 8."US Traveler New Yorker 4 Piece Luggage Set Expandable", 9."Soffe Juniors Football Capri"]. The target (ground truth) item "It's You Babe Mini Cradle, Medium" is ranked at position 19 out of 20 items in the candidate set. By feeding Prompts 1-a and 1-b into ChatGPT, the target item holds the 16th position in the re-ranked candidate set.

Prompt 1-b: *Input Data*

Current session interactions: {[idx."item title", . . .]}
Candidate item set: {[idx."item title", . . .]}

3.2 Prompt Optimization

Prompt Optimization strives to evaluate, refine, augment, and optimize the initial task description with an iterative self-reflection. The detailed process is elaborated in what follows.

Collecting Error Cases. We randomly sample a batch (N_t) of sessions from training sessions (N) and guide ChatGPT to predict the next items with the initial prompts. Afterwards, we evaluate the recommendation outcomes, considering sessions where the target item ranks in the bottom half of the re-ranked candidate set as error cases. Following this rule, the example session mentioned in Section 3.1 is an error case, since the ranking position of the target item by using the initial prompt is 16 out of 20 candidates. These error cases, indicating that the current prompts cannot effectively guide LLMs in performing the SR task, will be utilized to further refine the prompts. We use N_e ($0 \leq N_e \leq N_t$) to denote the total number of such error cases for each batch.

Prompt 2: *Inferring Reasons for Errors*

I'm trying to write a zero-shot recommender prompt.
My current prompt is {prompt}.
But this prompt gets the following example wrong: {error_case}.
In this wrong example, in the re-ranked candidate set obtained using the current prompt, the ground truth item {ground_truth} should ideally be ranked first. However, this ground truth item is currently not placed at the top of the list.
Give {N_r} reasons why the prompt could have gotten this example wrong.
Wrap each reason with <START> and <END>.

Inferring Reasons. Understanding the reasons behind these error cases would greatly help refine the prompt, consequently enhancing recommendation performance. Thus, we leverage the self-reflection ability of ChatGPT, that is, asking ChatGPT to reconsider and offer justifications rooted in the identified errors. Inspired by [41], we adopt Prompt 2 to generate N_r reasons for each of the N_e error cases, where the '*error_case*' includes the user session and candidate item set. Below are the generated reasons for one error case.

- *One reason why the prompt could have gotten these examples wrong is that it assumes that the user's interactive intent can be accurately inferred based solely on the items within each combination. However, the prompt does not provide any information about the user's preferences, tastes, or previous interactions. Without this context, it is difficult to accurately infer the user's intent from the items alone.*
- *Another reason is that the prompt does not specify how the combinations of items within the session should be discovered. It assumes that the combinations are already known and provided as input. However, in real-world scenarios, discovering meaningful combinations of items from a user's session interactions can be a complex task. The prompt does not provide any guidance on how to perform this discovery process, which can lead to incorrect results.*

Prompt 3: *Refining Prompts with Reasons*

I'm trying to write a zero-shot recommender prompt.
My current prompt is {prompt}.
But this prompt gets the following example wrong: {error_case}.
Based on the example the problem with this prompt is that {reasons}.
Based on the above information, please write one improved prompt. The prompt is wrapped with <START> and <END>.
The new prompt is:

Refining Prompts. With the inferred N_r reasons for each error case, we now ask ChatGPT to refine the current prompt accordingly using Prompt 3. One example of the refined prompt is illustrated as Prompt 4. By comparing the initial task description (Prompt 1-a) and the refined Prompt 4, we can easily note that the initial prompt is improved by considering two aspects: (1) user preference and taste and (2) the definition of item combinations. These two aspects are exactly consistent with the inferred reasons by ChatGPT.

Prompt 4: *The Refined Prompt*

Given the user's current session interactions, you need to answer the following subtasks step by step:
1. *Identify any patterns or relationships between the items within the session.*
2. *Based on the identified patterns, infer the user's interactive intent within each combination of items.*
3. *Consider the user's preferences, tastes, or previous interactions to select the intent that best represents their current preferences.*
4. *Rerank the items in the candidate set according to the likelihood of potential user interactions. Provide the ranking results with the item index.*

Ensure that the order of all items in the candidate set is given, and the items for ranking are within the candidate set.

Augmenting Prompts. With the refined prompts, we further ask ChatGPT to augment prompts (Prompt 6 is an example of augmentation) with the same semantic meanings using Prompt 5. Accordingly, for the N_e error cases, we finally obtain $2N_e$ improved prompts through refinement and augmentation. These prompts will be further utilized for an iterative optimization based on their recommendation performance, as introduced in what follows.

Evaluating Prompts. From the pool of $2N_e$ prompts, our goal is to identify the most efficient ones with the best recommendation accuracy. One greedy way is to evaluate their performance with all historical user sessions. However, this may be quite computationally expensive. To improve the efficiency, we employ the upper confidence bound (UCB) Bandits [41] to efficiently estimate the performance of these prompts. In particular, it iteratively samples one prompt based on its estimated performance and then evaluates the prompt on a random batch of training sessions (N_t), and finally updates its performance based on the observed performance. The process is depicted by Algorithm 1, where the reward is calculated by NDCG measuring the ranking position of the target item; and γ

Algorithm 1: UCB-Bandits

Input: prompt set $\mathcal{P}$, training session set $\mathcal{S}$, sampled session size N_t, maximum epoch E_1, reward function $f(\cdot)$;
Output: the estimated performance $R[\mathcal{P}]$;

```
// Initialization
for each p_i ∈ P do
    R[p_i] = 0;            // initial estimated performance of p_i
    S[p_i] = 0;            // initial frequency of p_i being evaluated
for e_1 = 1; e_1 ≤ E_1; e_1 ++ do
    Sample p_i ← arg max_p ( R[p] + γ sqrt( log(e_1) / S[p] ) );
    Randomly sample S_t ⊂ S where |S_t| = N_t;
    r[p_i] = 0;            // the initial accumulated reward
    for each s ∈ S_t do
        r[p_i] ← r[p_i] + f(p_i, s);      // evaluate p_i with f(·)
    // Update evaluation frequency and performance
    S[p_i] ← S[p_i] + N_t;
    R[p_i] ← R[p_i] + r[p_i] / N_t;
return R[P];
```

Algorithm 2: Iterative-Optimization

Input: $\mathcal{S}, N_t, N_o, E_1, E_2, f(\cdot)$;
Output: the optimized prompt set $\mathcal{P}_o$;

```
E ← ∅, P_o ← p_init;
for e_2 = 1; e_2 ≤ E_2; e_2 ++ do
    P̃ ← ∅;
    for each p_i ∈ P_o do
        Randomly sample S_t ⊂ S where |S_t| = N_t;
        for each s ∈ S_t do
            if s is an error case then
                E ← E.append(s);          // collect error cases
        for each s ∈ E, |E| = N_e do
            Generate N_r reasons;                  // infer reasons
            p_ir ← p_i&N_r reasons;                // refine prompt
            p_ia ← p_ir;                           // augment prompt
            P̃.append(p_ir, p_ia);
    R[P̃] ← UCB-Bandits(P̃, S, N_t, E_1, f(·));   // evaluate prompts
    P_o ← Top-N_o of P̃ based on R[P̃];            // update prompts
return P_o;
```

is the exploration parameter. With UCB Bandits, we can efficiently obtain the estimated performance of the $2N_e$ prompts. Note that besides UCB Bandits, there are other alternative methods, such as ϵ-greedy, Thompson Sampling [5], and Contextual Bandits [54]. In our study, we select UCB Bandits due to its high efficiency and will explore other methods in future work.

Prompt 5: ***Augmenting Prompts***

Generate a variation of the following prompt while keeping the semantic meaning.
Input: {refined_prompt}.
Output:

Prompt 6: ***The Augmented Prompt***

Please follow these steps to answer the subtasks based on the user's current session interactions:

1. *Analyze the session items to find any patterns or relationships.*
2. *Use the identified patterns to determine the user's interactive intent for each combination of items.*
3. *Take into account the user's preferences, tastes, or previous interactions to choose the intent that best represents their current preferences.*
4. *Rank the items in the candidate set according to the likelihood of potential user interactions. Provide the ranking results along with the item index.*

Make sure to include all items in the candidate set and only rank items within the candidate set.

Prompt 7: ***The Optimized Prompt***

Please follow these steps to answer the given subtasks:

1. *Analyze the combinations of items in the user's session, considering any patterns or criteria.*
2. *Deduce the user's interactive intent within each combination, taking into account their previous interactions and preferences.*
3. *Determine the most representative intent from the inferred ones that aligns with the user's current preferences.*
4. *Reorder the items in the candidate set based on the selected intent, considering potential user interactions. Please provide the ranking results with item index.*

Remember to provide the order of all items in the candidate set and ensure that the items for ranking are within the candidate set. Take into consideration the relevance of the items in the current session interactions to the candidate set, and incorporate the user's preferences and history into the recommendations.

Iterative Optimization. Based on UCB Bandits, we then iteratively optimize the prompts [41]. According to the estimated performance R, we select the Top-N_o prompts and carry these promising prompts forward to the subsequent iteration. Specifically, we first use the Top-N_o prompts to replace the prompts in the previous iteration, and then repeat the series of tasks, i.e., collecting error cases, inferring reasons, refining, augmenting, and evaluating prompts, as described in Algorithm 2. This iterative loop fosters incremental enhancements and exploration among various promising prompt candidates. Prompt 7 is an example of an optimized prompt with the best estimated performance in the final iteration. After applying it on the example session mentioned in Section 3.1, the target item has been re-ranked to the 11th position, advancing eight positions from its original placement in the candidate set.

3.3 Prompt Selection

At the end of the iterative optimization, we have the Top-N_o prompts. Accordingly, one straightforward way is to choose the Top-1 prompt as the final selection due to its superior overall performance. However, we notice that although the majority of sessions achieve their peak accuracy with the Top-1 prompt, a subset of them displays the best performance with other Top prompts. This can be verified by Figure 3, which shows the performance gap between the Top-1 and Top-2 prompts on the validation sessions across three datasets in the domains of movie, games, and bundle (e-commerce)[3]. From the results, it's evident that there are data points positioned where the gap is less than zero, signifying that the Top-2 prompt surpasses the Top-1 prompt in some cases. Thus, one can come up with potential solutions: (1) ensemble all top prompts to get the compensation results; and (2) train a classifier to select the best top prompt for each session. Nevertheless, we empirically discovered that (1) fails to yield promising results because the enhancements in the subset do not offset the declines in the majority; while (2) heavily relies on the accuracy of the classifier, thus bringing in extra uncertainty to the final performance.

[3]The details of the datasets utilized in our study are introduced in Section 4.1.1.

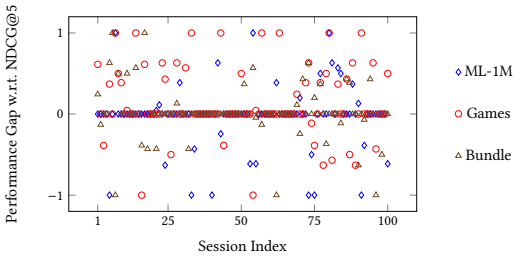

Figure 3: Performance of Top-1&-2 prompts (same domain).

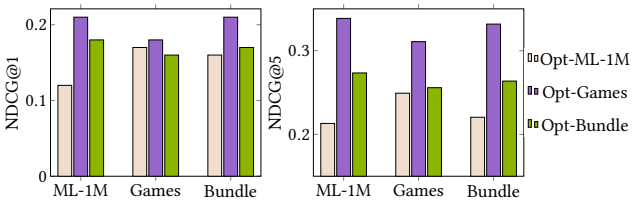

Figure 4: Performance of Top-1 prompt (cross-domain).

Fortunately, the robust generalizability of LLMs inspires us to explore the cross-domain performance of these optimized prompts. Specifically, Figure 4 depicts the performance of the Top-1 prompts from the three domains across the three datasets, for instance, 'Opt-Games' (in purple) refers to the Top-1 prompt in the games domain. From the figure, we can easily notice that Opt-Games consistently performs the best not only in its games domain but also in the other two domains. One possible explanation could be attributed to the Games dataset having the shortest average session length (refer to the 'Avg. Session Length' in Table 1) alongside a moderate level of sparsity (see 'Density Indicator' in Table 1). These factors collectively alleviate the challenge of identifying the optimal prompt to capture crucial and unique information from user sessions, thereby enhancing the overall performance of SR. Accordingly, we select the Opt-Games as the final prompt for all domains, the efficacy of which is verified in Section 4.2.

4 EXPERIMENTS AND RESULTS

We conduct extensive experiments to answer five research questions[4]. (**RQ1**) Does LLM4ISR outperform baselines? (**RQ2**) How do different components affect LLM4ISR? (**RQ3**) How do essential parameters affect LLM4ISR? (**RQ4**) How does LLM4ISR provide comprehensible SR? (**RQ5**) Is there any limitation of LLM4ISR?

4.1 Experimental Setup

4.1.1 Datasets. We use three real-world datasets from various domains. In particular, MovieLens-1M (ML-1M)[5] contains users' ratings of movies. Games is one subcategory from the Amazon dataset [37], containing users' ratings towards various video games. Bundle [51, 52] contains session data for three subcategories (Electronic, Clothing, and Food) of Amazon, where the intents for each session are explicitly annotated via crowdsourcing workers. Table 1 shows the statistics of the datasets. For ML-1M and Games, we

[4]Our code and data are available at https://github.com/hyllll/LLM4ISR
[5]https://grouplens.org/datasets/movielens/

Table 1: Statistics of datasets. 'Density Indicator' refers to the average frequency of each item appearing in the dataset, calculated as (#Sessions × Avg. Session Length)/#Items.

	#Items	#Sessions	Avg. Session Length	Density Indicator
ML-1M	3,416	784,860	6.85	1573.86
Games	17,389	100,018	4.18	24.04
Bundle	14,240	2,376	6.73	1.12

chronologically order the rated items into a sequence for each user and then divide it into sessions by day. For each dataset, we split the sessions into training, validation, and test sets with a ratio of 8:1:1, i.e., 80% of the early sessions as the training sets; the subsequent 10% as the validation set; and the final 10% as the test set.

4.1.2 Baselines. We compare LLM4ISR with 11 baselines in three types. *The first type is the conventional methods.* **Mostpop** recommends the most popular items; **SKNN** [25] recommends session-level similar items; and **FPMC** [45] is the matrix factorization method with the first-order Markov chain. *The second type is the deep learning-based methods, which can be classified into single-intent, multi-intent and cross-domain based ones.* **NARM** [29] is an RNN-based model with the attention mechanism to capture the main purpose from the hidden states; **STAMP** [35] learns a user's main intent by emphasizing the effect of the last item in the context; **GCE-GNN** [59] uses both local and global graphs to learn item representation thus obtaining the main intent of the session; **MCPRN** [57] models users' multiple purposes to get the final session representation; **HIDE** [32] splits the item embedding into multiple chunks, with each chunk representing a specific intention to learn diverse intentions; **Atten-Mixer** [72] learns multi-granularity consecutive user intents for more accurate session representations; and **UniSRec** [23] is a representative cross-domain model using item descriptions to learn transferable representations across different domains. *The last type is the LLM-based method.* **NIR** [55] adopts zero-shot prompting for the next item recommendation.

4.1.3 Parameter Settings. For all methods, the maximum training epoch is 100 with the early stop mechanism, and the candidate size is 20 following NIR [55]. We use Optuna (optuna.org) to find out the optimal hyperparameters of all methods with 50 trails [50]. The search space for batch size, item embedding size, and learning rate are $\{64, 128, 256\}$, $\{32, 64, 128\}$ and $\{10^{-4}, 10^{-3}, 10^{-2}\}$, respectively. For SKNN, K is searched from $\{50, 100, 150\}$. For NARM, the hidden size and layers are searched in $\{50, 100, 150, 200\}$ and $\{1, 2, 3\}$, respectively. For GCE-GNN, the number of hops, and the dropout rate for global and local aggregators are respectively searched in $\{1, 2\}$, $[0, 0.2, 0.4, 0.6, 0.8]$ and $\{0, 0.5\}$. For MCPRN, τ and the number of purpose channels are separately searched in $\{0.01, 0.1, 1, 10\}$ and $\{1, 2, 3, 4\}$. For HIDE, the number of factors is searched in $\{1, 3, 5, 7, 9\}$; the regularization and balance weights are searched in $\{10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}\}$; the window size is searched in $[1, 10]$ stepped by 1; and the sparsity coefficient is set as 0.4. For Atten-Mixer, the intent level L and the number of attention heads are respectively searched in $[1, 10]$ stepped by 1 and in $\{1, 2, 4, 8\}$. For UniSRec, we follow the original paper and use same settings for model pre-training and fine-tuning. For LLM4ISR, $N = 50, N_t = 32, N_r = 2, N_o = 5, E_1 = 16, E_2 = 2$; and we randomly select 8

Table 2: Performance comparison on all datasets, where the best and runner-up results are highlighted in bold and marked by '*'; '-' means a very small value; and 'Improve' indicates the relative improvements comparing the best and runner-up results.

Data	Metrics	Conventional			Single-Intent			Multi-Intent			Cross-Domain	LLMs		Improve	p-value
		MostPop	SKNN	FPMC	NARM	STAMP	GCE-GNN	MCPRN	HIDE	Atten-Mixer	UniSRec	NIR	LLM4ISR		
ML-1M	HR@1	0.0004	0.1270	0.1132	0.1692*	0.1584	0.1312	0.1434	0.1498	0.1490	0.0508	0.0572	**0.2000**	18.20%	$4.3e^{-3}$
	HR@5	0.0070	0.3600	0.3748	0.5230*	0.5078	0.4748	0.4788	0.4998	0.4932	0.2508	0.2326	**0.5510**	5.35%	$5.9e^{-2}$
	NDCG@1	0.0004	0.1270	0.1132	0.1692*	0.1584	0.1312	0.1434	0.1498	0.1490	0.0508	0.0572	**0.2000**	18.20%	$4.3e^{-3}$
	NDCG@5	0.0053	0.2530	0.2464	0.3501*	0.3367	0.3044	0.3157	0.3256	0.3216	0.1459	0.1436	**0.3810**	8.83%	$3.5e^{-3}$
Games	HR@1	–	0.0020	0.0498	0.0572	0.0556	0.0692	0.0522	0.0696	0.0530	0.0544	0.1168*	**0.2588**	121.58%	$6.4e^{-5}$
	HR@5	–	0.0020	0.2564	0.2574	0.2586	0.2744	0.2416	0.2694	0.2472	0.2512	0.3406*	**0.5866**	72.23%	$3.5e^{-5}$
	NDCG@1	–	0.0020	0.0498	0.0572	0.0556	0.0692	0.0522	0.0696	0.0530	0.0544	0.1168*	**0.2588**	121.58%	$6.4e^{-5}$
	NDCG@5	–	0.0020	0.1508	0.1534	0.1555	0.1701	0.1432	0.1662	0.1475	0.1482	0.2310*	**0.4313**	86.71%	$7.3e^{-6}$
Bundle	HR@1	–	–	0.0398	0.0322	0.0365	0.0360	0.0360	0.0458	0.0525	0.0496	0.0975*	**0.1697**	74.05%	$2.0e^{-5}$
	HR@5	0.0042	–	0.2475	0.2322	0.2352	0.2237	0.2352	0.2585	0.2644	0.2402	0.2832*	**0.4328**	52.82%	$2.6e^{-4}$
	NDCG@1	–	–	0.0398	0.0322	0.0365	0.0360	0.0360	0.0458	0.0525	0.0496	0.0975*	**0.1697**	74.05%	$2.0e^{-5}$
	NDCG@5	0.0021	–	0.1395	0.1303	0.1339	0.1267	0.1490	0.1495	0.1549	0.1430	0.1939*	**0.3040**	56.78%	$3.4e^{-5}$

Table 3: The results of ablation study across all datasets on the test set (seed 0).

	ML-1M					Games				Bundle				
	Initial	Top-1	EnSame	EnCross	LLM4ISR	Initial	EnSame	EnCross	LLM4ISR	Initial	Top-1	EnSame	EnCross	LLM4ISR
HR@1	0.1430	0.2070	0.1120	0.1070	**0.2110**	0.0790	0.1540	0.1090	**0.2600**	0.0504	0.1176	0.0294	0.0840	**0.1933**
HR@5	0.4150	0.5130	0.4130	0.4250	**0.5730**	0.3510	0.5250	0.4410	**0.5960**	0.2437	0.3193	0.1891	0.2689	**0.4454**
NDCG@1	0.1430	0.2070	0.1120	0.1070	**0.2110**	0.0790	0.1540	0.1090	**0.2600**	0.0504	0.1176	0.0294	0.0840	**0.1933**
NDCG@5	0.2823	0.3662	0.2693	0.2640	**0.3975**	0.2110	0.3574	0.2779	**0.4381**	0.1396	0.2202	0.1119	0.1745	**0.3183**

prompts as the input of the UCB Bandits in each iteration. The best parameter settings of baselines are reported on our GitHub Repo.

Regarding the training set, LLM4ISR uses 50 randomly sampled sessions ($N = 50$) to optimize prompts in each domain and considers the performance from the three domains for optimal prompt selection. For UniSRec, the training set from the target domain is used for model fine-tuning while those from the two source domains are used for model pre-training. For a fair comparison, the rest baselines are trained with 150 sessions (a superset of LLM4ISR's training set) in each domain, as it is non-trivial to use cross-domain data to train these models. Following [29, 59, 72], we adopt the data augmentation strategy for all non-LLM baselines, that is, a session s with size $|s|$ will be expanded to $(|s| - 1)$ training samples. To manage the API call cost, for ML-1M and Games, we randomly sample 1000 sessions from the validation set and test set, respectively.

4.1.4 Evaluation Metrics. Following state-of-the-arts [55, 57, 68], HR@K and NDCG@K are adopted as the evaluation metrics. We set $K = \{1, 5\}$ since most users tend to prioritize the quality of items appearing at the top positions in real scenarios [34]. Generally, higher metric values indicate better ranking results. For robust performance, we repeat the test procedure five times where each time we set different seed values (e.g., 0, 10, 42, 625, and 2023) to generate different candidate sets. Finally, we report the average results as presented in Table 2.

4.2 Results and Analysis

4.2.1 Overall Comparison (RQ1). Table 2 shows the performance of all methods on the three datasets. Several findings are noted.

(1) Regarding conventional methods (CMs), the model-based FPMC performs the best on sparser datasets Games and Bundles, while is slightly defeated by SKNN on the denser dataset ML-1M (see 'Density Indicator' in Table 1). This exhibits the strong capability of model-based methods in learning sequential patterns with sparse data. **(2)** For single-intent methods (SIMs), each gains its best performance on different datasets, e.g., NARM excels on ML-1M, whereas GCE-GNN wins on Games. Generally, GCE-GNN showcases advantages on shorter sessions with sparser datasets (e.g., Games). This is because such sessions do not contain sufficient local information, thus requiring global information to compensate. Conversely, NARM performs well on longer sessions with denser datasets (e.g., ML-1M), since such sessions possess substantial information (e.g., frequent patterns) making the fusion of global information potentially noisy. **(3)** Concerning multi-intent methods (MIMs), MCPRN displays the poorest performance across all datasets, as limited training data hinders the learning of proper parameters for multiple RNN channels. Besides, HIDE exceeds Atten-Mixer on relatively denser datasets (ML-1M and Games) but lags on the sparsest Bundle. **(4)** In terms of the cross-domain method (CDM), UniSRec tends to be outperformed by either SIMs or MIMs on all datasets, even CMs (e.g., SKNN and FPMC) on ML-1M. This could be attributed to the substantial number of parameters it possesses compared with other baselines. Despite being trained on augmented data from three domains, it appears to suffer from under-training due to a lack of sufficient data. **(5)** Among LLM-based methods (LLMMs), the fact that LLM4ISR outperforms NIR confirms our assertion that basic ICL fails to fully exploit LLMs and emphasizes the superiority of our prompt designing paradigm.

Overall, CMs underperform SIMs or MIMs, underscoring the importance of capturing user intents for improved SR. The victory of MIMs over SIMs on Bundle validates the effectiveness of multi-intent learning. However, some SIMs (e.g., NARM) surpass MIMs (e.g., MCPRN) on ML-1M and Games, suggesting that fixed numbers of intents might constrain the capability of MIMs. CDMs typically exhibit inferior performance compared to SIMs and MIMs, primarily because of their dependency on extensive training data. This suggests a limitation in the generability of traditional CDMs. In contrast, LLMMs exhibit strength on sparser datasets compared to denser ones (e.g., NIR on Games vs. ML-1M), and shorter sessions compared to longer ones (e.g., NIR on Games vs. Bundles). Similar

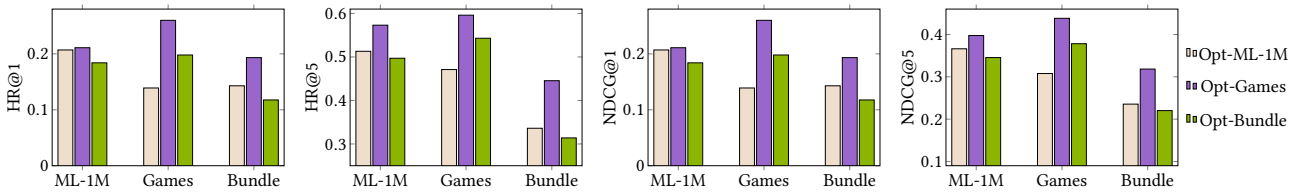

Figure 5: Performance of Top-1 prompt from each domain in the cross-domain scenario on the test set (seed 0).

trends are also observed in LLM4ISR, where the average improvement (12.65%) on ML-1M is smaller than that (100.55%) on Games. This demonstrates the advanced ability of LLMs to address the data sparsity issue. Lastly, our LLM4ISR consistently achieves the best performance among all baselines, with an average improvement of 57.37% and 61.03% on HR and NDCG, respectively.

4.2.2 Ablation Study (RQ2). To verify the efficacy of different components, we compare LLM4ISR with its four variants. In particular, 'Initial' means we only use the initial prompt without optimization for each domain; 'Top-1' means we merely adopt the Top-1 prompt within each domain; 'EnSame' means we ensemble the ranking results of both Top-1 and -2 prompts within each domain; 'EnCross' means we ensemble the ranking results of Top-1 prompts across all domains. The performance is shown in Table 3, where three observations are noted: (1) Initial performs worse than Top-1, which indicates the effectiveness of iterative prompt optimization; (2) both EnSame and EnCross underperform Top-1, implying that simply ensembling the top prompts either within the same domain or across different domains cannot efficiently improve the performance; and (3) LLM4ISR with the Top-1 prompt in the games domain consistently achieves the best performance across all datasets, showcasing the efficacy of our cross-domain prompt selection strategy. This is further confirmed by Figure 5, presenting the performance of Top-1 prompt from each domain in the cross-domain scenario.

4.2.3 Parameter Analysis (RQ3). We investigate the impact of vital hyperparameters. First, we check the impact of different initial prompts. Thus, we substitute the task description Prompt (1-a) with Prompts (1-c) and (1-d). In particular, Prompt (1-c) does minor changes on Prompt (1-a), e.g., changes 'combination' to 'group', and adds the definition for the 'group'. Prompt (1-d) involves two major changes: (1) using 'preferences' rather than 'intentions', although they share the same meaning in ISR; and (2) simplifying the four subtasks into two subtasks. The results are presented in Table 4 (rows 1-9), where the performance ranking of directly using the initial prompts is (1-c) < (1-a)< (1-d); meanwhile, the corresponding Top-1 prompts optimized based on these initial prompts hold the same performance ranking. This indicates that (1) simplified descriptions and subtasks division can improve the quality of the initial prompt, *vice versa*; (2) the quality of the initial prompts positively affects the final performance; (3) the lower-quality initial prompt yields larger overall improvements; (4) regardless of the quality of initial prompts, they can be largely enhanced with iterative optimization; and (5) the performance of LLM4ISR presented in Table 2 is not the upper bound, and can be further improved with better initial prompts, showcasing its great potential.

Table 4: Results of parameter analysis on Bundle (seed 0).

	HR@1	HR@5	NDCG@1	NDCG@5
Prompt 1-a (Initial)	0.0504	0.2437	0.0504	0.1396
Top-1 (N_t = 32)	0.1176	0.3193	0.1176	0.2202
Improve	133.33%	31.02%	133.33%	57.74%
Prompt 1-c (Initial)	0.0420	0.1639	0.0420	0.0985
Top-1 (N_t = 32)	0.1050	0.2479	0.1050	0.1689
Improve	150.00%	51.25%	150.00%	71.47%
Prompt 1-d (Initial)	0.1008	0.2479	0.1008	0.1706
Top-1 (N_t = 32)	0.1807	0.3739	0.1807	0.2744
Improve	79.27%	50.83%	79.27%	60.84%
Top-1 (N_t = 32, +GT)	0.0798	0.2647	0.0798	0.1725
Improve (vs. Prompt 1-a)	58.33%	8.62%	58.33%	23.57%
Top-1 (N_t = 16)	0.1008	0.2857	0.1008	0.1860
Improve (vs. Prompt 1-a)	100%	17.23%	100%	33.24%

Prompt 1-c: *Task Description*

Based on the user's current session interactions, answer the following subtasks step by step:

1. *Group items within the current session, where each item group should reflect a different pattern or theme.*
2. *Based on the inferred item groups, infer the user's intent within each group.*
3. *Select intents from the inferred ones that best represent the user's current preferences.*
4. *Rerank the 20 items in the candidate set based on their relevance to the selected intents, prioritizing higher relevance items to rank higher.*

Note that the order of all items in the candidate set must be given, and the items for ranking must be within the candidate set.

Prompt 1-d: *Task Description*

Based on the user's current session interactions, you need to answer the following tasks:

1. *Please infer the user's preferences, considering that the user may have one or multiple preferences.*
2. *Based on inferred preferences, please rerank the items in the candidate set according to the possibility of potential user interactions and show me your ranking results with the item index.*

Note that the order of all items in the candidate set must be provided, and the items for ranking must be within the candidate set.

Besides, we notice that providing ground truth information in 'Inferring Reasons' (see Prompt 2) cannot improve the recommendation accuracy (row 2 vs. row 10). The possible reason is that the ground truth information has been utilized in 'Collecting Error Cases', and repeatedly using it may hinder the self-reflection ability of LLMs. Therefore, in the optimization process, we omit the ground truth information in Prompt 2. Lastly, we further study the impact of batch size N_t on the final performance by varying its values in {16, 32}, and the results are presented in Table 4 (row 2 vs. row 12). Accordingly, we find that $N_t = 32$ is the optimal setting.

4.2.4 Visualization (RQ4). To illustrate the results generated by LLM4ISR, we randomly sample one test session from Bundle as

Figure 6: The case study on Bundle.

shown in Figure 6. Given the historical session, LLM4ISR first detects and analyzes the underlying user intents, providing the following response: *based on the user's current session interactions, it seems that the user is interested in tea and canned food products. The user has interacted with tea-related items such as 'Williamson London Skyline Tea Caddy' (i_1) and 'Williamson English Breakfast Elephant Tea 40 Teabags' (i_4). They have also interacted with canned food items like 'Worthington Super Links, 19-Ounce Cans' (i_2), and 'Loma Linda Tender Bits, 19-Ounce Cans' (i_3).* Then, based on the detected intents, it further justifies how to re-rank the candidate items: *considering the user's genre preference for tea and canned food, we can infer that the user might be interested in similar items from the candidate set. Therefore, we can rearrange the candidate set to prioritize tea and canned food items. Note: the rearranged list is based on the inferred intent of the user and the relevance of the items to the user's session interactions.* Finally, it furnishes a re-ranked recommendation list, positioning the ground truth item "Williamson English Breakfast Elephant Tea Caddy" (highlighted in a red-dot circle) from its initial 16th rank within the candidate set to the first position. In summary, the case study validates that LLM4ISR can help provide more accurate and comprehensible recommendations.

4.2.5 Discussion on Hallucination (RQ5). Despite the effectiveness of LLM4ISR, it showcases limitations due to the inherent issue of LLMs, that is, LLM4ISR may generate hallucination for some sessions (e.g., the response does not contain the ranking list or the ground truth item is not included in the ranking list), although we add hard constraints in the prompt such as "*the order of all items in the candidate set must be provided, and the items for ranking must be within the candidate set*". Table 5 illustrates the ratio of sessions having hallucinations on the test sets using the Top-1 prompt from each domain across the three datasets. Two major observations can be noted. **(1)** The ratio is at its lowest on ML-1M but peaks on Bundle (highlighted in blue). This discrepancy likely stems from ML-1M having the highest average frequency of items across sessions, whereas Bundle exhibits the lowest trend (see Table 1). The frequent appearance of items in various sessions may simplify the pattern mining process, thus reducing task complexity to some extent. Additionally, Bundle's diverse range of electronic, clothing, and food products elevates the complexity of the SR task compared

Table 5: Ratio of sessions with hallucination (seed 0).

	Opt-ML-1M	Opt-Games	Opt-Bundle	Average
ML-1M	0.30%	0.10%	0.30%	0.23%
Games	8.30%	6.80%	6.00%	7.03%
Bundle	7.56%	9.66%	10.92%	9.38%
Average	5.39%	5.52%	5.74%	5.55%

Table 6: Performance with and without JSON on Bundle.

	Opt-ML-1M		Opt-Games		Opt-Bundle	
	+JSON	-JSON	+JSON	-JSON	+JSON	-JSON
HR@1	0.0546	0.1429	0.0504	0.1933	0.0756	0.1176
HR@5	0.2227	0.3361	0.2185	0.4454	0.2395	0.3193
NDCG@1	0.0546	0.1429	0.0504	0.1933	0.0756	0.1176
NDCG@5	0.1336	0.2355	0.1289	0.3183	0.1531	0.2202
Ratio	7.14%	7.56%	8.82%	9.66%	9.66%	10.92%

to the more focused ML-1M dataset. **(2)** The optimal prompts from different domains show comparable performance as highlighted in pink. On average, there are around 5.55% sessions with hallucination, implying that further performance enhancements can be obtained by addressing this issue and exhibiting the latent potential of LLMs for SR (note that HR and NDCG values are set as 0 for sessions with hallucination in all results).

To this end, we involve JSON mode[6] in the response to better control the output by adding the constraint: *Provide the ranking results for the candidate set using JSON format, following this format without deviation: ["Item ID": "correspond item index", "Item Title": "correspond Item Title"].* Table 6 illustrates the performance contrast of LLM4ISR on Bundle, with and without JSON mode in the response. The results indicate that employing JSON mode marginally reduces the hallucination issue, yet significantly compromises recommendation accuracy. Contrarily, we find most such cases can be resolved using GPT-4 with improved accuracy.

5 CONCLUSION

Inspired by the reasoning capability of LLMs, we introduce a new paradigm – LLM4ISR – for intent-aware session recommendation. It aims to discover varying numbers of semantic intents hidden in different sessions for more accurate and comprehensible recommendations via delicate prompt designing. Specifically, the Prompt Initialization module creates the initial prompt to instruct LLMs to predict the next item by inferring varying intents reflected in a session. Then, the Prompt Optimization module is devised to optimize prompts with iterative self-reflection in an automatic manner. Finally, the Prompt Selection module seeks to appropriately select the optimal prompt based on the robust generalizability of LLMs across diverse domains. Experiments on real-world datasets show the superiority of LLM4ISR against other counterparts. Furthermore, several insightful discoveries are made to guide subsequent studies in this area.

ACKNOWLEDGMENTS

This work was supported by ARC Discovery Projects DP200101441 and DP230100676, A*Star Center for Frontier Artificial Intelligence Research, and in part by the Data Science and Artificial Intelligence Research Centre, School of Computer Science and Engineering at the Nanyang Technological University (NTU), Singapore.

[6] https://platform.openai.com/docs/guides/text-generation/json-mode

REFERENCES

[1] Jean-Yves Audibert, Sébastien Bubeck, and Rémi Munos. 2010. Best arm identification in multi-armed bandits.. In *Proceedings of the 23rd Annual Conference on Learning Theory (COLT)*. 41–53.

[2] Jay Ayres, Jason Flannick, Johannes Gehrke, and Tomi Yiu. 2002. Sequential pattern mining using a bitmap representation. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. 429–435.

[3] Keqin Bao, Jizhi Zhang, Wenjie Wang, Yang Zhang, Zhengyi Yang, Yancheng Luo, Fuli Feng, Xiangnaan He, and Qi Tian. 2023. A bi-step grounding paradigm for large language models in recommendation systems. *arXiv preprint arXiv:2308.08434* (2023).

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

[5] Olivier Chapelle and Lihong Li. 2011. An empirical evaluation of thompson sampling. *Proceedings of the 25th Annual Conference on Neural Information Processing Systems 2011 (NIPS)* (2011), 2249–2257.

[6] Jingfan Chen, Guanghui Zhu, Haojun Hou, Chunfeng Yuan, and Yihua Huang. 2022. AutoGSR: Neural architecture search for graph-based session recommendation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. 1694–1704.

[7] Qian Chen, Zhiqiang Guo, Jianjun Li, and Guohui Li. 2023. Knowledge-enhanced Multi-View Graph Neural Networks for Session-based Recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. 352–361.

[8] Qian Chen, Jianjun Li, Zhiqiang Guo, Guohui Li, and Zhiying Deng. 2023. Attribute-enhanced dual channel representation learning for session-based recommendation. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management (CIKM)*. 3793–3797.

[9] Tianwen Chen and Raymond Chi-Wing Wong. 2020. Handling information loss of graph neural networks for session-based recommendation. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. 1172–1180.

[10] Wanyu Chen, Pengjie Ren, Fei Cai, Fei Sun, and Maarten de Rijke. 2020. Improving end-to-end sequential recommendations with intent-aware diversification. In *Proceedings of the 29th ACM International Conference on Information and Knowledge Management (CIKM)*. 175–184.

[11] Sunhao Dai, Ninglu Shao, Haiyuan Zhao, Weijie Yu, Zihua Si, Chen Xu, Zhongxiang Sun, Xiao Zhang, and Jun Xu. 2023. Uncovering chatgpt's capabilities in recommender systems. In *Proceedings of the 17th ACM Conference on Recommender Systems (RecSys)*. 1126–1132.

[12] Mingkai Deng, Jianyu Wang, Cheng-Ping Hsieh, Yihan Wang, Han Guo, Tianmin Shu, Meng Song, Eric P Xing, and Zhiting Hu. 2022. Rlprompt: Optimizing discrete text prompts with reinforcement learning. *arXiv preprint arXiv:2205.12548* (2022).

[13] Jiayan Guo, Yaming Yang, Xiangchen Song, Yuan Zhang, Yujing Wang, Jing Bai, and Yan Zhang. 2022. Learning multi-granularity consecutive user intent unit for session-based recommendation. In *Proceedings of the 15th ACM International Conference on Web Search and Data Mining (WSDM)*. 343–352.

[14] Jiayan Guo, Peiyan Zhang, Chaozhuo Li, Xing Xie, Yan Zhang, and Sunghun Kim. 2022. Evolutionary preference learning via graph nested gru ode for session-based recommendation. In *Proceedings of the 31st ACM International Conference on Information and Knowledge Management (CIKM)*. 624–634.

[15] Yiduo Guo, Yaobo Liang, Chenfei Wu, Wenshan Wu, Dongyan Zhao, and Nan Duan. 2023. Learning to program with natural language. *arXiv preprint arXiv:2304.10464* (2023).

[16] Qilong Han, Chi Zhang, Rui Chen, Riwei Lai, Hongtao Song, and Li Li. 2022. Multi-faceted global item relation learning for session-based recommendation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. 1705–1715.

[17] Jesse Harte, Wouter Zorgdrager, Panos Louridas, Asterios Katsifodimos, Dietmar Jannach, and Marios Fragkoulis. 2023. Leveraging large language models for sequential recommendation. In *Proceedings of the 17th ACM Conference on Recommender Systems (RecSys)*. 1096–1102.

[18] Zhankui He, Zhouhang Xie, Rahul Jha, Harald Steck, Dawen Liang, Yesu Feng, Bodhisattwa Majumder, Nathan Kallus, and Julian Mcauley. 2023. Large language models as zero-shot conversational recommenders. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management (CIKM)*. Just Accepted.

[19] Balázs Hidasi and Alexandros Karatzoglou. 2018. Recurrent neural networks with top-k gains for session-based recommendations. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management (CIKM)*. 843–852.

[20] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2016. Session-based recommendations with recurrent neural networks. In *Proceedings of the 4th International Conference on Learning Representations (ICLR)*.

[21] Balázs Hidasi, Massimo Quadrana, Alexandros Karatzoglou, and Domonkos Tikk. 2016. Parallel recurrent neural network architectures for feature-rich session-based recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems (RecSys)*. 241–248.

[22] Yupeng Hou, Binbin Hu, Zhiqiang Zhang, and Wayne Xin Zhao. 2022. Core: simple and effective session-based recommendation within consistent representation space. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. 1796–1801.

[23] Yupeng Hou, Shanlei Mu, Wayne Xin Zhao, Yaliang Li, Bolin Ding, and Ji-Rong Wen. 2022. Towards universal sequence representation learning for recommender systems. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*. 585–593.

[24] Yupeng Hou, Junjie Zhang, Zihan Lin, Hongyu Lu, Ruobing Xie, Julian McAuley, and Wayne Xin Zhao. 2023. Large language models are zero-shot rankers for recommender systems. *arXiv preprint arXiv:2305.08845* (2023).

[25] Dietmar Jannach and Malte Ludewig. 2017. When recurrent neural networks meet the neighborhood for session-based recommendation. In *Proceedings of the 11th ACM Conference on Recommender Systems (RecSys)*. 306–310.

[26] Di Jin, Luzhi Wang, Yizhen Zheng, Guojie Song, Fei Jiang, Xiang Li, Wei Lin, and Shirui Pan. 2023. Dual intent enhanced graph neural network for session-based new item recommendation. In *Proceedings of the ACM Web Conference (TheWebConf)*. 684–693.

[27] Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive sequential recommendation. In *2018 IEEE International Conference on Data Mining (ICDM)*. 197–206.

[28] Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691* (2021).

[29] Jing Li, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tao Lian, and Jun Ma. 2017. Neural attentive session-based recommendation. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management (CIKM)*. 1419–1428.

[30] Jinming Li, Wentao Zhang, Tian Wang, Guanglei Xiong, Alan Lu, and Gerard Medioni. 2023. GPT4Rec: A generative framework for personalized recommendation and user interests interpretation. *arXiv preprint arXiv:2304.03879* (2023).

[31] Yinfeng Li, Chen Gao, Xiaoyi Du, Huazhou Wei, Hengliang Luo, Depeng Jin, and Yong Li. 2022. Spatiotemporal-aware Session-based Recommendation with Graph Neural Networks. In *Proceedings of the 31st ACM International Conference on Information and Knowledge Management (CIKM)*. 1209–1218.

[32] Yinfeng Li, Chen Gao, Hengliang Luo, Depeng Jin, and Yong Li. 2022. Enhancing hypergraph neural networks with intent disentanglement for session-based recommendation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. 1997–2002.

[33] Xinyu Lin, Wenjie Wang, Yongqi Li, Fuli Feng, See-Kiong Ng, and Tat-Seng Chua. 2023. A Multi-facet Paradigm to Bridge Large Language Model and Recommendation. *arXiv preprint arXiv:2310.06491* (2023).

[34] Hongyang Liu, Zhu Sun, Xinghua Qu, and Fuyong Yuan. 2021. Top-aware recommender distillation with deep reinforcement learning. *Information Sciences (INS)* 576 (2021), 642–657.

[35] Qiao Liu, Yifu Zeng, Refuoe Mokhosi, and Haibin Zhang. 2018. STAMP: short-term attention/memory priority model for session-based recommendation. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. 1831–1839.

[36] Malte Ludewig and Dietmar Jannach. 2018. Evaluation of session-based recommendation algorithms. *User Modeling and User-Adapted Interaction (UMUAI)* 28 (2018), 331–390.

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

[38] Sejoon Oh, Ankur Bhardwaj, Jongseok Han, Sungchul Kim, Ryan A Rossi, and Srijan Kumar. 2022. Implicit session contexts for next-item recommendations. In *Proceedings of the 31st ACM International Conference on Information and Knowledge Management (CIKM)*. 4364–4368.

[39] Yitong Pang, Lingfei Wu, Qi Shen, Yiming Zhang, Zhihua Wei, Fangli Xu, Ethan Chang, Bo Long, and Jian Pei. 2022. Heterogeneous global graph neural networks for personalized session-based recommendation. In *Proceedings of the 15th ACM International Conference on Web Search and Data Mining (WSDM)*. 775–783.

[40] Andreas Peintner, Amir Reza Mohammadi, and Eva Zangerle. 2023. SPARE: Shortest path global item relations for efficient session-based recommendation. In *Proceedings of the 17th ACM Conference on Recommender Systems (RecSys)*. 58–69.

[41] Reid Pryzant, Dan Iter, Jerry Li, Yin Tat Lee, Chenguang Zhu, and Michael Zeng. 2023. Automatic prompt optimization with "gradient descent" and beam search. *arXiv preprint arXiv:2305.03495* (2023).

[42] Shutong Qiao, Wei Zhou, Junhao Wen, Hongyu Zhang, and Min Gao. 2023. Bi-channel multiple sparse graph attention networks for session-based recommendation. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management (CIKM)*. 2075–2084.

[43] Ruihong Qiu, Jingjing Li, Zi Huang, and Hongzhi Yin. 2019. Rethinking the item order in session-based recommendation with graph neural networks. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management (CIKM)*. 579–588.

[44] Massimo Quadrana, Alexandros Karatzoglou, Balázs Hidasi, and Paolo Cremonesi. 2017. Personalizing session-based recommendations with hierarchical recurrent neural networks. In *Proceedings of the 11th ACM Conference on Recommender Systems (RecSys)*. 130–137.

[45] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing personalized markov chains for next-basket recommendation. In *Proceedings of the 19th International Conference on World Wide Web (WWW)*. 811–820.

[46] Scott Sanner, Krisztian Balog, Filip Radlinski, Ben Wedin, and Lucas Dixon. 2023. Large language models are competitive near cold-start recommenders for language-and item-based preferences. In *Proceedings of the 17th ACM Conference on Recommender Systems (RecSys)*. 890–896.

[47] Wenzhuo Song, Shoujin Wang, Yan Wang, Kunpeng Liu, Xueyan Liu, and Minghao Yin. 2023. A counterfactual collaborative session-based recommender system. In *Proceedings of the ACM Web Conference (TheWebConf)*. 971–982.

[48] Jiajie Su, Chaochao Chen, Weiming Liu, Fei Wu, Xiaolin Zheng, and Haoming Lyu. 2023. Enhancing hierarchy-aware graph networks with deep dual clustering for session-based recommendation. In *Proceedings of the ACM Web Conference (TheWebConf)*. 165–176.

[49] 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 *Proceedings of the 28th ACM International Conference on Information and Knowledge Management (CIKM)*. 1441–1450.

[50] Zhu Sun, Hui Fang, Jie Yang, Xinghua Qu, Hongyang Liu, Di Yu, Yew-Soon Ong, and Jie Zhang. 2022. Daisyrec 2.0: Benchmarking recommendation for rigorous evaluation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* (2022).

[51] Zhu Sun, Kaidong Feng, Jie Yang, Hui Fang, Xinghua Qu, Yew-Soon Ong, and Wenyuan Liu. 2024. Revisiting bundle recommendation for intent-aware product bundling. *ACM Transactions on Recommender Systems (TORS)* (2024).

[52] Zhu Sun, Jie Yang, Kaidong Feng, Hui Fang, Xinghua Qu, and Yew Soon Ong. 2022. Revisiting bundle recommendation: datasets, tasks, challenges and opportunities for intent-aware product bundling. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. 2900–2911.

[53] Yong Kiam Tan, Xinxing Xu, and Yong Liu. 2016. Improved recurrent neural networks for session-based recommendations. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*. 17–22.

[54] Joannes Vermorel and Mehryar Mohri. 2005. Multi-armed bandit algorithms and empirical evaluation. In *Proceedings of the 16th European Conference on Machine Learning (ECML)*. Springer, 437–448.

[55] Lei Wang and Ee-Peng Lim. 2023. Zero-Shot next-item recommendation using large pretrained language models. *arXiv preprint arXiv:2304.03153* (2023).

[56] Shoujin Wang, Liang Hu, and Longbing Cao. 2017. Perceiving the next choice with comprehensive transaction embeddings for online recommendation. In *Proceedings of the 2017 Machine Learning and Knowledge Discovery in Databases: European Conference (ECML/PKDD)*. Springer, 285–302.

[57] Shoujin Wang, Liang Hu, Yan Wang, Quan Z Sheng, Mehmet Orgun, and Longbing Cao. 2019. Modeling multi-purpose sessions for next-item recommendations via mixture-channel purpose routing networks. In *Proceedings of International Joint Conference on Artificial Intelligence (IJCAI)*.

[58] Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2022. Self-instruct: Aligning language model with self generated instructions. *arXiv preprint arXiv:2212.10560* (2022).

[59] Ziyang Wang, Wei Wei, Gao Cong, Xiao-Li Li, Xian-Ling Mao, and Minghui Qiu. 2020. Global context enhanced graph neural networks for session-based recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. 169–178.

[60] Zhidan Wang, Wenwen Ye, Xu Chen, Wenqiang Zhang, Zhenlei Wang, Lixin Zou, and Weidong Liu. 2022. Generative session-based recommendation. In *Proceedings of the ACM Web Conference (TheWebConf)*. 2227–2235.

[61] Chunyu Wei, Bing Bai, Kun Bai, and Fei Wang. 2022. Gsl4rec: Session-based recommendations with collective graph structure learning and next interaction prediction. In *Proceedings of the ACM Web Conference (TheWebConf)*. 2120–2130.

[62] Shu Wu, Yuyuan Tang, Yanqiao Zhu, Liang Wang, Xing Xie, and Tieniu Tan. 2019. Session-based recommendation with graph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, Vol. 33. 346–353.

[63] Xin Xia, Hongzhi Yin, Junliang Yu, Qinyong Wang, Lizhen Cui, and Xiangliang Zhang. 2021. Self-supervised hypergraph convolutional networks for session-based recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, Vol. 35. 4503–4511.

[64] Chengfeng Xu, Pengpeng Zhao, Yanchi Liu, Victor S Sheng, Jiajie Xu, Fuzhen Zhuang, Junhua Fang, and Xiaofang Zhou. 2019. Graph contextualized self-attention network for session-based recommendation. In *Proceedings of International Joint Conference on Artificial Intelligence (IJCAI)*, Vol. 19. 3940–3946.

[65] Heeyoon Yang, YunSeok Choi, Gahyung Kim, and Jee-Hyong Lee. 2023. LOAM: Improving Long-tail Session-based Recommendation via Niche Walk Augmentation and Tail Session Mixup. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. 527–536.

[66] Liqi Yang, Linhan Luo, Lifeng Xin, Xiaofeng Zhang, and Xinni Zhang. 2022. DAGNN: Demand-aware graph neural networks for session-based recommendation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*.

[67] Ghim-Eng Yap, Xiao-Li Li, and Philip S Yu. 2012. Effective next-items recommendation via personalized sequential pattern mining. In *Proceedings of the 17th International Conference on Database Systems for Advanced Applications (DASFAA)*. 48–64.

[68] Qing Yin, Hui Fang, Zhu Sun, and Yew-Soon Ong. 2023. Understanding diversity in session-based recommendation. *ACM Transactions on Information Systems (TOIS)* 42, 1 (2023), 1–34.

[69] Dianer Yu, Qian Li, Hongzhi Yin, and Guandong Xu. 2023. Causality-guided graph learning for session-based recommendation. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management (CIKM)*. 3083–3093.

[70] Feng Yu, Yanqiao Zhu, Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. 2020. TAGNN: Target attentive graph neural networks for session-based recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. 1921–1924.

[71] 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. In *The 1st Workshop on Personalized Generative AI @CIKM (PGAI)*.

[72] Peiyan Zhang, Jiayan Guo, Chaozhuo Li, Yueqi Xie, Jae Boum Kim, Yan Zhang, Xing Xie, Haohan Wang, and Sunghun Kim. 2023. Efficiently leveraging multi-level user intent for session-based recommendation via atten-mixer network. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining (WSDM)*. 168–176.

[73] Xiaokun Zhang, Bo Xu, Liang Yang, Chenliang Li, Fenglong Ma, Haifeng Liu, and Hongfei Lin. 2022. Price does matter! modeling price and interest preferences in session-based recommendation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. 1684–1693.

[74] Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, et al. 2023. Least-to-most prompting enables complex reasoning in large language models. In *Proceedings of the 17th International Conference on Learning Representations (ICLR)*.