LLM4DSR: Leveraging Large Language Model for Denoising Sequential Recommendation

BOHAO WANG, Zhejiang University, Hangzhou, China
FENG LIU and CHANGWANG ZHANG, OPPO Research Institute, Shenzhen, China
JIAWEI CHEN, State Key Laboratory of Blockchain and Data Security, Zhejiang University, Hangzhou, China and Hangzhou High-Tech Zone (Binjiang) Institute of Blockchain and Data Security, Hangzhou, China
YUDI WU and SHENG ZHOU, Zhejiang University, Hangzhou, China
XINGYU LOU and JUN WANG, OPPO Research Institute, Shenzhen, China
YAN FENG, CHUN CHEN, and CAN WANG, Zhejiang University, Hangzhou, China

Sequential recommenders generate recommendations based on users' historical interaction sequences. However, in practice, these sequences are often contaminated by noisy interactions, which can arise from various factors such as clickbait, the influence of prominently positioned items, or accidental interactions. Such noise can significantly degrade recommendation performance. Accurately identifying such noisy interactions without additional information is particularly challenging due to the absence of explicit supervisory signals indicating noise. Large Language Models (LLMs), equipped with extensive open knowledge and semantic reasoning abilities, offer a promising avenue to bridge this information gap. However, employing LLMs for denoising in sequential recommendation presents notable challenges: (1) Direct application of pretrained LLMs may not be competent for the denoising task, frequently generating nonsensical responses; (2) Fine-tuning on the denoising task can partially mitigate the issue of generating nonsensical outputs. However, even after fine-tuning, the reliability of LLM outputs remains questionable, especially given the complexity of the denoising task and the inherent hallucination issue of LLMs.

To tackle these challenges, we propose LLM4DSR, a tailored approach for denoising sequential recommendation using LLMs. We constructed a self-supervised fine-tuning task to activate LLMs' capabilities to identify noisy items and suggest replacements. Furthermore, we developed an uncertainty estimation module that ensures only high-confidence responses are utilized for sequence corrections. Remarkably, LLM4DSR is model-agnostic, allowing corrected sequences to be flexibly applied across various recommendation models. To the best of our knowledge, this is the first work that employs LLMs for sequential recommendation denoising while addressing the unique challenges of adapting LLMs to this task. Extensive experiments conducted on three

This work is supported by the National Natural Science Foundation of China (Grant Nos. 62372399, 62476244), OPPO Research Fund, and the advanced computing resources provided by the Supercomputing Center of Hangzhou City University.
Authors' Contact Information: Bohao Wang, Zhejiang University, Hangzhou, China; e-mail: bohao.wang@zju.edu.cn; Feng Liu, OPPO Research Institute, Shenzhen, China; e-mail: liufeng4hit@gmail.com; Changwang Zhang, OPPO Research Institute, Shenzhen, China; e-mail: changwangzhang@foxmail.com; Jiawei Chen (corresponding author), State Key Laboratory of Blockchain and Data Security, Zhejiang University, Hangzhou, China and Hangzhou High-Tech Zone (Binjiang) Institute of Blockchain and Data Security, Hangzhou, China; e-mail: sleepyhunt@zju.edu.cn; Yudi Wu, Zhejiang University, Hangzhou, China; e-mail: wuyudi@zju.edu.cn; Sheng Zhou, Zhejiang University, Hangzhou, China; e-mail: zhousheng_zju@zju.edu.cn; Xingyu Lou, OPPO Research Institute, Shenzhen, China; e-mail: louxingyu@oppo.com; Jun Wang, OPPO Research Institute, Shenzhen, China; e-mail: junwang.lu@gmail.com; Yan Feng, Zhejiang University, Hangzhou, China; e-mail: fengyan@zju.edu.cn; Chun Chen, Zhejiang University, Hangzhou, China; e-mail: chenc@cs.zju.edu.cn; Can Wang, Zhejiang University, Hangzhou, China; e-mail: wcan@zju.edu.cn.

ACM 1558-2868/2025/10-ART6
https://doi.org/10.1145/3762182

real-world datasets across two noise settings validate the effectiveness of LLM4DSR, demonstrating an average improvement of 12.9% in NDCG@20. The code is available at https://github.com/WANGBohaO-jpg/LLM4DSR.

CCS Concepts: • **Information systems** → **Recommender systems**;

Additional Key Words and Phrases: Sequential Recommendation, Denoise, Large Language Model

ACM Reference format:
Bohao Wang, Feng Liu, Changwang Zhang, Jiawei Chen, Yudi Wu, Sheng Zhou, Xingyu Lou, Jun Wang, Yan Feng, Chun Chen, and Can Wang. 2025. LLM4DSR: Leveraging Large Language Model for Denoising Sequential Recommendation. *ACM Trans. Inf. Syst.* 44, 1, Article 6 (October 2025), 32 pages.
https://doi.org/10.1145/3762182

1 Introduction

Large Language Models (LLMs) have demonstrated remarkable capabilities in content comprehension [7], generation [43], and semantic reasoning [92], thereby driving significant progress in **Natural Language Processing (NLP)** and related fields [1]. Recently, LLMs have been extensively applied in the field of **Recommender Systems (RSs)** [19, 20, 48, 97]. Some researchers have directly utilized LLMs by prompting or fine-tuning them to act as specialized recommenders [4, 32, 53]. Meanwhile, others have employed LLMs as auxiliary tools to enhance traditional recommendation models, e.g., serving as encoders for user/item features [94], supplementary knowledge bases [99], advanced recommendation explainers [75], and conversational agents [27].

Given the powerful capabilities of LLMs, this work explores an innovative question: *Can LLMs effectively function as denoisers for sequential recommendation?* Sequential recommendation critically relies on the accuracy of users' historical interaction sequences to predict subsequent items. However, in practice, these sequences are often contaminated by noise due to various factors—e.g., being attracted by clickbait [85], influenced by item prominent positions [12], or from accidental interactions [58]. Such noise can significantly mislead recommendation models and degrade their performance. Figure 1 presents empirical evidence of this impact. We assess the model's performance on datasets with varying ratios of artificially introduced noise. Specifically, to incorporate a predefined level of noise into the real-world dataset, each item in the sequence is replaced with a randomly selected item with a certain probability. The details of this noise-setting process are detailed in Section 4.1.1. Even with a relatively small noise proportion (e.g., 20%), we observe substantial performance drops of 41% and 36% in SASRec [42] and LLaRA [53], respectively. These results underscore the critical importance of denoising in recommendation systems.

Despite its importance, the task of denoising is inherently challenging. A primary issue is the absence of labels for noisy data, resulting in a lack of clear signals regarding the nature of the noise. Recent approaches to denoising in sequential recommendation either rely on human-designed heuristics [72, 108, 109], which require extensive expertise and often lack precision; or are trained jointly with recommendation models guided by the recommendation objective [10, 58, 107, 108], which may still be heavily influenced by the noisy data, particularly since the labels of the recommendation objective (i.e., the next item) might also be contaminated. *Thus, this work explores a new avenue of leveraging LLMs for recommendation denoising, which can potentially compensate for this knowledge gap.* Leveraging their extensive open knowledge and advanced semantic reasoning capabilities [7], LLMs can infer user preferences by analyzing the attributes of interacted items and applying semantic reasoning. This enables them to identify noisy items that do not accurately reflect users' true preferences. For example, if a user's viewing history primarily consists of comedy films, an occasional interaction with a horror film might be attributable to an accidental click. LLMs

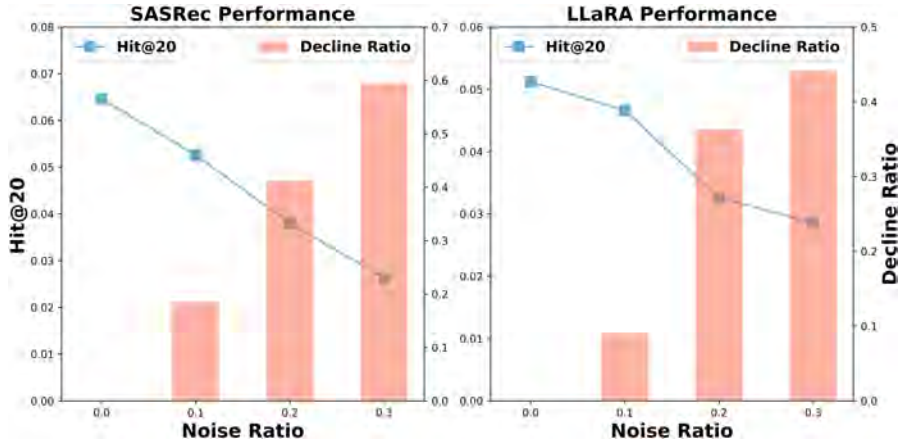

Fig. 1. Performance of SASRec [42] (a traditional sequential recommendation model) and LLaRA [53] (a sequential recommendation model based on LLMs) under different artificial noise ratios and the proportion of performance degradation relative to the artificial noise-free condition. Both models exhibit significant performance degradation as the noise ratio increases. The experiments are conducted on the Amazon Game dataset, with dataset details provided in Section 4.1.1.

can detect such anomalies based on their understanding of item semantics and common-sense knowledge. Moreover, by leveraging the generative capabilities of LLMs, they can not only identify noisy items but also potentially suggest replacements that align more closely with user preferences.

However, employing pretrained LLMs for denoising sequential recommendation also presents significant challenges:

— *Challenge 1: Direct application of pretrained LLMs may not be competent for denoising tasks.* A common approach to utilizing LLMs involves structuring the denoising task as a prompt and directly feeding it into the LLMs. However, since LLMs are primarily trained for tasks such as commonsense reasoning, language understanding, response generation, coding, and mathematics [1, 22], the significant gap between the pretraining objectives of LLMs and the specific requirements of sequential recommendation denoising can make such direct application problematic. In our preliminary experiments with the pretrained LLaMA3 8B model [22], we observe that a significant portion (over 40%) of the model's responses are nonsensical. Figure 2 presents typical failure cases, where the LLM frequently misclassifies all items in a sequence as noise.

— *Challenge 2: Responses from LLMs might not always be reliable.* Although fine-tuning can partially mitigate meaningless outputs, LLMs remain susceptible to the notorious phenomenon of *hallucination* [41], where they might incorrectly label non-existent items as noise or suggest non-existent replacements. Moreover, given the complexity of the recommendation denoising task, the accuracy of an LLM-based denoiser still has an upper-bound. Blindly trusting LLMs for corrections may not be advisable.

To address these challenges, we propose LLM4DSR, a tailored approach for denoising sequential recommendations using LLMs. To overcome the first challenge, we adapt pretrained LLMs to the denoising task through a self-supervised fine-tuning process. We construct an instruction dataset by replacing a proportion of items in sequences with randomly selected alternatives, which are then used for fine-tuning LLMs to identify noisy items and suggest appropriate replacements.

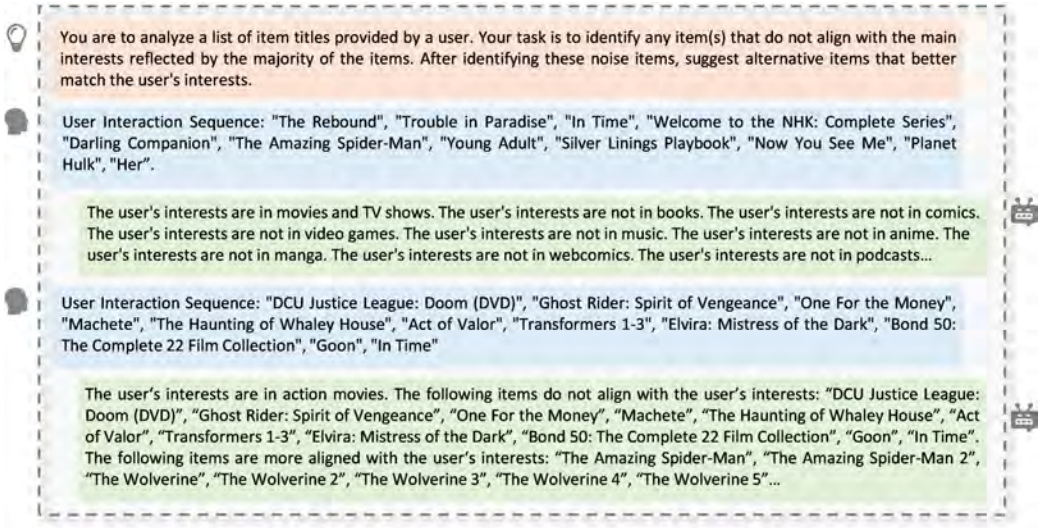

Fig. 2. Failure cases of using the un-finetuned LLaMA3 8B model for denoising tasks. The problems in the output include meaningless output, classifying all items as noise, generating fake items, and so on. Fifty test data are used to generate responses with LLaMA3, and the reasonableness of these responses is evaluated using GPT-4. Over 40% of the responses are deemed nonsensical.

This approach activates the potential denoising ability of LLMs. To tackle the second challenge, we introduce an uncertainty estimation module to assess the reliability of the identified noisy items, ensuring that only high-confidence judgments are used to correct sequences for subsequent recommendations. Notably, LLM4DSR is data-centric and model-agnostic, which can be considered a data preprocessing technique (cf. Section 2.2 for discussions). Our extensive experiments demonstrate that LLM4DSR significantly enhances the performance of both traditional sequential recommendation models and advanced LLM-based models.

The primary contributions of this work are:

—Introducing the novel application of LLMs in denoising sequential recommendations and identifying key challenges.
—Proposing LLM4DSR, a novel LLM-based recommendation denoising method that utilizes self-supervised fine-tuning and an uncertainty estimation mechanism to fully exploit the capabilities of LLMs.
—Conducting comprehensive experiments that demonstrate the superiority of LLM4DSR over state-of-the-art denoising methods across three recommendation backbone models.

The remainder of this article is organized as follows: Section 2 outlines the basic definitions of the tasks and methods involved. The proposed method, LLM4DSR, is introduced in Section 3. Section 4 presents our experimental results and subsequent discussions. Related work is reviewed in Section 5. Finally, we conclude the article and suggest future research directions in Section 6.

2 Preliminary

In this section, we present the background of the task of sequential recommendation denoising and the LLMs. Table 1 summarizes the notations involved in this article.

Table 1. Notations in This Article

Notations	Descriptions
v_i	The ith item in the item set $\mathcal{V}$
S	The historical temporal interaction sequence
S'	The noiseless sequence of S generated by the model
$\hat{S}$	The modified sequence of S generated by randomly replacing
s_t	The tth item in the sequence S
y_t	Whether the interaction s_t is noise predicted by the model
r_t	The replacement item generated by the model for s_t
$text(s_t)$	The textual description of the item s_t
x	The token sequence of the input part of the prompt
z	The token sequence of the out part of the prompt
z_{noise}	The token sequence of the token sequence of the noise item given by the LLM
Φ	The model parameter
T_i	The token sequence of the textual description of the item s_i
η	The probability threshold to classify items as noise
α	Artificial noise ratio of real-world dataset with artificial noise
b	The maximum number of items modified in a single sequence
q_{ui}	The ranking position of the item i for the user u.

2.1 Background of Sequential Recommendation

Our problem definitions directly refer to recent work [2, 42, 71]. Given a sequential RS, let $\mathcal{V} = \{v_1, v_2, \cdots, v_{|\mathcal{V}|}\}$ be the set of items. We use $S^i = \left[s_1^i, \cdots, s_t^i, \cdots, s_{|S^i|}^i\right]$ to denote ith historical collected interaction sequence where $s_t^i \in \mathcal{V}$ denotes the tth item interacted within sequence S^i, and $|S^i|$ represents the sequence's length. We denote a subsequence $[s_j^i, \cdots, s_k^i]$ of S^i as $S_{j:k}^i$. For each sequence S^i, sequential recommendation takes $S_{1:t}^i$ sequence data as input and outputs the item that the user is likely to interact with next, which is expected as S_{t+1}^i. For the sake of brevity, we will use S to replace S^i in the remainder of the article.

2.2 Sequential Recommendation Denoising

The presence of noisy interactions in user behavior sequences can impact model performance. As shown in Figure 1, both traditional sequential recommendation models and recent LLM-based recommendation models exhibit a significant decline in performance as the noise ratio increases. Therefore, developing denoising techniques for sequential recommendation is an important task.

Recent work on sequential recommendation denoising can be mainly categorized into two types: (1) the explicit denoising and (2) the implicit denoising. For a given sequence S, the objective of the explicit denoising strategies [58, 72, 108] target at identifying and removing the noise within the sequence and ultimately generate a noiseless sequence S'. Implicit denoising methods [10, 31, 107, 115] do not directly remove items from the sequence. Instead, they reduce the impact of noise with respect to the final representation by decreasing the weight of noise items, or by filtering noise at the representation level through filters.

Traditional denoising methods encounter significant challenges due to the absence of labeled noise data. These methods often rely on heuristic prior knowledge for noise identification, which requires extensive expertise and frequently lacks precision [72, 108, 109]. Alternatively, some denoising approaches employ recommendation loss as a supervisory signal during the denoising

process [10, 58, 107, 108]. However, this can be adversely affected by the presence of noise. Moreover, explicit denoising methods risk information loss from sequences due to potential erroneous deletions, which can degrade model performance.

To address these limitations, this work explores leveraging LLMs in sequential recommendation denoising. The advantages of this strategy are multi-faceted: (1) LLMs possess extensive open knowledge and advanced semantic reasoning abilities, which can potentially bridge the information gap in the denoising task; (2) The denoising task can be decoupled from the learning of recommendation models, serving as a data preprocessing step, thus allowing the corrected sequence to benefit various sequential recommendation models; (3) Owing to the generative ability of LLMs, the method can not only identify noisy items but also potentially suggest replacements. This can effectively reduce the impact of erroneous deletions, and the additional items introduced by the replacement operation may indeed provide useful information, benefiting sequential recommendation. We also empirically validate the effectiveness of such generation in our experiments.

Formally, let S' denote the corrected sequence of the original sequence S, where each $s'_t \in S'$ represents the tth item of the corrected sequence. The denoising process in this article can be expressed as follows:

$$s'_t = \begin{cases} s_t, & y_t = 0 \\ r_t, & y_t = 1 \end{cases},$$

where $y_t = 1$ indicates that the denoising model classify the item s_t as noise, while $y_t = 0$ indicates not. r_t denotes the replacement item generated by the LLMs for s_t.

2.3 Background of LLM

LLMs have become transformative tools in NLP. These models, built on the transformer architecture [77], effectively capture relationships in sequential data and can easily scaled to unprecedented sizes, reaching billions of parameters [1, 22, 91, 100]. As the number of parameters increases, LLMs exhibit emergent abilities not present in traditional NLP models, leading to significant improvements in language understanding and generation tasks [92]. Consequently, LLMs have demonstrated remarkable capabilities in tasks such as translation [116], summarization [5], and question-answering [39], making them a pivotal component of modern AI research and applications.

In addition, LLMs pretrained on extensive datasets exhibit strong zero-shot or few-shot capabilities for downstream tasks, requiring only minimal fine-tuning to quickly adapt to specific tasks [111]. Instruction tuning [16] is a fine-tuning method that enables LLMs to better understand and follow human instructions. By training the model on a labeled dataset of instructional prompts and corresponding outputs, instruction tuning significantly improves the model's performance in specified instruction formats. This approach not only refines the model's outputs but also enhances its interpretability and reliability in practical applications. Specifically, the instruction data is organized in the format (x, z), where x represents the text-based instruction and z represents the corresponding response. LLMs have demonstrated a significant improvement in performance on specific tasks when trained with a limited amount of instructional data [111].

3 Methodology

In this section, we present the proposed LLM4DSR. We first employ a self-supervised instruction tuning approach to enable the denoising capability of the LLM (Section 3.1), and then introduce an uncertainty estimation module to improve the reliability of the denoising process (Section 3.2). Finally, we describe the procedure for replacing noisy samples with items suggested by the LLM (Section 3.3). The schematic diagram of LLM4DSR is shown in Figure 3.

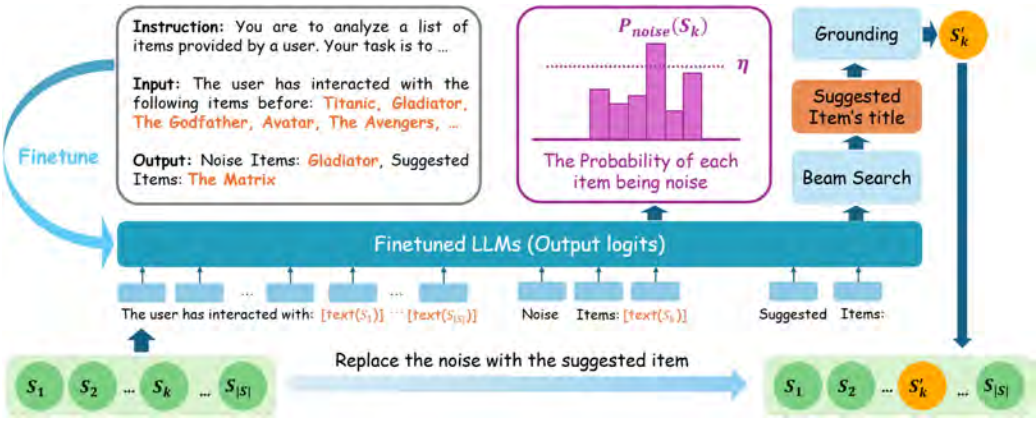

Fig. 3. Schematic diagram of the LLM4DSR.

3.1 Fine-Tuning LLM as a Denoiser

To better activate the denoising capabilities of LLMs, we propose leveraging an instruction tuning strategy. Instruction tuning has been widely adopted in various LLM-based applications and has proven effective in enabling LLMs to quickly adapt to new tasks [111]. Specifically, the template for the instruction data in this task can be formulated as follows:

Template of Instruction Tuning

Instruction: You are to analyze a list of items provided by a user. Your task is to identify an item that do not align with the main interests reflected by the majority of the items. After identifying these noise items, suggest alternative items that better match the user's interests.

Input: The user has interacted with the following items before: $text(s_1)$,... , $text(\hat{s_t})$,... , $text(s_{|S|})$

Output: Noise Items: $text(\hat{s_t})$, Suggested Items: $text(s_t)$

where, $text(s_i)$ represents the textual description of item s_i (e.g., titles). Here, we organize the language descriptions of sequences as instruction inputs and expect the LLMs to identify noisy items and suggest replacements. Specifically, the prompt "*The user has interacted with the following items before*:" provides the model with information about the user's historical interactions. "*Noise Items*:" guides the model to identify noise within the sequence from the semantic context. "*Suggested Items*:" encourages the model to infer the user's interests based on the sequence and suggest replacements that align more closely with the user's preferences.

However, one challenge of such an instruction fine-tuning strategy is the lack of explicit signals denoting the ground truth of the noise items and ideal replacement items. To address this, we propose leveraging a self-supervised strategy to construct instruction data. We randomly select an item s_t from the original sequences and replace it with a noise item $\hat{s_t}$ that is randomly sampled from V. LLMs can be fine-tuned on the corrupted sequence, aiming to predict the modified noise item and make predictions of the original item. The schematic diagram of the self-supervised task is shown

Fig. 4. The self-supervised task for fine-tuning LLMs as denoiser.

in Figure 4. In this way, LLMs can effectively understand the task of sequential recommendation denoising and generalize to identify potential abnormal items in the original sequence and make corrections, which would significantly boost recommendation performance.

Formally, let us denote the token sequence of input part as x and output part as z. Then we can fine-tune the LLMs on the aforementioned corpus and the objective function can be formalized as follows:

$$\max_{\Phi} \sum_{(x,z)} \sum_{i=1}^{|z|} \log \left(P_{\Phi} \left(z_i \mid x, z_{<i} \right) \right), \tag{1}$$

where Φ represents the model parameter and $|z|$ is the total length of token sequence of the output part. $P_{\Phi}\left(z_i \mid x, z_{<i}\right)$ signifies the probability of the token z_i, given the input sequence x and all preceding tokens $z_{<i}$.

3.2 Uncertainty Estimation Module

While the aforementioned instruction tuning has activated the denoising potential of LLMs, they still face two challenges during applications: (1) Given the notorious hallucination phenomenon inherent in LLMs and the complexity of the denoising task, directly relying on the model's response to identify noise items might not be reliable or precise. The model may generate fake items that are not actually present in the sequence, or it may lack high confidence in the given item, thereby compromising the accuracy of the denoising results. (2) The response is limited to addressing a single noisy item per sequence. This is because the training data used during fine-tuning contains only one noisy item in the output, leading the model to consistently output a single item during inference.[1] Consequently, this strategy is inflexible in handling cases where multiple noisy items are present.

To tackle these issues, we introduce an uncertainty estimation strategy. Rather than directly utilizing the response from LLMs, which could be imprecise and inflexible, we opt to scrutinize the probability of each item s_k in the sequence S being regarded as a noise item by LLMs:

$$P_{\text{noise}}(s_k) = P(z_{\text{noise}} = T_k | x) = \prod_{j=1}^{|T_k|} P(T_{k,j} | x, T_{k,<j}), \tag{2}$$

where we examine the generative probability of the description of the item s_k. Here, z_{noise} represents the token sequence of the noise item. T_k denotes the token sequence of $text(s_k)$, $|T_k|$ denotes the total length of T_k, and $T_{k,j}$ represents the jth token of T_k.

Considering the calculation of $P_{noise}(s_k)$ can be time-consuming due to the need to iterate over each token, in practice, we may simplify the process by calculating the probability of only the first token. This is because the first token of the items in a sequence is typically unique, suggesting that the generated item description can be determined primarily by the first token.

[1]We also have attempted to construct instruction data with multiple noisy items and guide the LLM to identify several noise items. However, this approach resulted in very low accuracy. We hypothesize that this is due to the increased difficulty of directly identifying a varying number of noisy items.

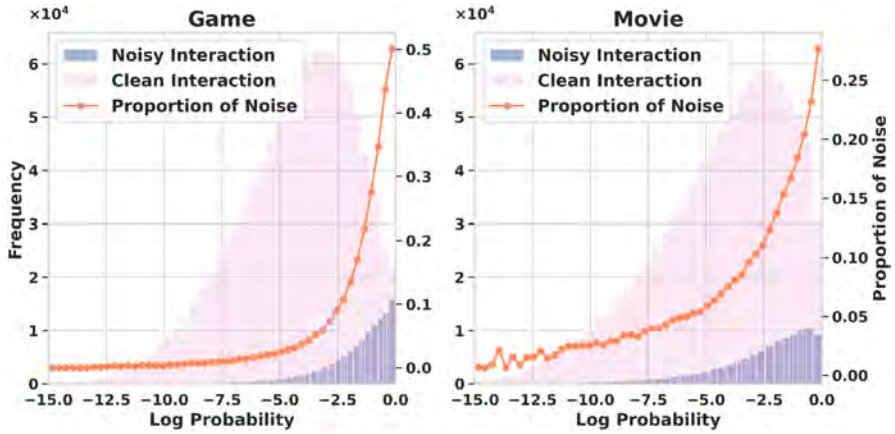

Fig. 5. The distribution of the logarithmic probability of the model-generated samples being noise.

The probability $P_{noise}(s_k)$ reflects the confidence of the LLM's judgment—the higher the probability, the more likely the item is to be a noisy item. To empirically validate its effectiveness, we conducted experiments on a real-world dataset with an artificial noise ratio of 10%.[2] The details of the dataset are provided in Section 4.1.1. As shown in Figure 5, as the probability $P_{noise}(s_k)$ increases, both the number and proportion of noise samples increase, while the number of clean samples decreases.

Thus, in our approach, $P_{noise}(s_k)$ is used to identify noisy items within the sequence. Formally, we apply a threshold η for classification:

$$y_k = \begin{cases} 1, & P_{noise}(s_k) > \eta \\ 0, & P_{noise}(s_k) \leq \eta \end{cases}. \tag{3}$$

This uncertainty-based strategy mitigates the hallucination issue inherent in LLMs by avoiding reliance on potentially erroneous generative outputs. Besides, this strategy provides flexibility in handling multiple noisy items, as the model can assign a high $P_{noise}(s_k)$ to multiple items in the sequence.

3.3 Replace Noise with Suggested Items

Beyond removing noise items, we propose leveraging the generative capabilities of LLMs to suggest replacement items. The replacement items could bring additional knowledge of user preferences, potentially leading to improved recommendation performance. Our reasoning is as follows: (1) Missed Interactions: A user may have interacted with the item on another platform or in a context not captured by the dataset. In such cases, introducing the item as a replacement can help recover missing information and better align with the user's actual preferences. (2) Supplementary Information: The replacement item can serve as supplementary information, enriching the sequence and enhancing the learning process for downstream recommendation models. This is particularly advantageous when the replacement aligns well with the user's inferred preferences at the time. (3) Mitigating the Impact of Erroneous Deletions: Denoising methods often face challenges in accurately distinguishing noise from valid interactions, leading to the erroneous deletion of correct interactions. This is a common issue in such tasks. By introducing a replacement item, the strategy

[2]We use a dataset with artificially injected noise because only in this setting the ground-truth labels for noisy items are available.

can compensate for these errors, as the replacement may still reflect the user's preferences and help maintain the integrity of the sequence.

Specifically, for each identified noisy item s_k, we use the prompt <*User Interactions, Noise Items:* $text(s_k)$*, Suggested Items:*> to guide the LLM in generating a description of the suggested replacement item $text_{suggest}$. Given the hallucination issue of LLMs, $text_{suggest}$ could be a fake item that does not necessarily exist in the item set. Grounding techniques [2] can be applied to find the most close existing item:

$$s'_k = \arg\min_{v_i \in \mathcal{V}} f(text_{suggest}, text(v_i)), \tag{4}$$

where $f(.,.)$ represents the similarity between two textual descriptions, which can be implemented via the embedding similarity of texts. The item s'_k is then used to replace s_k, generating a corrected sequence that can be leveraged to enhance the performance of various recommendation models.

3.4 Discussion

In this section, we investigate whether traditional denoising methods can be enhanced through the integration of LLMs:

- *HSD* [108] relies on modeling both long- and short-term user interests within interaction sequences. It quantifies the inconsistency between each item in the sequence and the user's interest, identifying items with high inconsistency as noise. However, when interaction sequences are short, HSD struggles to accurately capture user interests, leading to reduced accuracy in noise identification. LLMs can mitigate this limitation by leveraging their semantic understanding to model user preferences more effectively. Techniques such as **Chain of Thought (CoT)** [93] can be used to reason through and summarize user interest patterns, thereby enhancing noise identification.
- *STEAM* [58] utilizes a denoising self-supervised learning task to train both a RNN-based noise discriminator and an interaction generator. The noise discriminator is responsible for identifying noisy interactions, while the interaction generator reconstructs missing interactions. However, since STEAM relies solely on item ID and lacks explicit noise labels, its noise discriminator exhibits relatively low accuracy, as demonstrated in Section 4.2.3. LLMs can effectively leverage the meta information of items, such as titles and category attributes. By incorporating this information encoded by LLMs, STEAM can capture richer item representations, thereby enhancing its ability to perform more accurate denoising.
- *CL4SRec* [101] enhances model robustness against noisy data by employing contrastive learning and utilizing sequence augmentation to generate positive samples. The quality of these positive samples is critical for effective training. However, CL4SRec relies on heuristic augmentation, such as item cropping, masking, and reordering, which may inadvertently distort meaningful sequence information. LLMs can enhance the sequence augmentation process by evaluating user intent and ensuring sequence coherence. This allows LLMs to refine sequences while preserving their original semantic structure, ultimately generating higher-quality positive samples.
- *FMLP* [115] incorporates a filter layer after the embedding layer, enabling noise mitigation at the item representation level. Since FMLP modifies the model architecture itself and defines the denoising task at the representation level—rather than at the interaction sequence level—it is challenging to design an appropriate mechanism for LLM integration.

This article explores the implementation of a denoising task using an LLM-based denoising model. The aforementioned concept of an LLM-enhanced traditional denoising model can be considered as a potential direction for future research.

Table 2. Statistics of the Datasets

Datasets	#Users	#Items	#Interactions	#Sequences	#Density
Game	54,955	17,256	494,934	149,393	0.0522%
Toy	19,124	11,758	165,247	47,931	0.0735%
Movie	9,329	5,101	214,980	121,690	0.4518%

4 Experiments

We aim to answer the following **Research Questions (RQs)**:

— *RQ1*: How does LLM4DSR perform compared to existing state-of-the-art sequential recommendation denoising methods?
— *RQ2*: What are the impacts of the components on LLM4DSR?
— *RQ3*: How does the efficiency of LLM4DSR compare to other methods?
— *RQ4*: How does LLM4DSR perform across different user groups and various types of datasets?
— *RQ5*: How does hyperparameter η affect model performance?
— *RQ6*: How does LLM4DSR identify noise and complete the sequence?

4.1 Experimental Settings

4.1.1 Datasets. Five conventional real-world datasets, *Amazon Video Games*, *Amazon Toy and Games*, *Amazon Movies*, *Amazon Instrument*,[3] and *LastFM*,[4] are utilized in our experiments. These datasets are commonly used for the studies of LLM-based sequential recommendation [2, 8, 18, 45, 53]. To ensure a fair comparison, we adopted the same data preprocessing used in recent studies [2]. That is, for each user sequence longer than 11 interactions, a sliding window of length 11 is used to segment the sequences. The sequences are then organized in ascending order of timestamps to partition each dataset into training, validation, and testing sets with ratios of 8:1:1. We randomly retain 20,000 items for *Amazon Movies* due to its large size. The dataset statistics are presented in Table 2.

We consider two noise settings in our experiments: (1) the natural noise setting, which directly utilizes real-world datasets that inherently contain noise introduced during practical data collection; and (2) the natural + artificial noise setting, where additional artificial noise is injected into the real-world datasets. Specifically, a certain proportion of noise, denoted by α, is introduced by replacing each item in a sequence with a randomly selected noise item with probability α. This setting allows us to evaluate the effectiveness of the denoising strategy under varying levels of noise, and also provides ground-truth labels for the noise items that can be used to assess the accuracy of different methods in identifying noise items. Notably, artificial noise is added before segmenting sequences using the sliding window approach, ensuring that no information leakage occurs between sequences.

4.1.2 Baselines. We conducted experiments on three backbones, including the traditional sequential recommendation models SASRec [42] and BERT4Rec [71], as well as a LLM-based recommendation model LLaRA [53]. We compare LLM4DSR with the following denoising baselines[5]:

[3] https://jmcauley.ucsd.edu/data/amazon/index_2014.html.
[4] https://www.last.fm/.
[5] We acknowledge the existence of other explicit denoising methods for sequential recommendation, such as SSDRec [109] and END4Rec [31]. We excluded them from our study because these methods are not compatible with the setting of this article.

— *HSD* [108] is an explicit denoising method. It posits that noisy interactions within a sequence do not align with the user's long-term and short-term interests and exhibit noticeable discontinuities with the sequence context. It jointly identifies noise items in the sequence from both user-level and sequence-level signals.
— *STEAM* [58] is an explicit denoising method. It constructs a self-supervised task by randomly replacing items in the original sequence, training a discriminator and a generator to determine whether an item in the sequence is noise and to complete the sequence, respectively.
— *FMLP* [115] is an implicit denoising method. It uses Fast Fourier Transform and a learnable filter to eliminate noise in the representation.
— *CL4SRec* [101] is an implicit denoising method. It constructs positive samples of the original sequence through transformations such as item cropping, item masking, and item reordering and then uses contrastive learning to enhance the model's robustness to noise in the sequence.

We tested implicit denoising methods only on traditional sequential recommendation backbones. This is due to the coupling of these methods with specific models, which prevents their application to LLM-based recommendation systems.

4.1.3 Evaluation Metrics. We utilize the following metrics to evaluate our method:

— *Hit Ratio@K* measures the proportion of positive items that appear in the top K positions of the recommendation list:

$$\begin{aligned} \mathrm{HR}_u@K &= \frac{1}{|\mathcal{P}(u)|} \sum_{i \in \mathcal{P}(u)} \mathrm{I}\left[q_{ui} \leq K\right] \\ \mathrm{HR}@K &= \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \mathrm{HR}_u@K, \end{aligned} \tag{5}$$

where I[] denotes an indicator function; $\mathcal{P}(u)$ denotes the positive item set in the test data for a user u; q_{ui} denotes the ranking position of the item i for the user u. $\mathcal{U}$ denotes the set of all users.
— *NDCG@K* measures the ranking quality of recommendation through discounted importance based on the position:

$$\begin{aligned} DCG_u@K &= \sum_{i \in \mathcal{P}(u)} \frac{\mathrm{I}[q_{ui} \leq K]}{\log\left(q_{ui}+1\right)} \\ NDCG@K &= \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \frac{DCG_u@K}{IDCG_u@K}. \end{aligned} \tag{6}$$

$IDCG_u@K$ is the $DCG_u@K$ value of the ideal ranking with the optimal ranking for the user u.

In addition to the aforementioned metrics, following prior research [40], we use the following metrics to measure the novelty and diversity of the recommendation:

— ***Intra-List Diversity (ILD)*** measures the similarity between items within a recommendation list to evaluate recommendation diversity:

$$\begin{aligned} \mathrm{ILD}(R_u) &= \frac{2}{n(n-1)} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \mathrm{dissimilarity}(r_i, r_j) \\ \mathrm{ILD} &= \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \mathrm{ILD}\left(R_u\right), \end{aligned} \tag{7}$$

where $R_u = \{r_1, r_2, \ldots, r_n\}$ denotes the recommendation list of the user u and dissimilarity (r_i, r_j) quantifies the difference between items r_i and r_j. We measure dissimilarity using the Jaccard distance based on item category information.

—*Novelty* measures the popularity of the recommended items:

$$\begin{aligned} \text{Novelty}(R_u) &= \frac{1}{n} \sum_{i \in R_u} (1 - \text{popularity\_score}(i)) \\ \text{Novelty} &= \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \text{Novelty}(R_u), \end{aligned} \tag{8}$$

where popularity_score(i) represents the normalized popularity of item i, calculated by dividing the popularity of item i by the maximum popularity observed in the catalog.

4.1.4 Implementation Details. For training LLM4DSR, we select LLaMA3-8B [22] as the backbone and randomly sample 5,000 prompts to fine-tune the model over 50 epochs using LoRA [37]. For the threshold hyperparameter η, we conduct a search within the [0,1] quantiles of the noise probabilities across all items, with a search interval of 0.05. In the experiment, we found that η demonstrated strong robustness. Searching above the 0.7 quantile yielded satisfying results. A sensitivity analysis of this parameter is provided in Section 4.6. Additionally, to prevent excessive modification of any sequence, we limit each sequence to be modified by at most b items where the parameter b is chosen from $\{1, 2, 3\}$.

For SASRec and BERT4Rec, we fix the learning rate at 0.001 and the embedding size at 64. For the LLM-based recommendation model, LLaRA, the original configuration involves selecting the Top 1 item from the candidate items as the recommendation result, without item ranking process, which prevents the calculation of previously introduced evaluation metrics. To align with standard experimental settings, we follow [2] and implement a grounding operation to enable LLaRA rank the entire item set.

We use the source code provided in the original papers and search for optimal hyperparameters for all comparison methods according to the instructions in the original papers. Besides, for the two explicit denoising methods, HSD and STEAM, we performed end-to-end training on the backbone used in the original paper (i.e., BERT4Rec) and generate the denoised dataset. To facilitate comparison with other backbones (i.e., SASRec and LLaRA), we transferred the denoised dataset to these backbones and trained them from scratch.

4.2 Performance Comparison (RQ1)

In this section, we analyze the superior of LLM4DSR under two different noise settings as compared with other baselines.

4.2.1 Evaluations on the Natural Noise Setting. In Tables 3 and 4, we present the test results of various methods on real-world datasets. Notably, LLM4DSR consistently achieves the best performance across almost all datasets.

For explicit denoising methods (i.e., STEAM and HSD), we observed that while these methods perform well on BERT4Rec, their effectiveness diminishes when retraining on SASRec using the denoised dataset they generated. For instance, on Game dataset, STEAM improves NDCG@20 from 0.0351 to 0.0373 on BERT4Rec. However, when the same denoised dataset is evaluated on SASRec, NDCG@20 drops from 0.0299 to 0.0233. This suggests that the denoised dataset generated by STEAM lacks good transferability. Additionally, the performance of HSD significantly declined. We argue that this issue may arise from HSD's inability to effectively capture sequence information in short sequence scenarios. This limitation can result in the loss of critical information when incorrect

Table 3. The NDCG Performance Comparison on the Natural Noise Setting Using Different Backbone

Backbone	Method	Game		Toy		Movie	
		NDCG@20	NDCG@50	NDCG@20	NDCG@50	NDCG@20	NDCG@50
SASRec	None	0.0299	0.0377	0.0190	0.0239	0.0468	0.0548
	FMLP	0.0297	0.0377	0.0200	0.0245	0.0488	0.0574
	CL4SRec	**0.0331**	**0.0405**	0.0201	0.0247	0.0492	0.0579
	HSD	0.0246	0.0324	0.0121	0.0172	0.0369	0.0451
	STEAM	0.0233	0.0325	0.0187	0.0242	0.0475	0.0569
	LLM4DSR	0.0315	0.0404	**0.0208**	**0.0262**	**0.0505**	**0.0597**
BERT4Rec	None	0.0351	0.0436	0.0205	0.0256	0.0584	0.0684
	FMLP	0.0369	0.0449	0.0195	0.0242	0.0554	0.0641
	CL4SRec	0.0359	0.0438	0.0192	0.0246	0.0587	0.0676
	HSD	0.0301	0.0387	0.0187	0.0230	0.0467	0.0565
	STEAM	0.0373	0.0453	0.0195	0.0250	0.0548	0.0643
	LLM4DSR	**0.0375**	**0.0464**	**0.0222**	**0.0280**	**0.0604**	**0.0697**
LLaRA	None	0.0261	0.0341	0.0224	0.0275	0.0388	0.0472
	HSD	0.0231	0.0305	0.0136	0.0188	0.0271	0.0347
	STEAM	0.0200	0.0284	0.0182	0.0244	0.0396	0.0488
	LLM4DSR	**0.0290**	**0.0375**	**0.0241**	**0.0306**	**0.0409**	**0.0504**

The best result is bolded and the runner-up is underlined.

items are deleted. In contrast, LLM4DSR demonstrates consistently superior performance across all datasets and backbones, proving its ability to effectively identify and replace noisy samples based on the internal knowledge of LLMs.

For implicit denoising methods (i.e., FMLP and CL4SRec), we found that their performance is unstable and can even decline on certain datasets. For example, on the Toy dataset, both FMLP and CL4SRec experienced performance drops, with NDCG@20 decreasing from 0.256 to 0.242 and 0.246, respectively. We hypothesize that for CL4SRec, the substantial difference from the original sequence during the sequence enhancement phase may degrade the quality of positive samples, and using contrastive loss for alignment might introduce additional noise. FMLP, which performs denoising at the representation level through filtering, appears unable to effectively mitigate the impact of noise samples within the sequence.

4.2.2 Evaluations on the Natural + Artificial Noise Setting. The test results for various methods on the natural + artificial noise setting, with a noise ratio of 10%, are presented in Tables 5 and 6. The results indicate that LLM4DSR consistently outperforms all comparative methods. Artificial noise, being more distinguishable than natural noise in conventional real-world datasets, makes the improvement of LLM4DSR more pronounced on such datasets. For example, on Game dataset, LLM4DSR combined with SASRec achieves a 5.3% improvement in NDCG@20, whereas on the artificially noise-added dataset, the improvement reaches 15.8%. Furthermore, we evaluated the performance of all comparative methods under different noise ratios, as illustrated in Figure 6. LLM4DSR significantly surpasses other comparative methods demonstrate its robust performance across varying noise conditions.

Table 4. The Hit Ratio Performance Comparison on the Natural Noise Setting Using Different Backbone

Backbone	Method	Game		Toy		Movie	
		HR@20	HR@50	HR@20	HR@50	HR@20	HR@50
SASRec	None	0.0646	0.1040	0.0501	0.0749	0.0900	0.1308
	FMLP	0.0600	0.1004	0.0532	0.0761	0.0924	0.1360
	CL4SRec	0.0696	0.1072	0.0515	0.0751	0.0958	0.1400
	HSD	0.0514	0.0908	0.0319	0.0582	0.0780	0.1196
	STEAM	0.0578	0.1052	0.0501	0.0778	0.0906	0.1386
	LLM4DSR	**0.0696**	**0.1148**	**0.0557**	**0.0830**	**0.0976**	**0.1440**
BERT4Rec	None	0.0688	0.1120	0.0461	0.0718	0.0982	0.1484
	FMLP	0.0710	0.1116	0.0426	0.0699	0.0952	0.1394
	CL4SRec	0.0692	0.1088	0.0469	0.0738	0.1022	0.1476
	HSD	0.0708	0.1144	0.0430	0.0651	0.0802	0.1298
	STEAM	0.0714	0.1122	0.0446	0.0724	0.0932	0.1414
	LLM4DSR	**0.0748**	**0.1202**	**0.0505**	**0.0799**	**0.1046**	**0.1516**
LLaRA	None	0.0512	0.0918	0.0538	0.0797	0.0786	0.1208
	HSD	0.0490	0.0866	0.0357	0.0617	0.0688	0.1072
	STEAM	0.0554	0.0990	0.0474	0.0786	0.0820	0.1288
	LLM4DSR	**0.0602**	**0.1026**	**0.0576**	**0.0905**	**0.0870**	**0.1350**

The best result is bolded and the runner-up is underlined.

4.2.3 Evaluation of the Accuracy of Explicit Denoising Methods in Identifying Artificial Noise. The explicit denoising method achieves denoising by identifying and removing noise, with the accuracy of noise identification being closely related to its overall effectiveness. To better understand the performance of explicit denoising methods, we evaluated the F1 score of various explicit denoising methods on datasets with artificially introduced noise, where noise labels are available. As illustrated in Figure 7, we compare LLM4DSR with two other state-of-the-art explicit denoising methods on the Game and Toy datasets with different noise ratios of 10%, 20%, and 30%. Among these methods, LLM4DSR consistently achieved the highest performance, suggesting that leveraging the open knowledge of LLM can significantly enhance noise identification.

4.2.4 Evaluation on Diversity and Novelty. This section investigates whether LLM4DSR influences the diversity and novelty of the recommendation system. The metrics used to evaluate diversity and novelty are detailed in Section 4.1.3. Experiments are conducted on naturally noisy datasets using SASRec as the backbone. We evaluate the top 10 ranked items as the recommendation list R_u. As shown in Table 7, the diversity and novelty of LLM4DSR are comparable to SASRec, which suggests that LLM4DSR does not have a significant effect on the diversity or novelty of the recommendations.

4.3 Ablation Study (RQ2)

4.3.1 The Impact of Different Components in LLM4DSR. To verify the contributions of various components of LLM4DSR, we conducted an ablation study on different variants under SASRec, using both real and artificially noisy datasets from the Game and Toy datasets. The variants tested include: (1) "w/o uncertainty," which omits uncertainty estimation and directly uses the LLM

Table 5. The NDCG Performance Comparison on the Natural + Artificial Noise Setting Using Different Backbone

Backbone	Method	Game		Toy		Movie	
		NDCG@20	NDCG@50	NDCG@20	NDCG@50	NDCG@20	NDCG@50
SASRec	None	0.0252	0.0314	0.0139	0.0178	0.0390	0.0455
	FMLP	0.0253	0.0315	0.0138	0.0173	0.0415	0.0484
	CL4SRec	0.0278	0.0336	0.0151	0.0183	0.0405	0.0464
	HSD	0.0198	0.0250	0.0090	0.0127	0.0340	0.0407
	STEAM	0.0261	0.0333	0.0123	0.0161	0.0401	0.0476
	LLM4DSR	**0.0292**	**0.0357**	**0.0152**	**0.0200**	**0.0431**	**0.0517**
BERT4Rec	None	0.0266	0.0335	0.0150	0.0195	0.0485	0.0569
	FMLP	0.0281	0.0353	0.0147	0.0188	0.0478	0.0561
	CL4SRec	0.0268	0.0341	0.0153	0.0196	0.0498	0.0579
	HSD	0.0288	0.0361	0.0152	0.0189	0.0390	0.0459
	STEAM	0.0294	0.0367	0.0182	0.0218	0.0472	0.0544
	LLM4DSR	**0.0316**	**0.0397**	**0.0183**	**0.0231**	**0.0509**	**0.0591**
LLaRA	None	0.0202	0.0264	**0.0152**	0.0195	0.0323	0.0387
	HSD	0.0146	0.0198	0.0116	0.0153	0.0197	0.0267
	STEAM	0.0207	0.0275	0.0144	0.0191	0.0325	0.0395
	LLM4DSR	**0.0252**	**0.0321**	0.0150	**0.0204**	**0.0358**	**0.0445**

The noise ratio is 10%. The best result is bolded and the runner-up is underlined.

output for denoising; (2) "w/o completion," which excludes the completion capability of LLM4DSR and performs only deletion operations on sequences; and (3) "w/ popularity completion," which replaces noise with items randomly sampled from the item set based on popularity weights. As presented in Table 8, the consistently superior performance of LLM4DSR across all datasets highlights the effectiveness of each component. In particular, the performance decline observed in the "w/o completion" and "w/ popularity completion" highlights the critical role of the replacement operation. This demonstrates that LLM4DSR effectively extracts user interests and suggests items that better align with the sequence, while also mitigating the negative impact of erroneous deletions.

4.3.2 Performance of Separately Denoising on Training and Test Sets. To more effectively demonstrate the impact of LLM4DSR on model performance, we evaluated the model after denoising the training and testing sets separately. As depicted in Figure 8, we conducted experiments on the Game and Toy datasets. Figure 8(a) presents the results on real-world datasets, while Figure 8(b) shows results on datasets with 10% artificial noise. Across all datasets and settings, denoising both the training and testing sets individually enhanced performance, with greater improvements observed when denoising the training set. Furthermore, simultaneously denoising both sets led to the most significant performance gains. The results highlight the importance of clean data in both training and evaluation phases, and demonstrate that LLM4DSR can be a valuable tool in enhancing data quality, ultimately leading to better model performance.

4.3.3 Comparison with Using Perplexity as a Noise Uncertainty Discrimination Metric. In this section, we replace the noise discrimination metric in LLM4DSR with perplexity (PPL), a widely used

Table 6. The Hit Ratio Performance Comparison on the Natural + Artificial Noise Setting Using Different Backbone

Backbone	Method	Game		Toy		Movie	
		HR@20	HR@50	HR@20	HR@50	HR@20	HR@50
SASRec	None	0.0526	0.0838	0.0348	0.0542	0.0750	0.1084
	FMLP	0.0508	0.0824	0.0365	0.0547	0.0772	0.1120
	CL4SRec	0.0556	0.0864	0.0394	0.0557	0.0770	0.1070
	HSD	0.0412	0.0678	0.0254	0.0444	0.0700	0.1042
	STEAM	0.0532	0.0900	0.0321	0.0513	0.0740	0.1118
	LLM4DSR	**0.0624**	**0.0956**	**0.0401**	**0.0642**	**0.0876**	**0.1314**
BERT4Rec	None	0.0534	0.0880	0.0346	0.0574	0.0826	0.1258
	FMLP	0.0598	0.0962	0.0330	0.0536	0.0818	0.1240
	CL4SRec	0.0538	0.0906	0.0369	0.0586	0.0864	0.1276
	HSD	0.0600	0.0970	0.0340	0.0530	0.0682	0.1032
	STEAM	0.0562	0.0936	0.0388	0.0572	0.0836	0.1202
	LLM4DSR	**0.0644**	**0.1056**	**0.0401**	**0.0645**	**0.0890**	**0.1306**
LLaRA	None	0.0466	0.0784	0.0363	0.0582	0.0668	0.0992
	HSD	0.0332	0.0598	0.0273	0.0459	0.0482	0.0842
	STEAM	0.0476	0.0822	0.0361	0.0599	0.0664	0.1018
	LLM4DSR	**0.0514**	**0.0866**	**0.0380**	**0.0655**	**0.0770**	**0.1216**

The noise ratio is 10%. The best result is bolded and the runner-up is underlined.

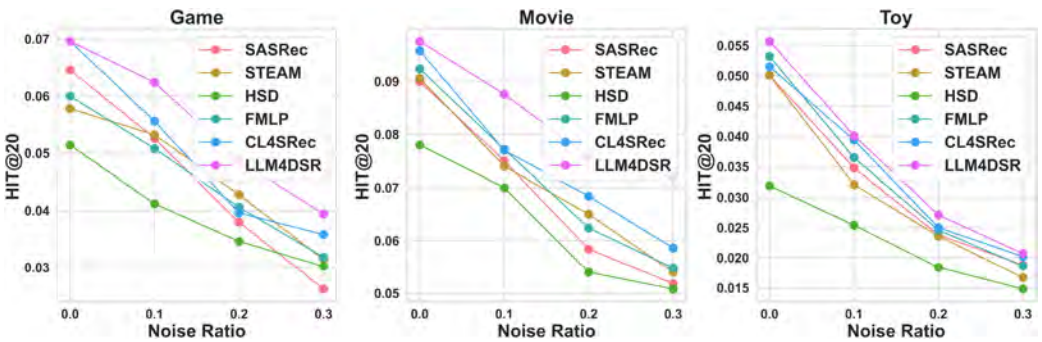

Fig. 6. The performance of different methods on the natural + artificial noise setting with varying noise ratios from 0% to 30%. SASRec serves as the backbone model. A noise ratio of 0.0 represents a real-world dataset containing natural noise.

measure of uncertainty in NLP [6], and compare its performance with the original LLM4DSR. PPL mitigates the issue of longer sequences receiving lower scores by applying length normalization. As shown in Table 9, LLM4DSR, which utilizes the probability of the first token for noise discrimination, consistently outperforms the perplexity-based method across all datasets. This result suggests that the probability of an item's first token is a more effective indicator for assessing uncertainty in noise determination. Furthermore, as presented in Table 10, we examine the average ratio of items

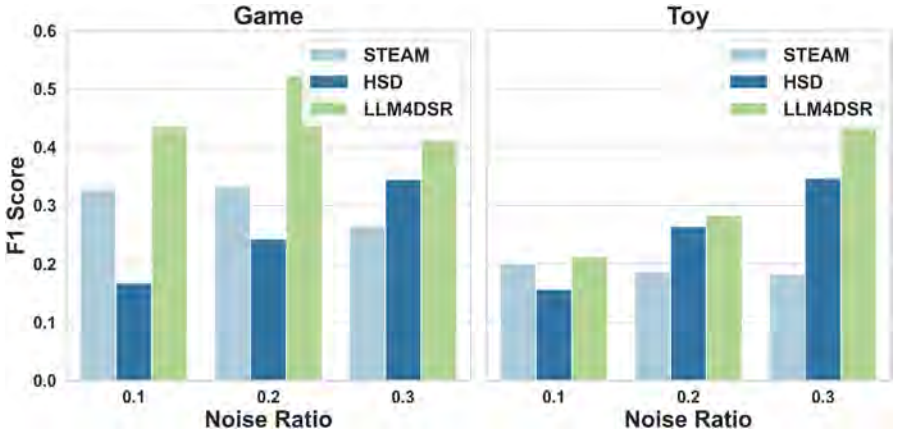

Fig. 7. F1 score of explicit denoising method for noise recognition.

Table 7. The Evaluation Results for Novelty and Diversity Metric on Naturally Noisy Datasets

Method	Metric	Game	Movie	Toy
SASRec	Diversity	0.4943	0.7252	\
	Novelty	0.9002	0.7599	0.8565
LLM4DSR	Diversity	0.5009	0.7226	\
	Novelty	0.9040	0.7578	0.8762

The Toy dataset cannot be used to test diversity due to incomplete item category information.

within a user interaction sequence whose first token is duplicated by other items. The relatively low duplicated ratio indicates that the first token effectively distinguishes an item in most cases.

4.3.4 Comparison of the Performance Using Different LLM Tuning and Noise Handling Strategies. In this section, we evaluate the impact of different LLM tuning and noise-handling strategies on denoising performance. We conducted experiments using four types of tuning strategies: (1) *Zero-shot*: Directly using the LLaMA3 model without any fine-tuning. (2) *ICL*: Applying an in-context learning strategy to the pretrained LLaMA3 model without fine-tuning. (3) *Rec tune*: Fine-tuning LLaMA3 on a sequential recommendation task. (4) *Denoise tune*: Fine-tuning LLaMA3 using the self-supervised learning strategy proposed in this article. Additionally, we evaluated two distinct noise-handling strategies: (1) *Delete*: Directly removing noisy items from the user interaction sequence. (2) *Replace*: Replacing noisy items with items the user might be interested in, as inferred by LLMs (this strategy is incorporated in LLM4DSR).

As shown in Figure 9, experimental results on three naturally noisy datasets demonstrate that the "Replace" strategy consistently outperforms the "Delete" strategy, indicating that replacing noisy samples effectively mitigates the risk of information loss due to erroneous deletions while also providing supplementary information. Furthermore, LLMs fine-tuned using the proposed self-supervised learning approach achieve superior performance compared to other methods. While pretrained LLMs exhibit some degree of generalization to unseen tasks, self-supervised fine-tuning specifically tailored for denoising further enhances their effectiveness. In contrast, the performance

Table 8. The Impact of Different Components in LLM4DSR

Dataset	Method	NDCG@20	HR@20
Game	SASRec	0.0299	0.0646
	w/o uncertainty	0.0309	0.0648
	w/o completion	0.0241	0.0566
	w/ popularity completion	0.0239	0.0460
	LLM4DSR	**0.0315**	**0.0696**
Toy	SASRec	0.0190	0.0501
	w/o uncertainty	0.0191	0.0501
	w/o completion	0.0185	0.0478
	w/ popularity completion	0.0156	0.0417
	LLM4DSR	**0.0208**	**0.0557**
Game w/ artificial noise	SASRec	0.0252	0.0526
	w/o uncertainty	0.0277	0.0572
	w/o completion	0.0264	0.0548
	w/ popularity completion	0.0241	0.0472
	LLM4DSR	**0.0292**	**0.0624**
Toy w/ artificial noise	SASRec	0.0139	0.0348
	w/o uncertainty	0.0133	0.0361
	w/o completion	0.0142	0.0359
	w/ popularity completion	0.0106	0.0267
	LLM4DSR	**0.0152**	**0.0401**

"w/ artificial noise" indicates a dataset with artificially added noise and noise ratio is 10%. The best result is bolded.

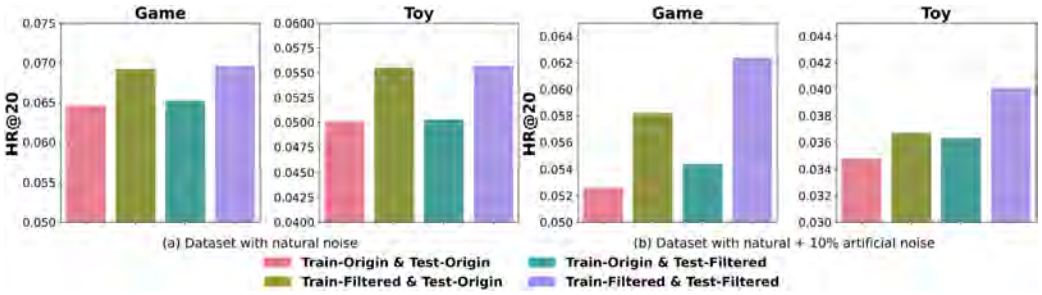

Fig. 8. The performance of LLM4DSR for denoising the train and test sets, respectively. Panel (a) shows the experimental results on the natural noise setting, and Panel (b) shows the experimental results on the natural + artificial noise setting with 10% artificial noise. The experiment is conducted with SASRec as the backbone.

improvements achieved through in-context learning and fine-tuning on a recommendation task are relatively limited.

4.4 Efficiency Comparison (RQ3)

In this section, we compare the per-epoch training time of different methods on datasets with natural noise and analyze the computational overhead associated with LLM4DSR during the data preprocessing phase, as shown in Table 11.

LLM4DSR requires the training and inference of LLMs, resulting in higher computational costs compared to other denoising methods. However, this overhead is confined in the data preprocessing stage and does not impact the efficiency of downstream tasks. Besides, for a given dataset, this

Table 9. Experiments Using Different Uncertainty Discrimination Indicators

Dataset	Method	NDCG@50	HIT@50
Game	PPL	0.0389	0.1100
	LLM4DSR	0.0404	0.1148
Toy	PPL	0.0251	0.0809
	LLM4DSR	0.0262	0.0830
Game w/ artificial noise	PPL	0.0336	0.0924
	LLM4DSR	0.0357	0.0956
Toy w/ artificial noise	PPL	0.0193	0.0630
	LLM4DSR	0.0200	0.0642

"w/ artificial noise" indicates a dataset with artificially added noise and noise ratio is 10%.

Table 10. The Average Ratio of Items Whose First Token Is Duplicated by Other Items within the Same Sequence on the Dataset with Artificially Added Noise and Noise Ratio is 10%

Dataset	Duplicated Ratio
Game	0.0852
Toy	0.0599
Movie	0.0812

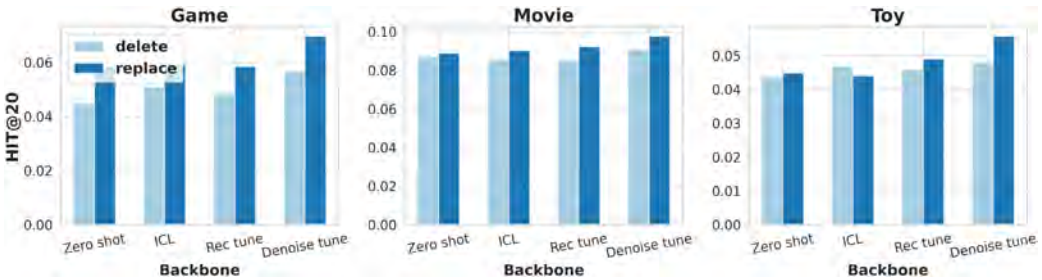

Fig. 9. Comparison of the performance using different tuning strategies and noise handling strategies.

procedure needs to be conducted only once, and the refined dataset can be seamlessly integrated into any downstream task, underscoring its applicability in industrial scenarios.

4.5 Comparative Analysis across Datasets and User Groups (RQ4)

4.5.1 Analysis across User Groups. In this section, we assess the performance of LLM4DSR across different user groups. Specifically, we categorize users into two groups based on their interaction frequency. The top 10% of users with the highest number of interactions are designated as warm users, while the bottom 10% are classified as cold users. Figure 10 illustrates the improvement in HR@50 for both groups. The experimental results demonstrate that LLM4DSR achieves a greater performance improvement for warm users than for cold users across both the Toy and Movie datasets. This consistent trend across datasets suggests that LLM4DSR benefits more from users with richer interaction, leading to significantly enhanced denoising effectiveness for warm users.

4.5.2 Analysis across Datasets. In this section, we assess the performance of LLM4DSR using datasets from diverse domains. Our experiments are conducted on five datasets encompassing various item categories, including the music-related LastFM and the instrument-related Amazon Instrument. Figure 11 presents the improvement in HR@50 achieved by integrating LLM4DSR with SASRec under both "natural noise setting" and "natural + artificial noise setting." The results demonstrate that LLM4DSR consistently enhances performance across all datasets in both noise settings, highlighting its effectiveness and generalizability across different domain-specific datasets.

Table 11. Comparison of Per-Epoch Training Time for Different Methods and the Preprocessing Time in LLM4DSR

Method	Per-Epoch Training Time (s)		
	Game	Movie	Toy
SASRec	4.89	4.14	1.85
CL4Rec	13.80	11.29	5.42
FMLP	6.72	5.55	3.44
STEAM	26.61	15.01	8.15
HSD	55.96	23.70	17.52
LLM4DSR	Data Preprocessing Time (s)		
	5,804	4,654	3,823
	Per-Epoch Training Time (s)		
	4.86	4.24	1.86

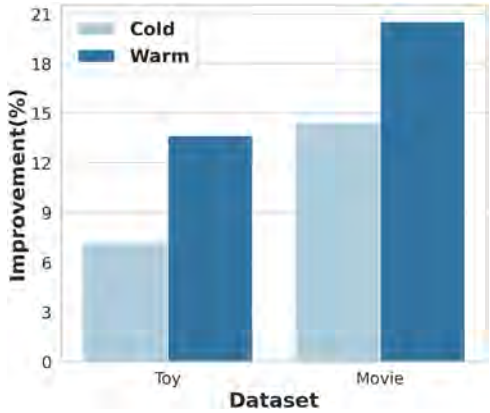

Fig. 10. Performance improvement of LLM4DSR across user groups with varying numbers of interactions in terms of the HR@50.

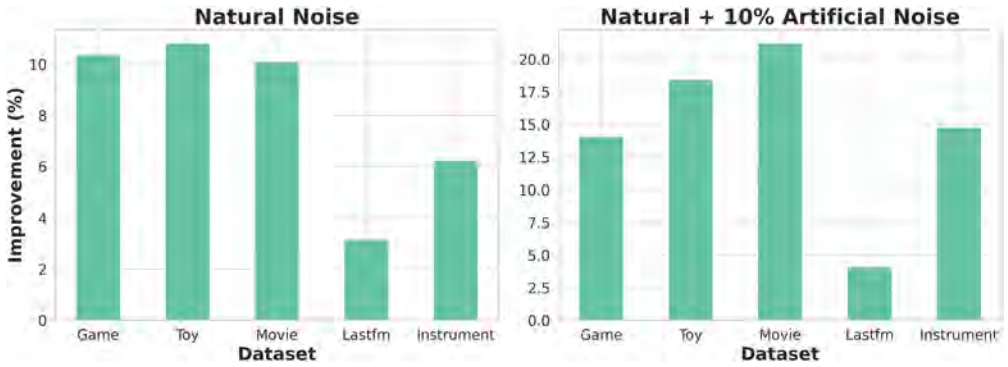

Fig. 11. The performance improvement of LLM4DSR on different datasets. The artificial noise ratio is 10%.

Furthermore, we observe that LLM4DSR performs better under the "natural + artificial noise setting" compared to the "natural noise setting," which can be attributed to the fact that artificial noise is more easily distinguishable from genuine interactions due to its inconsistency, making it easier for the model to identify and mitigate its impact.

4.6 Sensitivity Analysis of Hyperparameter η (RQ5)

In the Uncertainty Estimation module, η serves as the threshold distinguishing classification noise from clean samples. As η increases, fewer items are identified as noise, leading to a lower modification ratio of the original sequence. To better understand the impact of η on model performance, we examined the relationship between the hyperparameter η and model performance under two noise settings, as illustrated in Figure 12. The x-axis represents the quantile of η; a quantile of 1 indicates that all samples are classified as clean, equivalent to training on the original dataset, whereas a quantile of 0 indicates that all samples are identified as noise and replaced (note that we limit the maximum number of modified items per sequence to b). We adjusted η only for the training set, without filtering noise from the test data, to accurately reflect the impact of parameter changes.

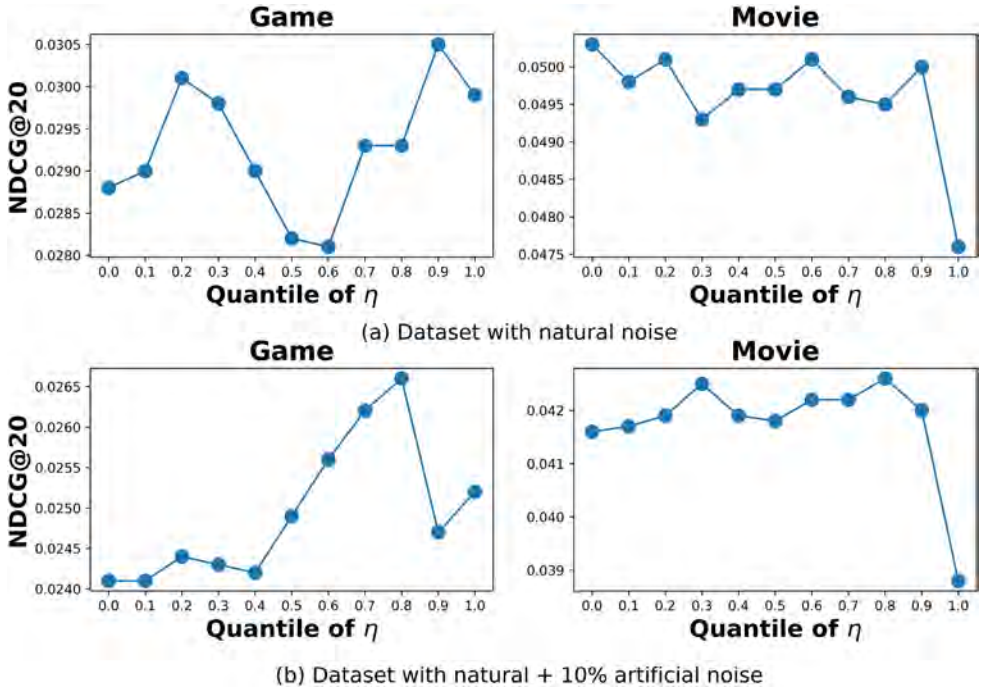

Fig. 12. Sensitivity analysis of hyperparameter η.

Figure 12(a) shows the results on the natural noise setting, while Figure 12(b) presents the results on the natural + artificial noise setting.

On the Game dataset under the natural + artificial noise setting, NDCG@20 first increases and then decreases as η is reduced. The initial increase reflects LLM4DSR's ability to filter out many high-confidence noisy samples; the subsequent decline occurs once the method begins to modify some clean samples. For Game under the natural noise setting, NDCG@20 exhibits a secondary rise, likely because many noisy samples are concentrated among medium-confidence instances. On the Movie dataset, under both noise settings, NDCG@20 increases as η decreases and then remains stable, indicating that LLM4DSR's replacement operations effectively offset the information loss caused by erroneous deletions, thereby improving robustness to the choice of η. Overall, performance is relatively insensitive to η. In practice, satisfactory results can be achieved by searching over relatively large values of η (e.g., above the 0.7 quantile).

4.7 Case Study (RQ6)

In this case study, we illustrate how LLM4DSR leverages open knowledge to achieve denoising. As depicted in Figure 13, we analyze a user's viewing history from the Movie dataset, which predominantly consists of films in the drama, romance, and comedy genres. A notable outlier in this collection is the science fiction film "*Escape from the Planet of the Apes*." Utilizing prior knowledge about movie classifications, the LLM identifies this film as noise with a probability of 0.967. To enhance the sequence information, LLM4DSR recommends "*The Man from Snowy River*," a film that aligns with the user's preferred drama/romance genres, as a suitable replacement for the identified noise.

Fig. 13. A case study on *Amazon Movies* shows how LLMs use open knowledge to identify and replace noise.

5 Related Work

5.1 Sequential Recommendation

The sequential recommendation infers the next item of interest to the user based on their historical interactions. Compared to collaborative filtering [13, 60, 104], sequential recommendation takes into account the temporal order of interactions to better capture the dynamic changes in user interests. Benefiting from the development of deep learning models in recent years, sequential recommendation employs various deep models to model users' historical interactions and capture their interests. For instance, GRU4Rec [33] uses RNN, while Caser [74] uses CNN. SASRec [42] and BERT4Rec [71] are based on the transformers architecture [14, 77], which automatically learns the weight of each interaction.

In addition to innovations in model architecture, recent sequential recommendation models try to address the challenge of capturing rapidly changing user interests over time. For example, DROS [106] introduces distributionally robust optimization [64, 78, 96] to enhance the model's ability to handle out-of-distribution problems caused by temporal changes. SURGE [9] uses GNNs [21] to capture dynamic user interests within implicit interactions. Some other methods focus on the intrinsic biases [11, 25, 55, 114].

Furthermore, some studies have incorporated contrastive learning into sequential recommendations to alleviate the issue of sparse recommendation data. Such methods primarily focus on how to construct positive and negative samples within the contrastive loss. CL4SRec [101] constructs positive samples of the original sequence through sequence transformations such as item cropping, item masking, and item reordering. DuoRec [62] proposes a model-level data augmentation method based on dropout to achieve better semantic retention. DCRec [105] introduces a contrastive learning loss based on conformity and interest disentanglement to address the popularity bias present in data augmentation. SoftCSR [113] extends contrastive learning to regional-level sample comparisons, offering more flexibility than single-sample contrasts, and employs adversarial contrastive loss to enhance model robustness.

The readers may refer to the excellent survey [24, 83] for more details. However, the aforementioned methods do not consider the negative impact of noisy interactions on modeling the sequence, which is quite common in real-world scenarios.

5.2 Denoising Recommendation

In this section, we first review the current sequential recommendation denoising schemes, followed by a discussion of denoising recommendation under collaborative filtering scenarios.

5.2.1 Denoising Sequential Recommendation. The primary objective of denoising sequential recommendation is to enhance the performance of recommendation systems by mitigating the impact of noisy data. Denoising methods for sequential recommendation can be broadly categorized into implicit denoising and explicit denoising. Implicit denoising methods primarily reduce the impact of noise without explicitly removing them. Some works reduce the influence of noisy interactions to the sequence representation by increasing the sparsity of attention weights in attention-based models [10, 107]. FMLP and END4Rec [31, 115] consider using filters to remove noise signals from representations. The drawback of these methods is that noise still affect the model, and their high coupling with the model architecture makes it difficult to transfer them to other backbones or downstream tasks, including the widely researched LLM-based recommendation systems in recent years.

Explicit denoising methods directly modify the sequence content, effectively addressing the limitations of implicit denoising techniques. These methods often rely on heuristically designed denoising modules or metrics. BERD [72] models the uncertainty of each interaction and eliminates modules with high loss but low uncertainty to achieve denoising. HSD [108] assesses the similarity of each sample to other items in the sequence and its alignment with user interests to detect noise. STEAM [58] trains a discriminator model and a generator model, which are used to identify noise and complete sequences, respectively.

Regardless of whether explicit or implicit denoising methods are used, the most significant challenge stems from the lack of effective supervision signals from noise labels to train the denoising model. Existing methods often rely on heuristic designs inspired by prior knowledge to develop denoising algorithms, which requires substantial expert knowledge and lacks precision. Additionally, since these methods usually utilize supervision information from recommendation loss, they are still adversely affected by noise as the labels of the recommendation label (i.e., the next item) could also be contaminated. Our proposed method, LLM4DSR, addresses these issues by leveraging the extensive open knowledge of LLMs to achieve more effective denoising.

5.2.2 Denoising Recommendation under Collaborative Filtering Scenarios. Considering other recommendation scenarios, most of work focuses on denoising under collaborative filtering scenarios. Due to the lack of labeled noise data, some works utilize prior knowledge about noise samples to aid in denoising for implicit feedback data. For instance, ADT [84] and SGDL [26] utilize the phenomenon that models tend to fit clean samples more easily during the early stages of training, while noisy samples exhibit higher loss, to design denoising strategies. ADT simply sets a threshold to reduce the weights of samples with higher loss, whereas SGDL uses the information from the clean samples that the model has fitted to help train a sample weight adjustment model. DeCA [88] identifies noise by observing that different models exhibit significant prediction differences for noise samples. However, such prior knowledge is not always reliable. For example, Shu et al. [69] point out that samples with high loss are not necessarily noise and could also be hard samples. RGCF [76] and GraphDA [23] utilize node features extracted by GNNs [21, 98] to reconstruct the graph structure based on feature similarity, aiming to reduce the impact of noise signals in the graph. BOD [89] employs a bi-level optimization approach to train a weight generator that automatically learns the weight of each sample. These methods often involve complex module designs and, due to the lack of labeled noise signals, their performance is still not guaranteed.

To the best of our knowledge, a recent study has also explored the use of LLMs for recommendation denoising in the collaborative filtering scenario [70]. However, our approach, LLM4DSR, significantly differs from LLMHD in several key aspects: (1) Different tasks: LLM4DSR addresses the task of sequential recommendation, while LLMHD focuses on collaborative filtering. Specifically, our work targets denoising sequential data, whereas LLMHD aims to differentiate between noisy samples and hard negative samples; (2) Different methods: LLM4DSR directly employs instruction tuning and uncertainty estimation to tackle the denoising task, while LLMHD directly uses untuned LLMs and crafts specific prompts to reassess samples, identifying those that deviate from traditional model predictions as noise. Moreover, LLMHD [70] is a preprint available on arXiv as of 16 September 2024, and has not been formally published. Therefore, we consider our work to be parallel to LLMHD.

5.3 LLMs for Recommendation

LLMs, with their extensive knowledge, generalization capabilities, and reasoning abilities, have achieved remarkable results across numerous fields. LLMs are utilized in various recommendation tasks, including collaborative filtering [86, 117], sequential recommendation [2, 34, 35], graph-based recommendation [94], and Click-Through Rate tasks [4, 81, 102]. Additionally, LLMs have been utilized for tasks that were challenging for traditional model to achieve, such as explaining item embeddings [75], explaining recommendation results [27, 87], and conversational recommendations [27]. Two primary paradigms for using LLMs in RS are as follows.

LLM-Based Recommenders. The initial attempts directly use pretrained LLMs as the backbone for recommendations to explore their zero-shot capabilities [27, 36, 59, 82, 90]. These methods perform not well due to the significant gap between recommendation tasks and the training tasks of LLMs. Subsequent research has organized recommendation data into prompt formats and then fine-tuned LLMs to enhance their recommendation performance, achieving better results [2, 4, 28, 38, 47, 49, 50, 52, 68, 73, 79, 80, 110, 112]. Some works finetune LLMs on individual recommendation tasks, then use consistent prompt formats for prediction [2, 4, 46, 51]. Building on this, other studies have extended fine-tuning across multiple recommendation tasks (e.g., sequential recommendation, rating prediction, review summarization) to enhance generalization and multi-task learning capabilities [8, 30].

However, researchers have observed that LLMs, relying solely on semantic understanding, struggle to capture collaborative signals, which are critical for recommendation performance [44]. Consequently, significant efforts have been devoted to integrating collaborative signals into the LLM-based RS. One approach involves leveraging the collaborative signals learned by traditional ID-based recommendation models and incorporating them into LLMs. For instance, some works [44, 53] represent items using both text descriptions (which contains semantic information) and embeddings learned by traditional recommendation models (which encode collaborative filtering signals). Some other works [86] construct item indices (token IDs) based on embeddings learned from traditional recommendation models. Beyond explicitly integrating traditional recommendation models, CoRAL [95] has explored reinforcement learning-based methods to enable LLMs to directly capture collaborative signals from training data. Additionally, S-DPO [15] improves the loss function to better align with the objectives of recommendation tasks based on the DPO [63] framework, which incorporates the ranking relationships between positive and negative samples.

Furthermore, researchers have investigated various challenges associated with LLM-based RS, such as handling long historical interaction sequences [29, 54], reducing training [57] and inference [29, 47] costs, and mitigating recommendation homogenization issues [3], constructing item indexes [34, 56, 65]. Addressing these challenges is crucial for improving the practical deployment of LLM-based RS.

LLM-Enhanced Recommenders. Besides generating recommendation results directly, LLMs can also serve as enhancers to improve the performance of conventional recommenders. Existing methods mainly employ LLMs as encoders to encode the semantic information of users and items into embeddings [17, 66, 81, 94, 99] or as an additional knowledge base [61, 103]. However, such methods largely fail to leverage the inherent reasoning capabilities of LLMs. To address this shortcoming, SLIM [87] has enabled LLMs to generate CoT data [93] as additional input for downstream models. Since the content generated by LLMs is not optimized with recommendation task, this kind of information may not necessarily benefit traditional models. To address this, SeRALM [67] considered optimizing the outputs of LLMs using recommendation loss by alignment training. DLLM2Rec [18] designs a distillation algorithm to effectively transfer the knowledge of LLM-based RS to traditional recommendation models.

To the best of our knowledge, this article is the first to explore the application of LLMs in the field of denoising sequential recommendation.

6 Conclusion

In this article, we propose LLM4DSR, a novel method for sequence recommendation denoising using LLMs, which is the first exploration of LLM performance on this task. Leveraging the extensive open knowledge embedded in LLMs, LLM4DSR offers significant advantages over traditional denoising methods. We have designed a self-supervised task that endows LLMs with enhanced denoising capabilities and integrated the Uncertainty Estimation module to improve both the accuracy and flexibility of the denoising process. Furthermore, we utilized the generative capabilities of LLMs to complete the denoised sequences, thereby enriching sequence information and ensuring robust performance in downstream tasks. Comprehensive experiments demonstrate that LLM4DSR outperforms existing state-of-the-art sequence denoising techniques.

Despite the notable performance of LLM4DSR, it incurs higher computational costs compared to traditional methods due to the use of LLMs. A promising direction for future research is to employ techniques such as knowledge distillation to transfer the denoising capabilities of LLMs to smaller models, thereby achieving superior performance without increasing computational overhead.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. arXiv:2303.08774. Retrieved from https://arxiv.org/abs/2303.08774

[2] Keqin Bao, Jizhi Zhang, Wenjie Wang, Yang Zhang, Zhengyi Yang, Yanchen Luo, Chong Chen, Fuli Feng, and Qi Tian. 2025. A bi-step grounding paradigm for large language models in recommendation systems. *ACM Transactions on Recommender Systems* 3, 4 (2025), 1–27.

[3] Keqin Bao, Jizhi Zhang, Yang Zhang, Xinyue Huo, Chong Chen, and Fuli Feng. 2024. Decoding matters: Addressing amplification bias and homogeneity issue in recommendations for large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, 10540–10552.

[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*, 1007–1014.

[5] Lochan Basyal and Mihir Sanghvi. 2023. Text summarization using large language models: A comparative study of MPT-7B-instruct, Falcon-7b-instruct, and OpenAI Chat-GPT models. arXiv:2310.10449. Retrieved from https://arxiv.org/abs/2310.10449

[6] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *Journal of Machine Learning Research* 3 (Feb. 2003), 1137–1155.

[7] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D. Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. In *Proceedings of the Advances in Neural Information Processing Systems*, Vol. 33, 1877–1901.

[8] Yuwei Cao, Nikhil Mehta, Xinyang Yi, Raghunandan Keshavan, Lukasz Heldt, Lichan Hong, Ed. H. Chi, and Maheswaran Sathiamoorthy. 2024. Aligning large language models with recommendation knowledge. arXiv:2404.00245. Retrieved from https://arxiv.org/abs/2404.00245

[9] Jianxin Chang, Chen Gao, Yu Zheng, Yiqun Hui, Yanan Niu, Yang Song, Depeng Jin, and Yong Li. 2021. Sequential recommendation with graph neural networks. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 378–387.

[10] Huiyuan Chen, Yusan Lin, Menghai Pan, Lan Wang, Chin-Chia Michael Yeh, Xiaoting Li, Yan Zheng, Fei Wang, and Hao Yang. 2022. Denoising self-attentive sequential recommendation. In *Proceedings of the 16th ACM Conference on Recommender Systems*, 92–101.

[11] Jiawei Chen, Hande Dong, Yang Qiu, Xiangnan He, Xin Xin, Liang Chen, Guli Lin, and Keping Yang. 2021. AutoDebias: Learning to debias for recommendation. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 21–30.

[12] Jiawei Chen, Hande Dong, Xiang Wang, Fuli Feng, Meng Wang, and Xiangnan He. 2023. Bias and debias in recommender system: A survey and future directions. *ACM Transactions on Information Systems* 41, 3 (2023), 1–39.

[13] Jiawei Chen, Junkang Wu, Jiancan Wu, Xuezhi Cao, Sheng Zhou, and Xiangnan He. 2023. Adap-τ: Adaptively modulating embedding magnitude for recommendation. In *Proceedings of the ACM Web Conference 2023*, 1085–1096.

[14] Sirui Chen, Jiawei Chen, Sheng Zhou, Bohao Wang, Shen Han, Chanfei Su, Yuqing Yuan, and Can Wang.2024. SIGformer: Sign-aware graph transformer for recommendation. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1274–1284.

[15] Yuxin Chen, Junfei Tan, An Zhang, Zhengyi Yang, Leheng Sheng, Enzhi Zhang, Xiang Wang, and Tat-Seng Chua. 2024. On softmax direct preference optimization for recommendation. In *Proceedings of the Advances in Neural Information Processing Systems*, Vol. 37, 27463–27489.

[16] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2024. Scaling instruction-finetuned language models. *Journal of Machine Learning Research* 25, 70 (2024), 1–53.

[17] Yu Cui, Feng Liu, Jiawei Chen, Xingyu Lou, Changwang Zhang, Jun Wang, Yuegang Sun, Xiaohu Yang, and Can Wang. 2025. Field matters: A lightweight LLM-enhanced method for CTR prediction. arXiv:2505.14057. Retrieved from https://arxiv.org/abs/2505.14057

[18] Yu Cui, Feng Liu, Pengbo Wang, Bohao Wang, Heng Tang, Yi Wan, Jun Wang, and Jiawei Chen. 2024. Distillation matters: Empowering sequential recommenders to match the performance of large language models. In *Proceedings of the 18th ACM Conference on Recommender Systems*, 507–517.

[19] Yashar Deldjoo, Zhankui He, Julian McAuley, Anton Korikov, Scott Sanner, Arnau Ramisa, René Vidal, Maheswaran Sathiamoorthy, Atoosa Kasirzadeh, and Silvia Milano. 2024. A review of modern recommender systems using generative models (Gen-RecSys). In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 6448–6458.

[20] Yashar Deldjoo, Zhankui He, Julian McAuley, Anton Korikov, Scott Sanner, Arnau Ramisa, Rene Vidal, Maheswaran Sathiamoorthy, Atoosa Kasrizadeh, Silvia Milano, et al. 2024. Recommendation with generative models. arXiv:2409.15173. Retrieved from https://arxiv.org/abs/2409.15173

[21] Hande Dong, Jiawei Chen, Fuli Feng, Xiangnan He, Shuxian Bi, Zhaolin Ding, and Peng Cui. 2021. On the equivalence of decoupled graph convolution network and label propagation. In *Proceedings of the Web Conference 2021*, 3651–3662.

[22] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The Llama 3 herd of models. arXiv:2407.21783. Retrieved from https://arxiv.org/abs/2407.21783

[23] Ziwei Fan, Ke Xu, Zhang Dong, Hao Peng, Jiawei Zhang, and Philip S. Yu. 2023. Graph collaborative signals denoising and augmentation for recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2037–2041.

[24] Hui Fang, Danning Zhang, Yiheng Shu, and Guibing Guo. 2020. Deep learning for sequential recommendation: Algorithms, influential factors, and evaluations. *ACM Transactions on Information Systems* 39, 1 (2020), 1–42.

[25] Chongming Gao, Kexin Huang, Jiawei Chen, Yuan Zhang, Biao Li, Peng Jiang, Shiqi Wang, Zhong Zhang, and Xiangnan He. 2023. Alleviating Matthew effect of offline reinforcement learning in interactive recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 238–248.

[26] Yunjun Gao, Yuntao Du, Yujia Hu, Lu Chen, Xinjun Zhu, Ziquan Fang, and Baihua Zheng. 2022. Self-guided learning to denoise for robust recommendation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1412–1422.

[27] Yunfan Gao, Tao Sheng, Youlin Xiang, Yun Xiong, Haofen Wang, and Jiawei Zhang. 2023. Chat-Rec: Towards interactive and explainable LLMs-augmented recommender system. arXiv:2303.14524. Retrieved from https://arxiv.org/abs/2303.14524

[28] Zhaolin Gao, Joyce Zhou, Yijia Dai, and Thorsten Joachims. 2024. End-to-end training for recommendation with language-based user profiles. arXiv:2410.18870. Retrieved from https://arxiv.org/abs/2410.18870

[29] Binzong Geng, Zhaoxin Huan, Xiaolu Zhang, Yong He, Liang Zhang, Fajie Yuan, Jun Zhou, and Linjian Mo. 2024. Breaking the length barrier: LLM-enhanced CTR prediction in long textual user behaviors. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2311–2315.

[30] Shijie Geng, Shuchang Liu, Zuohui Fu, Yingqiang Ge, and Yongfeng Zhang. 2022. Recommendation as language processing (RLP): A unified pretrain, personalized prompt & predict paradigm (p5). In *Proceedings of the 16th ACM Conference on Recommender Systems*, 299–315.

[31] Yongqiang Han, Hao Wang, Kefan Wang, Likang Wu, Zhi Li, Wei Guo, Yong Liu, Defu Lian, and Enhong Chen. 2024. END4Rec: Efficient noise-decoupling for multi-behavior sequential recommendation. arXiv:2403.17603. Retrieved from https://arxiv.org/abs/2403.17603

[32] 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*, 1096–1102.

[33] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2015. Session-based recommendations with recurrent neural networks. arXiv:1511.06939. Retrieved from https://arxiv.org/abs/1511.06939

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

[35] 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*, 585–593.

[36] Yupeng Hou, Junjie Zhang, Zihan Lin, Hongyu Lu, Ruobing Xie, Julian McAuley, and Wayne Xin Zhao. 2024. Large language models are zero-shot rankers for recommender systems. In *Proceedings of the European Conference on Information Retrieval*. Springer, 364–381.

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

[38] Zhiyu Hu, Yang Zhang, Minghao Xiao, Wenjie Wang, Fuli Feng, and Xiangnan He. 2025. Exact and efficient unlearning for large language model-based recommendation. *IEEE Transactions on Knowledge and Data Engineering* (2025).

[39] Siqing Huo, Negar Arabzadeh, and Charles Clarke. 2023. Retrieving supporting evidence for generative question answering. In *Proceedings of the Annual International ACM SIGIR Conference on Research and Development in Information Retrieval in the Asia Pacific Region*, 11–20.

[40] Aryan Jadon and Avinash Patil. 2024. A comprehensive survey of evaluation techniques for recommendation systems. In *Proceedings of the International Conference on Computation of Artificial Intelligence & Machine Learning*. Springer, 281–304.

[41] Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *ACM Computing Surveys* 55, 12 (2023), 1–38.

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

[43] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. arXiv:2001.08361. Retrieved from https://arxiv.org/abs/2001.08361

[44] Sein Kim, Hongseok Kang, Seungyoon Choi, Donghyun Kim, Minchul Yang, and Chanyoung Park. 2024. Large language models meet collaborative filtering: An efficient all-round LLM-based recommender system. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 1395–1406.

[45] Dong-Ho Lee, Adam Kraft, Long Jin, Nikhil Mehta, Taibai Xu, Lichan Hong, Ed. H. Chi, and Xinyang Yi. 2024. STAR: A simple training-free approach for recommendations using large language models. arXiv:2410.16458. Retrieved from https://arxiv.org/abs/2410.16458

[46] 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:2304.03879. Retrieved from https://arxiv.org/abs/2304.03879

[47] Lei Li, Yongfeng Zhang, and Li Chen. 2023. Prompt distillation for efficient LLM-based recommendation. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, 1348–1357.

[48] Lei Li, Yongfeng Zhang, Dugang Liu, and Li Chen. 2023. Large language models for generative recommendation: A survey and visionary discussions. arXiv:2309.01157. Retrieved from https://arxiv.org/abs/2309.01157

[49] Weitao Li, Junkai Li, Weizhi Ma, and Yang Liu. 2024. Citation-enhanced generation for LLM-based chatbot. arXiv:2402.16063. Retrieved from https://arxiv.org/abs/2402.16063

[50] Xinyi Li, Yongfeng Zhang, and Edward C. Malthouse. 2023. Exploring fine-tuning ChatGPT for news recommendation. arXiv:2311.05850. Retrieved from https://arxiv.org/abs/2311.05850

[51] Yaoyiran Li, Xiang Zhai, Moustafa Alzantot, Keyi Yu, Ivan Vulić, Anna Korhonen, and Mohamed Hammad. 2024. CalRec: Contrastive alignment of generative LLMs for sequential recommendation. In *Proceedings of the 18th ACM Conference on Recommender Systems*, 422–432.

[52] Zelong Li, Jianchao Ji, Yingqiang Ge, Wenyue Hua, and Yongfeng Zhang. 2024. PAP-REC: Personalized automatic prompt for recommendation language model. arXiv:2402.00284. Retrieved from https://arxiv.org/abs/2402.00284

[53] Jiayi Liao, Sihang Li, Zhengyi Yang, Jiancan Wu, Yancheng Yuan, Xiang Wang, and Xiangnan He. 2024. LLaRA: Large language-recommendation assistant. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1785–1795.

[54] Jianghao Lin, Rong Shan, Chenxu Zhu, Kounianhua Du, Bo Chen, Shigang Quan, Ruiming Tang, Yong Yu, and Weinan Zhang. 2024. ReLLa: Retrieval-enhanced large language models for lifelong sequential behavior comprehension in recommendation. In *Proceedings of the ACM Web Conference 2024*, 3497–3508.

[55] Siyi Lin, Chongming Gao, Jiawei Chen, Sheng Zhou, Binbin Hu, Yan Feng, Chun Chen, and Can Wang. 2025. How do recommendation models amplify popularity bias? An analysis from the spectral perspective. In *Proceedings of the 18th ACM International Conference on Web Search and Data Mining*, 659–668.

[56] Xinyu Lin, Wenjie Wang, Yongqi Li, Fuli Feng, See-Kiong Ng, and Tat-Seng Chua. 2024. Bridging items and language: A transition paradigm for large language model-based recommendation. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 1816–1826.

[57] Xinyu Lin, Wenjie Wang, Yongqi Li, Shuo Yang, Fuli Feng, Yinwei Wei, and Tat-Seng Chua. 2024. Data-efficient fine-tuning for LLM-based recommendation. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 365–374.

[58] Yujie Lin, Chenyang Wang, Zhumin Chen, Zhaochun Ren, Xin Xin, Qiang Yan, Maarten de, Rijke Xiuzhen, and Cheng Pengjie Ren. 2023. A self-correcting sequential recommender. In *Proceedings of the ACM Web Conference 2023*, 1283–1293.

[59] Qijiong Liu, Nuo Chen, Tetsuya Sakai, and Xiao-Ming Wu. 2024. Once: Boosting content-based recommendation with both open-and closed-source large language models. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, 452–461.

[60] Andriy Mnih and Russ R. Salakhutdinov. 2007. Probabilistic matrix factorization. In *Proceedings of the Advances in Neural Information Processing Systems*, Vol. 20.

[61] Jiarui Qin, Weiwen Liu, Weinan Zhang, and Yong Yu. 2025. D2K: Turning historical data into retrievable knowledge for recommender systems. In *Proceedings of the ACM on Web Conference 2025*, 472–482.

[62] Ruihong Qiu, Zi Huang, Hongzhi Yin, and Zijian Wang. 2022. Contrastive learning for representation degeneration problem in sequential recommendation. In *Proceedings of the 15th ACM International Conference on Web Search and Data Mining*, 813–823.

[63] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D. Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. In *Proceedings of the Advances in Neural Information Processing Systems*, Vol. 36, 53728–53741.

[64] Hamed Rahimian and Sanjay Mehrotra. 2019. Distributionally robust optimization: A review. arXiv:1908.05659. Retrieved from https://arxiv.org/abs/1908.05659

[65] Shashank Rajput, Nikhil Mehta, Anima Singh, Raghunandan Hulikal Keshavan, Trung Vu, Lukasz Heldt, Lichan Hong, Yi Tay, Vinh Tran, Jonah Samost, et al. 2023. Recommender systems with generative retrieval. In *Proceedings of the Advances in Neural Information Processing Systems*, Vol. 36, 10299–10315.

[66] Xubin Ren, Wei Wei, Lianghao Xia, Lixin Su, Suqi Cheng, Junfeng Wang, Dawei Yin, and Chao Huang. 2024. Representation learning with large language models for recommendation. In *Proceedings of the ACM on Web Conference 2024*, 3464–3475.

[67] Yankun Ren, Zhongde Chen, Xinxing Yang, Longfei Li, Cong Jiang, Lei Cheng, Bo Zhang, Linjian Mo, and Jun Zhou. 2024. Enhancing sequential recommenders with augmented knowledge from aligned large language models. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 345–354.

[68] Wentao Shi, Xiangnan He, Yang Zhang, Chongming Gao, Xinyue Li, Jizhi Zhang, Qifan Wang, and Fuli Feng. 2024. Enhancing long-term recommendation with bi-level learnable large language model planning. arXiv:2403.00843. Retrieved from https://arxiv.org/abs/2403.00843

[69] Jun Shu, Qi Xie, Lixuan Yi, Qian Zhao, Sanping Zhou, Zongben Xu, and Deyu Meng. 2019. Meta-weight-net: Learning an explicit mapping for sample weighting. In *Proceedings of the Advances in Neural Information Processing Systems*, Vol. 32.

[70] Tianrui Song, Wenshuo Chao, and Hao Liu. 2024. Large language model enhanced hard sample identification for denoising recommendation. arXiv:2409.10343. Retrieved from https://arxiv.org/abs/2409.10343

[71] 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*, 1441–1450.

[72] Yatong Sun, Bin Wang, Zhu Sun, and Xiaochun Yang. 2021. Does every data instance matter? Enhancing sequential recommendation by eliminating unreliable data. In *Proceedings of the 30th International Joint Conference on Artificial Intelligence (IJCAI)*, 1579–1585.

[73] Heng Tang, Feng Liu, Xinbo Chen, Jiawei Chen, Bohao Wang, Changwang Zhang, Jun Wang, Yuegang Sun, Bingde Hu, and Can Wang. 2025. Bridging the gap: Self-optimized fine-tuning for LLM-based recommender systems. arXiv:2505.20771. Retrieved from https://arxiv.org/abs/2505.20771

[74] Jiaxi Tang and Ke Wang. 2018. Personalized top-n sequential recommendation via convolutional sequence embedding. In *Proceedings of the 11th ACM International Conference on Web Search and Data Mining*, 565–573.

[75] Guy Tennenholtz, Yinlam Chow, Chih-Wei Hsu, Jihwan Jeong, Lior Shani, Azamat Tulepbergenov, Deepak Ramachandran, Martin Mladenov, and Craig Boutilier. 2023. Demystifying embedding spaces using large language models. arXiv:2310.04475. Retrieved from https://arxiv.org/abs/2310.04475

[76] Changxin Tian, Yuexiang Xie, Yaliang Li, Nan Yang, and Wayne Xin Zhao. 2022. Learning to denoise unreliable interactions for graph collaborative filtering. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 122–132.

[77] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the Advances in Neural Information Processing Systems*, Vol. 30.

[78] Bohao Wang, Jiawei Chen, Changdong Li, Sheng Zhou, Qihao Shi, Yang Gao, Yan Feng, Chun Chen, and Can Wang. 2024. Distributionally robust graph-based recommendation system. In *Proceedings of the ACM Web Conference 2024*, 3777–3788.

[79] Bohao Wang, Feng Liu, Jiawei Chen, Xingyu Lou, Changwang Zhang, Jun Wang, Yuegang Sun, Yan Feng, Chun Chen, and Can Wang. 2025. MSL: Not all tokens are what you need for tuning LLM as a recommender. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1912–1922.

[80] Hangyu Wang, Jianghao Lin, Bo Chen, Yang Yang, Ruiming Tang, Weinan Zhang, and Yong Yu. 2025. Towards efficient and effective unlearning of large language models for recommendation. *Frontiers of Computer Science* 19, 3 (2025), 193327.

[81] Hangyu Wang, Jianghao Lin, Xiangyang Li, Bo Chen, Chenxu Zhu, Ruiming Tang, Weinan Zhang, and Yong Yu. 2023. ALT: Towards fine-grained alignment between language and CTR models for click-through rate prediction. arXiv:2310.19453. Retrieved from https://arxiv.org/abs/2310.19453

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

[83] Shoujin Wang, Liang Hu, Yan Wang, Longbing Cao, Quan Z. Sheng, and Mehmet Orgun. 2019. Sequential recommender systems: Challenges, progress and prospects. arXiv:2001.04830. Retrieved from https://arxiv.org/abs/2001.04830

[84] Wenjie Wang, Fuli Feng, Xiangnan He, Liqiang Nie, and Tat-Seng Chua. 2021. Denoising implicit feedback for recommendation. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, 373–381.

[85] Wenjie Wang, Fuli Feng, Xiangnan He, Hanwang Zhang, and Tat-Seng Chua. 2021. Clicks can be cheating: Counterfactual recommendation for mitigating clickbait issue. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1288–1297.

[86] Yidan Wang, Zhaochun Ren, Weiwei Sun, Jiyuan Yang, Zhixiang Liang, Xin Chen, Ruobing Xie, Su Yan, Xu Zhang, Pengjie Ren, et al. 2024. Enhanced generative recommendation via content and collaboration integration. arXiv:2403.18480. Retrieved from https://arxiv.org/abs/2403.18480

[87] Yuling Wang, Changxin Tian, Binbin Hu, Yanhua Yu, Ziqi Liu, Zhiqiang Zhang, Jun Zhou, Liang Pang, and Xiao Wang. 2024. Can small language models be good reasoners for sequential recommendation? In *Proceedings of the ACM on Web Conference 2024*, 3876–3887.

[88] Yu Wang, Xin Xin, Zaiqiao Meng, Joemon M. Jose, Fuli Feng, and Xiangnan He. 2022. Learning robust recommenders through cross-model agreement. In *Proceedings of the ACM Web Conference 2022*, 2015–2025.

[89] Zongwei Wang, Min Gao, Wentao Li, Junliang Yu, Linxin Guo, and Hongzhi Yin. 2023. Efficient bi-level optimization for recommendation denoising. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2502–2511.

[90] Zhefan Wang, Weizhi Ma, and Min Zhang. 2024. To recommend or not: Recommendability identification in conversations with pre-trained language models. In *Proceedings of the International Conference on Database Systems for Advanced Applications*. Springer, 19–35.

[91] Jiateng Wei, Quan Lu, Ning Jiang, Siqi Li, Jingyang Xiang, Jun Chen, and Yong Liu. 2024. Structured optimal brain pruning for large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, 13991–14007.

[92] Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. 2022. Emergent abilities of large language models. arXiv:2206.07682. Retrieved from https://arxiv.org/abs/2206.07682

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

[94] Wei Wei, Xubin Ren, Jiabin Tang, Qinyong Wang, Lixin Su, Suqi Cheng, Junfeng Wang, Dawei Yin, and Chao Huang. 2024. LLMRec: Large language models with graph augmentation for recommendation. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, 806–815.

[95] Junda Wu, Cheng-Chun Chang, Tong Yu, Zhankui He, Jianing Wang, Yupeng Hou, and Julian McAuley. 2024. Coral: Collaborative retrieval-augmented large language models improve long-tail recommendation. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 3391–3401.

[96] Junkang Wu, Jiawei Chen, Jiancan Wu, Wentao Shi, Xiang Wang, and Xiangnan He. 2023. Understanding contrastive learning via distributionally robust optimization. In *Proceedings of the Advances in Neural Information Processing Systems*, Vol. 36, 23297–23320.

[97] Likang Wu, Zhi Zheng, Zhaopeng Qiu, Hao Wang, Hongchao Gu, Tingjia Shen, Chuan Qin, Chen Zhu, Hengshu Zhu, Qi Liu, et al. 2024. A survey on large language models for recommendation. *World Wide Web* 27, 5 (2024), 60.

[98] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S. Yu Philip. 2020. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems* 32, 1 (2020), 4–24.

[99] Yunjia Xi, Weiwen Liu, Jianghao Lin, Xiaoling Cai, Hong Zhu, Jieming Zhu, Bo Chen, Ruiming Tang, Weinan Zhang, and Yong Yu. 2024. Towards open-world recommendation with knowledge augmentation from large language models. In *Proceedings of the 18th ACM Conference on Recommender Systems*, 12–22.

[100] Jingyang Xiang and Saiqian Zhang. 2024. DFRot: Achieving outlier-free and massive activation-free for rotated LLMs with refined rotation. arXiv:2412.00648. Retrieved from https://arxiv.org/abs/2412.00648

[101] Xu Xie, Fei Sun, Zhaoyang Liu, Shiwen Wu, Jinyang Gao, Jiandong Zhang, Bolin Ding, and Bin Cui. 2022. Contrastive learning for sequential recommendation. In *Proceedings of the 2022 IEEE 38th International Conference on Data Engineering (ICDE)*. IEEE, 1259–1273.

[102] Wentao Xu, Qianqian Xie, Shuo Yang, Jiangxia Cao, and Shuchao Pang. 2024. Enhancing content-based recommendation via large language model. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, 4153–4157.

[103] Shenghao Yang, Weizhi Ma, Peijie Sun, Min Zhang, Qingyao Ai, Yiqun Liu, and Mingchen Cai. 2024. Common sense enhanced knowledge-based recommendation with large language model. In *Proceedings of the International Conference on Database Systems for Advanced Applications*. Springer, 381–390.

[104] Weiqin Yang, Jiawei Chen, Xin Xin, Sheng Zhou, Binbin Hu, Yan Feng, Chun Chen, and Can Wang. 2024. PSL: Rethinking and improving softmax loss from pairwise perspective for recommendation. In *Proceedings of the Advances in Neural Information Processing Systems*, Vol. 37, 120974–121006.

[105] Yuhao Yang, Chao Huang, Lianghao Xia, Chunzhen Huang, Da Luo, and Kangyi Lin. 2023. Debiased contrastive learning for sequential recommendation. In *Proceedings of the ACM Web Conference 2023*, 1063–1073.

[106] Zhengyi Yang, Xiangnan He, Jizhi Zhang, Jiancan Wu, Xin Xin, Jiawei Chen, and Xiang Wang. 2023. A generic learning framework for sequential recommendation with distribution shifts. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 331–340.

[107] Jiahao Yuan, Zihan Song, Mingyou Sun, Xiaoling Wang, and Wayne Xin Zhao. 2021. Dual sparse attention network for session-based recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35, 4635–4643.

[108] Chi Zhang, Yantong Du, Xiangyu Zhao, Qilong Han, Rui Chen, and Li Li. 2022. Hierarchical item inconsistency signal learning for sequence denoising in sequential recommendation. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, 2508–2518.

[109] Chi Zhang, Qilong Han, Rui Chen, Xiangyu Zhao, Peng Tang, and Hongtao Song. 2024. SSDRec: Self-augmented sequence denoising for sequential recommendation. In *Proceedings of the 2024 IEEE 40th International Conference on Data Engineering (ICDE)*. IEEE, 803–815.

[110] Junjie Zhang, Ruobing Xie, Yupeng Hou, Wayne Xin Zhao, Leyu Lin, and Ji-Rong Wen. 2023. Recommendation as instruction following: A large language model empowered recommendation approach. arXiv:2305.07001. Retrieved from https://arxiv.org/abs/2305.07001

[111] Shengyu Zhang, Linfeng Dong, Xiaoya Li, Sen Zhang, Xiaofei Sun, Shuhe Wang, Jiwei Li, Runyi Hu, Tianwei Zhang, Fei Wu, et al. 2023. Instruction tuning for large language models: A survey. arXiv:2308.10792. Retrieved from https://arxiv.org/abs/2308.10792

[112] Wenlin Zhang, Xiangyang Li, Yuhao Wang, Kuicai Dong, Yichao Wang, Xinyi Dai, Xiangyu Zhao, Huifeng Guo, and Ruiming Tang. 2024. Tired of plugins? Large language models can be end-to-end recommenders. arXiv:2404.00702. Retrieved from https://arxiv.org/abs/2404.00702

[113] Yabin Zhang, Zhenlei Wang, Wenhui Yu, Lantao Hu, Peng Jiang, Kun Gai, and Xu Chen. 2024. Soft contrastive sequential recommendation. *ACM Transactions on Information Systems* 42, 6, Article 154 (2024), 28 pages.

[114] Zihao Zhao, Jiawei Chen, Sheng Zhou, Xiangnan He, Xuezhi Cao, Fuzheng Zhang, and Wei Wu. 2022. Popularity bias is not always evil: Disentangling benign and harmful bias for recommendation. *IEEE Transactions on Knowledge and Data Engineering* 35, 10 (2022), 9920–9931.

[115] Kun Zhou, Hui Yu, Wayne Xin Zhao, and Ji-Rong Wen. 2022. Filter-enhanced MLP is all you need for sequential recommendation. In *Proceedings of the ACM Web Conference 2022*, 2388–2399.

[116] Wenhao Zhu, Hongyi Liu, Qingxiu Dong, Jingjing Xu, Shujian Huang, Lingpeng Kong, Jiajun Chen, and Lei Li. 2023. Multilingual machine translation with large language models: Empirical results and analysis. arXiv:2304.04675. Retrieved from https://arxiv.org/abs/2304.04675

[117] Yaochen Zhu, Liang Wu, Qi Guo, Liangjie Hong, and Jundong Li. 2024. Collaborative large language model for recommender systems. In *Proceedings of the ACM on Web Conference 2024*, 3162–3172.

Received 18 November 2024; revised 17 March 2025; accepted 3 August 2025