Benefit from Rich: Tackling Search Interaction Sparsity in Search Enhanced Recommendation

Teng Shi
Gaoling School of Artificial Intelligence
Renmin University of China
Beijing, China
shiteng@ruc.edu.cn

Weijie Yu*
School of Information Technology and Management
University of International Business and Economics
Beijing, China
yu@uibe.edu.cn

Xiao Zhang
Gaoling School of Artificial Intelligence
Renmin University of China
Beijing, China
zhangx89@ruc.edu.cn

Ming He
AI Lab at Lenovo Research
Beijing, China
heming01@foxmail.com

Jianping Fan
AI Lab at Lenovo Research
Beijing, China
jfan1@lenovo.com

Jun Xu
Gaoling School of Artificial Intelligence
Renmin University of China
Beijing, China
junxu@ruc.edu.cn

Abstract

In modern online platforms, search and recommendation (**S&R**) often coexist, offering opportunities for performance improvement through search-enhanced approaches. Existing studies show that incorporating search signals boosts recommendation performance. However, the effectiveness of these methods relies heavily on rich search interactions. They primarily benefit a small subset of users with abundant search behavior, while offering limited improvements for the majority of users who exhibit only sparse search activity. To address the problem of sparse search data in search-enhanced recommendation, we face two key challenges : (1) how to learn useful search features for users with sparse search interactions, and (2) how to design effective training objectives under sparse conditions. Our idea is to leverage the features of users with rich search interactions to enhance those of users with sparse search interactions. Based on this idea, we propose **GSERec**, a method that utilizes message passing on the User-Code **G**raphs to alleviate data sparsity in **S**earch-**E**nhanced **Rec**ommendation. Specifically, we utilize Large Language Models (LLMs) with vector quantization to generate discrete codes, which connect similar users and thereby construct the graph. Through message passing on this graph, embeddings of users with rich search data are propagated to enhance the embeddings of users with sparse interactions. To further ensure that the message passing captures meaningful information from truly similar users, we introduce a contrastive loss to better model user similarities. The enhanced user representations are then integrated into downstream search-enhanced recommendation models. Experiments on three real-world datasets show that GSERec consistently outperforms baselines, especially for users with sparse search behaviors.

CCS Concepts

• **Information systems → Recommender systems**.

Keywords

Recommendation; Search; Large Language Model

ACM Reference Format:
Teng Shi, Weijie Yu, Xiao Zhang, Ming He, Jianping Fan, and Jun Xu. 2025. Benefit from Rich: Tackling Search Interaction Sparsity in Search Enhanced Recommendation. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management (CIKM '25), November 10–14, 2025, Seoul, Republic of Korea.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3746252.3761314

1 Introduction

Nowadays, many commercial apps offer both search and recommendation (**S&R**) services to meet diverse user needs, such as e-commerce platforms (e.g., Taobao) and short video platforms (e.g., TikTok). In these scenarios, user S&R behaviors frequently influence each other, providing an opportunity to enhance recommendation through search, allowing us to better model users with search behavior.

Existing search-enhanced recommendation methods primarily enhance the model from two perspectives: (1) Feature-level enhancement: These methods introduce additional search-related features on top of traditional recommendation models. For example, many approaches incorporate users' search history into the model input [14, 37, 38, 46]. (2) Loss-level enhancement: Many models [34, 55, 56] adopt joint training of S&R by introducing combined loss functions to simultaneously optimize both objectives, aiming to learn better user and item representations.

While these methods have achieved promising results, their enhancement remains constrained by data sparsity. For example, in search-enhanced recommendation, users with limited search interactions contribute minimally in two ways: (1) the search history

[*] Weijie Yu is the corresponding author. Work partially done at Engineering Research Center of Next-Generation Intelligent Search and Recommendation, Ministry of Education.

CIKM '25, Seoul, Republic of Korea

ACM ISBN 979-8-4007-2040-6/2025/11
https://doi.org/10.1145/3746252.3761314

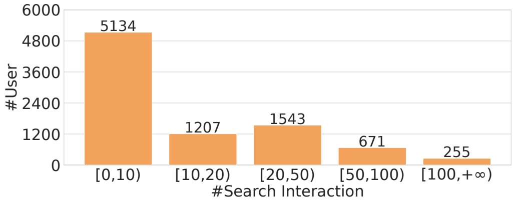


Figure 1: User count statistics across different groups on the Qilin [4] dataset, where users are grouped by the number of their search interactions. We observe that users with rich search interactions are few, while the majority are users with sparse interactions.

features incorporated into the model are scarce; and (2) the loss derived from their sparse search data has limited effect on optimizing representations during joint training. As a result, existing models yield greater improvements for users with rich search interactions, while offering limited benefits for users with sparse search behavior. However, as shown in Figure 1, users with rich search interactions are few, while the majority of users exhibit sparse search behavior. Figure 2 further reveals that the state-of-the-art search-enhanced recommendation model UniSAR [34] achieves greater improvements primarily for users with richer search interactions.

Alleviating the issue of sparse user search data in search-enhanced recommendation presents two main challenges: (1) How to effectively enhance features for users with sparse data. For example, in search-enhanced recommendation, how to derive informative search features for users with limited search interactions; (2) How to design improved loss functions during training to ensure that the representations of users with sparse data can still be well optimized. Our key idea is that not all users suffer from search sparsity—some have rich search interactions. We enhance the features of users with sparse search behaviors by propagating information from similar users with rich search histories.

Based on this idea, to address the above issues, we propose a method called **GSERec**, which performs message passing on the User-Code **G**raphs to enhance the representations of users with sparse search interactions, thereby alleviating data sparsity in **S**earch-**E**nhanced **Rec**ommendation. Specifically, we first utilize a Large Language Model (LLM) [70] to summarize users' S&R preferences. These preferences are then encoded using an embedding model and transformed into discrete codes via vector quantization [29, 57]. Next, we connect each user to their corresponding codes, and the shared codes link similar users together, forming the graph. Subsequently, by performing message passing on this graph, the embeddings of users with rich search interactions can be propagated to enhance the embeddings of users with sparse search interactions. Furthermore, to ensure that message passing captures useful information from similar users, we design contrastive learning [6, 11] objectives to help user embeddings better capture the similarity between users. Finally, the enhanced user representations are integrated with the S&R history features in downstream search-enhanced recommendation models for the final recommendation task.

The major contributions of the paper are summarized as follows:

- We identify a key limitation of existing search-enhanced recommendation methods: their performance is limited for users with sparse search histories. This highlights the challenge of extracting informative search representations and designing effective loss functions under data sparsity.

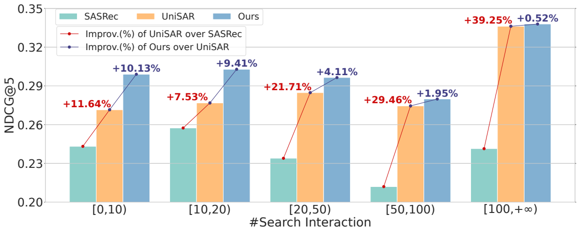


Figure 2: Relative improvements of the state-of-the-art model UniSAR [34] over the traditional recommendation model SASRec [16] across different user groups on the Qilin dataset, along with the improvements of our model over UniSAR. User groups are defined using the same strategy as in Figure 1. UniSAR shows greater improvements for users with more search interactions, while our model effectively alleviates data sparsity and achieves larger gains for users with fewer search interactions.

- We propose GSERec, which performs message passing on the User-Code graphs to enhance the embeddings of users with sparse search interactions, thereby alleviating the data sparsity problem in search-enhanced recommendation. Furthermore, we design contrastive learning objectives to better model user similarity, thereby enabling the message passing process to extract more informative signals.
- Experimental results on three datasets validate the effectiveness of GSERec: it not only outperforms traditional recommendation methods but also surpasses existing search-enhanced recommendation approaches. Moreover, as shown in Figure 2, GSERec achieves notably larger gains for users with sparse search interactions.

2 Related Work

Recommendation. Recommender systems [8, 32, 58–61, 66] model user preferences to suggest relevant items. Sequential recommendation [15, 41, 54, 62, 72–74] captures interests from historical interactions, with Transformer-based models [16, 39, 44] and contrastive learning [6, 11, 51] enhancing sequential modeling. Graph-based methods [3, 18, 42, 47, 53] exploit user–item relations, where LightGCN [13] aggregates neighborhood signals and SGL [47], SimGCL [53] apply contrastive learning. Recently, LLM-enhanced recommenders [7, 19, 20, 26, 30] have emerged, such as KAR [48] leveraging LLM-derived reasoning vectors and LLM-ESR [20] combining LLM semantics with collaborative signals. This work instead boosts recommendation performance by incorporating search data.

Search Enhanced Recommendation. Recently, leveraging search [24, 25, 27, 28, 31, 35, 40, 63–65] data to enhance recommendation performance has attracted growing interest [23, 33, 34, 36, 37, 45, 50, 55, 67–69]. Existing approaches include joint model training for S&R [55, 56], transformer-based integration of both behaviors [52], contrastive learning to separate similar and dissimilar interests [38], and dual-branch or masked transformer networks for unified encoding [34, 50]. In contrast, we address sparsity in search-enhanced recommendation by enriching sparse user embeddings via message passing.

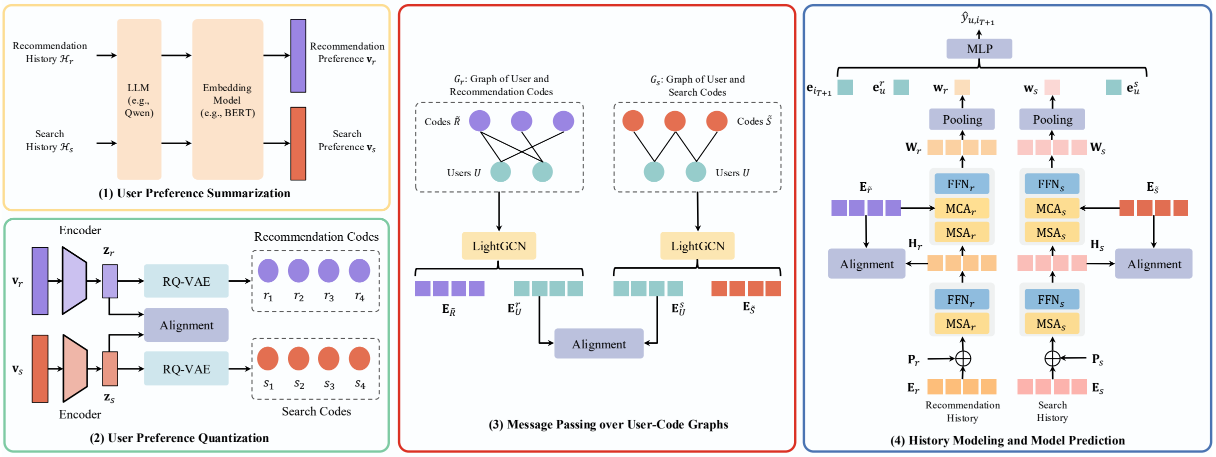


Figure 3: The overall framework of GSERec. The framework consists of two stages: User-Code Graph Construction: (1) User Preference Summarization; (2) User Preference Quantization. Search Enhanced Recommendation Modeling: (3) Message Passing over User-Code Graph; (4) History Modeling and Prediction.

3 Problem Formulation

We denote the sets of users, items, and queries as $\mathcal{U}$, $\mathcal{I}$, and $\mathcal{Q}$, respectively. Each user $u \in \mathcal{U}$ has a chronologically ordered recommendation history $\mathcal{H}_r = \{i_1, i_2, \ldots, i_{N_r}\}$ and a search history $\mathcal{H}_s = \{(q_1, \mathcal{I}_{q_1}), (q_2, \mathcal{I}_{q_2}), \ldots, (q_{N_s}, \mathcal{I}_{q_{N_s}})\}$, where N_r and N_s denote the lengths of the user's recommendation and search histories, respectively. The total number of user interactions is denoted as $T = N_r + N_s$. Here, $i_k \in \mathcal{I}$ is the k-th item the user interacted with, $q_k \in \mathcal{Q}$ is the k-th query issued by the user, and $\mathcal{I}_{q_k} = \{i_1, i_2, \ldots, i_{N_{q_k}}\}$ is the set of N_{q_k} items clicked by the user after searching query q_k. Our goal is to train a recommendation model Θ that predicts the next item i_{T+1} based on the user's recommendation history $\mathcal{H}_r$ and search history $\mathcal{H}_s$.

Existing search-enhanced recommendation methods tend to yield greater improvements for users with rich search interactions, while offering limited benefits for those with sparse search behavior. To address this imbalance, we aim to enhance the representations of users with sparse search interactions by propagating information from users with richer search histories, thereby alleviating the data sparsity issue in search-enhanced recommendation.

4 Our Approach

This section introduces our method, GSERec, illustrated in Figure 3, which includes two main components: **User-Code Graph Construction** (§ 4.1) and **Search Enhanced Recommendation Modeling** (§ 4.2). **User-Code Graph Construction** includes: (1) User Preference Summarization (§ 4.1.1): summarizes users' S&R preferences using the LLM; (2) User Preference Quantization (§ 4.1.2): discretizes the summarized preferences into codes via vector quantization; (3) Graph Construction (§ 4.1.3): connects similar users through shared codes to form the user-code bipartite graphs. **Search Enhanced Recommendation Modeling** includes: (1) Message Passing over the User-Code Graph (§ 4.2.2): enhances the representations of users with sparse search interactions by propagating information from users with richer interactions; (2) Historical Modeling (§ 4.2.3): integrates the enhanced user representations with their S&R histories for final prediction.

4.1 User-Code Graph Construction

This section introduces the construction of the user-code graph. We first use the LLM to summarize users' S&R preferences. These preferences are then encoded using an embedding model and discretized into codes via vector quantization. Finally, each user is connected to their corresponding codes, and users who share similar codes are linked, forming the user-code graph.

4.1.1 User Preference Summarization. For each user $u \in \mathcal{U}$, we input her search history $\mathcal{H}_s$ and recommendation history $\mathcal{H}_r$ into a LLM to summarize her S&R preferences. The prompts provided to the LLM are as follows:

Search Preference Summarization

Prompt: Please analyze the queries and clicked items in the user's search history, and summarize the user's interest topics, areas of focus, style tendencies, or preference types. Here is the user's search history *{history}*, where each record contains the user's query and the items the user clicked on under that query.

Recommendation Preference Summarization

Prompt: Please analyze the provided user recommendation history and summarize the user's possible interests, style tendencies, and preferred item types. Here is the user's recommendation history *{history}*, where each record represents an item the user has clicked on.

The user's S&R preferences, as summarized by the LLM, are individually encoded using a pretrained embedding model (e.g., BERT [10] or BGE [5, 49]) to obtain dense representations, denoted as $\mathbf{v}_s \in \mathbb{R}^{d_e}$ and $\mathbf{v}_r \in \mathbb{R}^{d_e}$. Here, d_e represents the dimensionality of the embedding model. It is important to note that the embedding model is pretrained and remains frozen during the entire training process.

4.1.2 User Preference Quantization. We discretize the encoded user preferences into codes to facilitate subsequent graph construction. Specifically, we adopt Residual Quantized Variational Autoencoder (RQ-VAE) [29, 57, 71], a widely used vector quantization method.

The user's S&R preferences are first encoded using two separate encoders:

$$\mathbf{z}_s = \text{Encoder}_s(\mathbf{v}_s), \quad \mathbf{z}_r = \text{Encoder}_r(\mathbf{v}_r),$$

where $\mathbf{z}_s, \mathbf{z}_r \in \mathbb{R}^{d_l}$ denote the latent representations of the user's S&R preferences, respectively, and d_l is the dimension of the embedding space. $\text{Encoder}_s(\cdot)$ and $\text{Encoder}_r(\cdot)$ are implemented as multilayer perceptrons (MLPs).

$\mathbf{z}_s$ and $\mathbf{z}_r$ encode the S&R preferences of the same user. To better model the similarity between different users, we align them before quantization. To this end, we apply contrastive learning by minimizing the following InfoNCE [22] loss:

$$\mathcal{L}_{\text{RQ-CL}} = -\Bigg[\log \frac{\exp(\text{sim}(\mathbf{z}_s, \mathbf{z}_r)/\tau_1)}{\sum_{z_r^- \in \mathcal{Z}_r^{\text{neg}}} \exp(\text{sim}(\mathbf{z}_s, \mathbf{z}_r^-)/\tau_1)} + \log \frac{\exp(\text{sim}(\mathbf{z}_s, \mathbf{z}_r)/\tau_1)}{\sum_{z_s^- \in \mathcal{Z}_s^{\text{neg}}} \exp(\text{sim}(\mathbf{z}_s^-, \mathbf{z}_r)/\tau_1)}\Bigg], \tag{1}$$

where $\text{sim}(\cdot)$ denotes a similarity function (e.g., cosine similarity), τ_1 is a learnable temperature coefficient. $\mathcal{Z}_r^{\text{neg}}$ and $\mathcal{Z}_s^{\text{neg}}$ denote the negative samples from other users within the same batch.

Next, $\mathbf{z}_s$ and $\mathbf{z}_r$ are independently discretized into L codes using two separate L-level codebooks. Taking the quantization of the user's search preferences as an example, at each level $l \in \{1, 2, \ldots, L\}$, we define a codebook $\mathcal{CS}_l = \{\mathbf{e}_k\}_{k=1}^{N_c}$, where N_c is the size of each codebook and $\mathbf{e}_k \in \mathbb{R}^{d_l}$ is a learnable code embedding. The residual quantization process is as follows:

$$\begin{cases} s_l = \arg\min_k ||\mathbf{r}_{l-1}^s - \mathbf{e}_k||_2^2, & \mathbf{e}_k \in \mathcal{CS}_l, \\ \mathbf{r}_l^s = \mathbf{r}_{l-1}^s - \mathbf{e}_{s_l}, & \mathbf{r}_0 = \mathbf{z}_s \in \mathbb{R}^{d_l}, \end{cases} \tag{2}$$

where s_l denotes the index of the selected code at level l, and $\mathbf{r}_{l-1}^s$ is the residual from the previous level.

Through the recursive quantization process described in Eq. (2), we obtain the discrete codes $\tilde{s}$ and the quantized embedding $\hat{\mathbf{z}}_s = \sum_{l=1}^{L} \mathbf{e}_{s_l}$ for the user's search preference. Similarly, we can obtain the discrete codes $\tilde{r}$ and the quantized embedding $\hat{\mathbf{z}}_r = \sum_{l=1}^{L} \mathbf{e}_{r_l}$ for the user's recommendation preference. The discrete codes $\tilde{s}$ and $\tilde{r}$ are as follows:

$$\tilde{s} = [s_1, s_2, \ldots, s_L], \quad \tilde{r} = [r_1, r_2, \ldots, r_L] \tag{3}$$

The quantized embeddings $\hat{\mathbf{z}}_s$ and $\hat{\mathbf{z}}_r$ are then passed through two separate decoders to reconstruct the original user S&R preference representations, $\mathbf{v}_s$ and $\mathbf{v}_r$, respectively:

$$\hat{\mathbf{v}}_s = \text{Decoder}_s(\hat{\mathbf{z}}_s), \quad \hat{\mathbf{v}}_r = \text{Decoder}_r(\hat{\mathbf{z}}_r),$$

where $\text{Decoder}_s(\cdot)$ and $\text{Decoder}_r(\cdot)$ denote two MLPs. The reconstruction loss for training the encoders and decoders is calculated as:

$$\mathcal{L}_{\text{Recon}} = ||\mathbf{v}_s - \hat{\mathbf{v}}_s||_2^2 + ||\mathbf{v}_r - \hat{\mathbf{v}}_r||_2^2. \tag{4}$$

To optimize the quantization process, we further introduce the residual quantization loss, which is formulated as:

$$\begin{cases} \mathcal{L}_{\text{RQ}}^s = \sum_{l=1}^{L} ||\text{sg}[\mathbf{r}_{l-1}^s] - \mathbf{e}_{s_l}||_2^2 + ||\mathbf{r}_{l-1}^s - \text{sg}[\mathbf{e}_{s_l}]||_2^2, \\ \mathcal{L}_{\text{RQ}}^r = \sum_{l=1}^{L} ||\text{sg}[\mathbf{r}_{l-1}^r] - \mathbf{e}_{r_l}||_2^2 + ||\mathbf{r}_{l-1}^r - \text{sg}[\mathbf{e}_{r_l}]||_2^2, \\ \mathcal{L}_{\text{RQ}} = \mathcal{L}_{\text{RQ}}^s + \mathcal{L}_{\text{RQ}}^r, \end{cases} \tag{5}$$

where $\text{sg}[\cdot]$ indicates the stop-gradient operation. The loss $\mathcal{L}_{\text{RQ}}$ is employed to optimize the code embeddings across all codebooks. Finally, the total objective for user preference quantization combines the reconstruction loss, quantization loss, and contrastive loss as:

$$\mathcal{L}_{\text{RQ-VAE}} = \mathcal{L}_{\text{Recon}} + \lambda_{\text{RQ}}\mathcal{L}_{\text{RQ}} + \lambda_{\text{RQ-CL}}\mathcal{L}_{\text{RQ-CL}}, \tag{6}$$

where λ_{RQ} and $\lambda_{\text{RQ-CL}}$ are hyper-parameters controlling the contributions of the respective loss components.

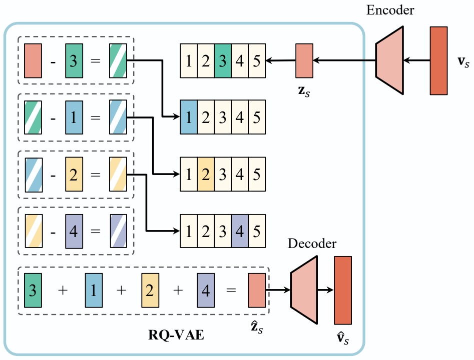


Figure 4: The Residual Quantized Variational Autoencoder (RQ-VAE) process. We illustrate the procedure using the quantization of the search preference embedding $\mathbf{v}_s$ as an example.

4.1.3 Graph Construction. After quantizing user S&R preferences into discrete codes, we construct two bipartite graphs to model the relationships between users and their corresponding S&R code representations. Specifically, let $\widetilde{\mathcal{S}}$ and $\widetilde{\mathcal{R}}$ denote the sets of search codes and recommendation codes, respectively. The affiliation matrices between users and these codes are defined as $\mathbf{AS} \in \{0, 1\}^{|\mathcal{U}| \times |\widetilde{\mathcal{S}}|}$ and $\mathbf{AR} \in \{0, 1\}^{|\mathcal{U}| \times |\widetilde{\mathcal{R}}|}$, where $\mathbf{AS}_{u,s} = 1$ indicates that user u is associated with the search code s, and similarly for $\mathbf{AR}$ with recommendation codes.

Based on the affiliation matrices, we construct two bipartite graphs: $\mathcal{G}_s = \{\mathcal{V}_s, \mathcal{E}_s\}$ for search preferences and $\mathcal{G}_r = \{\mathcal{V}_r, \mathcal{E}_r\}$ for recommendation preferences. The node sets are defined as $\mathcal{V}_s = \mathcal{U} \cup \widetilde{\mathcal{S}}$ and $\mathcal{V}_r = \mathcal{U} \cup \widetilde{\mathcal{R}}$. The edge sets are given by $\mathcal{E}_s = \{(u, s) | u \in \mathcal{U}, s \in \widetilde{\mathcal{S}}, \mathbf{AS}_{u,s} = 1\}$ and $\mathcal{E}_r = \{(u, r) | u \in \mathcal{U}, r \in \widetilde{\mathcal{R}}, \mathbf{AR}_{u,r} = 1\}$, where an edge indicates an affiliation between a user and a corresponding preference code.

The constructed bipartite graphs are utilized to enhance user embeddings for subsequent search enhanced recommendation modeling, which will be detailed in the next section.

4.2 Search Enhanced Recommendation Modeling

This section introduces the Search Enhanced Recommendation Modeling module. We first apply message passing over the user-code graph to enhance the embeddings of users with sparse search interactions by leveraging information from users with rich search interactions. Then, the enhanced user and code embeddings are integrated with the S&R histories in the downstream search-enhanced recommendation model for final prediction.

4.2.1 Embedding Layer. We maintain three embedding tables to represent users, items, and query words: $\mathbf{E}_{\mathcal{U}} \in \mathbb{R}^{|\mathcal{U}| \times d}$, $\mathbf{E}_{\mathcal{I}} \in \mathbb{R}^{|\mathcal{I}| \times d}$, and $\mathbf{E}_{\mathcal{W}} \in \mathbb{R}^{|\mathcal{W}| \times d}$, respectively. Here, $\mathcal{W}$ denotes the vocabulary comprising all words appearing in user queries, and d is the embedding dimension. Given a specific user u and item i, their corresponding embeddings $\mathbf{e}_u \in \mathbb{R}^d$ and $\mathbf{e}_i \in \mathbb{R}^d$ are retrieved via standard lookup operations. For a query q composed of a sequence of words $\{w_1, w_2, \dots, w_{|q|}\} \subseteq \mathcal{W}$, we follow prior work [34, 38] and represent the query by averaging its constituent word embeddings: $\mathbf{e}_q = \text{Mean}(\mathbf{e}_{w_1}, \mathbf{e}_{w_2}, \dots, \mathbf{e}_{w_{|q|}}) \in \mathbb{R}^d$, where $\mathbf{e}_{w_i} \in \mathbb{R}^d$ denotes the embedding of the i-th word in the query.

In addition, we introduce two embedding tables, $\mathbf{E}^{(0)}_{\widetilde{\mathcal{S}}} \in \mathbb{R}^{|\widetilde{\mathcal{S}}|\times d}$ and $\mathbf{E}^{(0)}_{\widetilde{\mathcal{R}}} \in \mathbb{R}^{|\widetilde{\mathcal{R}}|\times d}$, to represent the discrete codes corresponding to S&R preferences, respectively. The initial embedding of a search code $s \in \widetilde{\mathcal{S}}$ or a recommendation code $r \in \widetilde{\mathcal{R}}$, denoted as $\mathbf{e}^{(0)}_s$ and $\mathbf{e}^{(0)}_r$, is obtained via standard embedding lookup from the respective tables.

4.2.2 Message Passing over User-Code Graphs. To leverage the representations of users with rich search interactions to enhance those of users with sparse interactions, we perform message passing on the user-code graph defined in § 4.1.3. In these graphs, users are connected via shared preference codes, allowing semantically similar users to exchange information and mutually enhance their representations. Specifically, we adopt LightGCN [13] as the propagation framework, leveraging its simplified yet effective design to iteratively refine user and code embeddings through neighborhood aggregation.

Taking the propagation over the graph $\mathcal{G}_s$ as an example, the embeddings are iteratively updated through K layers of message passing. Let $\mathbf{e}^{S(k)}_u$ and $\mathbf{e}^{S(k)}_s$ denote the embeddings of user u and search code $s \in \widetilde{\mathcal{S}}$ at the k-th layer, respectively, where the initial embeddings are given by $\mathbf{e}^{S(0)}_u = \mathbf{e}_u$ and $\mathbf{e}^{S(0)}_s = \mathbf{e}^{(0)}_s$. The update rule at the k-th propagation layer is defined as:

$$\begin{aligned} \mathbf{e}^{S(k)}_u &= \sum_{s \in \mathcal{N}_u} \frac{1}{\sqrt{|\mathcal{N}_u||\mathcal{N}_s|}} \cdot \mathbf{e}^{S(k-1)}_s, \\ \mathbf{e}^{S(k)}_s &= \sum_{u \in \mathcal{N}_s} \frac{1}{\sqrt{|\mathcal{N}_u||\mathcal{N}_s|}} \cdot \mathbf{e}^{S(k-1)}_u, \end{aligned} \tag{7}$$

where $\mathcal{N}_u$ and $\mathcal{N}_s$ denote the neighboring codes of user u and the neighboring users of code s, respectively. The final embeddings are obtained by aggregating embeddings from all layers:

$$\mathbf{e}^s_u = \frac{1}{K+1} \sum_{k=0}^{K} \mathbf{e}^{S(k)}_u, \quad \mathbf{e}_s = \frac{1}{K+1} \sum_{k=0}^{K} \mathbf{e}^{S(k)}_s, \tag{8}$$

where $\mathbf{e}^s_u, \mathbf{e}_s \in \mathbb{R}^d$ are the final representations for user u and code s, respectively. In a similar manner, message passing is performed over the graph $\mathcal{G}_r$ to obtain the final representations $\mathbf{e}^r_u$ and $\mathbf{e}_r$ for user u and recommendation code $r \in \widetilde{\mathcal{R}}$. Then, we obtain the enhanced S&R embeddings for all users:

$$\mathbf{E}^s_{\mathcal{U}} = [\mathbf{e}^s_{u_1}, \mathbf{e}^s_{u_2}, \dots, \mathbf{e}^s_{u_{|\mathcal{U}|}}]^\top, \quad \mathbf{E}^r_{\mathcal{U}} = [\mathbf{e}^r_{u_1}, \mathbf{e}^r_{u_2}, \dots, \mathbf{e}^r_{u_{|\mathcal{U}|}}]^\top,$$

where $\mathbf{E}^s_{\mathcal{U}}, \mathbf{E}^r_{\mathcal{U}} \in \mathbb{R}^{|\mathcal{U}|\times d}$ denote the user's S&R embeddings, respectively. Similarly, we obtain the code embeddings for S&R:

$$\mathbf{E}_{\widetilde{\mathcal{S}}} = [\mathbf{e}_{s_1}, \mathbf{e}_{s_2}, \dots, \mathbf{e}_{s_{|\widetilde{\mathcal{S}}|}}]^\top, \quad \mathbf{E}_{\widetilde{\mathcal{R}}} = [\mathbf{e}_{r_1}, \mathbf{e}_{r_2}, \dots, \mathbf{e}_{r_{|\widetilde{\mathcal{R}}|}}]^\top,$$

where $\mathbf{E}_{\widetilde{\mathcal{S}}} \in \mathbb{R}^{|\widetilde{\mathcal{S}}|\times d}$ and $\mathbf{E}_{\widetilde{\mathcal{R}}} \in \mathbb{R}^{|\widetilde{\mathcal{R}}|\times d}$ represent the embeddings of the S&R codes, respectively.

After obtaining the enhanced user S&R preference embeddings, $\mathbf{e}^s_u$ and $\mathbf{e}^r_u$, through propagation over the graphs $\mathcal{G}_s$ and $\mathcal{G}_r$, respectively, we align the two embeddings to better capture user-level similarity, which facilitates more effective information transfer during message passing. Moreover, the aligned embeddings are also utilized in subsequent downstream tasks. Specifically, we adopt contrastive learning and compute the following InfoNCE loss:

$$\begin{aligned} \mathcal{L}_{\text{U-CL}} = -\Bigg[&\log \frac{\exp(\text{sim}(\mathbf{e}^s_u, \mathbf{e}^r_u)/\tau_2)}{\sum_{u^- \in \mathcal{U}_{\text{neg}}} \exp(\text{sim}(\mathbf{e}^s_u, \mathbf{e}^r_{u^-})/\tau_2)} \\ &+ \log \frac{\exp(\text{sim}(\mathbf{e}^s_u, \mathbf{e}^r_u)/\tau_2)}{\sum_{u^- \in \mathcal{U}_{\text{neg}}} \exp(\text{sim}(\mathbf{e}^s_{u^-}, \mathbf{e}^r_u)/\tau_2)} \Bigg], \end{aligned} \tag{9}$$

where τ_2 is a learnable temperature coefficient and $\mathcal{U}_{\text{neg}}$ is the set of in-batch negative users. After message passing, we retrieve the embeddings of the user's S&R code sequences, $\tilde{s}$ and $\tilde{r}$, by performing a lookup on the embedding tables $\mathbf{E}_{\widetilde{\mathcal{S}}}$ and $\mathbf{E}_{\widetilde{\mathcal{R}}}$, as learned from Eq. (3) in § 4.1.2, as follows:

$$\mathbf{E}_{\tilde{s}} = [\mathbf{e}_{s_1}, \mathbf{e}_{s_2}, \dots, \mathbf{e}_{s_L}]^\top \in \mathbb{R}^{L\times d}, \; \mathbf{E}_{\tilde{r}} = [\mathbf{e}_{r_1}, \mathbf{e}_{r_2}, \dots, \mathbf{e}_{r_L}]^\top \in \mathbb{R}^{L\times d}. \tag{10}$$

The aligned user embeddings $\mathbf{e}^s_u$ and $\mathbf{e}^r_u$, as well as the S&R code sequence embeddings $\mathbf{E}_{\tilde{s}}$ and $\mathbf{E}_{\tilde{r}}$, are subsequently utilized in downstream modeling.

4.2.3 History Modeling. We first obtain the embeddings of the user's S&R histories via the lookup operation. Specifically, the embedding of the recommendation history is obtained by concatenating the embeddings of the constituent items:

$$\mathbf{E}_r = [\mathbf{e}_{i_1}, \mathbf{e}_{i_2}, \dots, \mathbf{e}_{i_{N_r}}]^\top \in \mathbb{R}^{N_r \times d}.$$

For the search history, the embedding of each record is computed by summing the embedding of the query and the mean-pooled embedding of its associated clicked items. The overall embedding of the search history is formulated as:

$$\mathbf{E}_s = [\mathbf{e}_{q_1} + \text{M}(\mathcal{I}_{q_1}), \mathbf{e}_{q_2} + \text{M}(\mathcal{I}_{q_2}), \dots, \mathbf{e}_{q_{N_s}} + \text{M}(\mathcal{I}_{q_{N_s}})]^\top \in \mathbb{R}^{N_s \times d},$$

where $\text{M}(\mathcal{I}_{q_k}) = \text{MEAN}(\mathbf{e}_{i_1}, \mathbf{e}_{i_2}, \dots, \mathbf{e}_{i_{N_{q_k}}})$ denotes the mean of the embeddings of items clicked in response to query q_k.

To capture the sequential dependencies within user behavior sequences, we introduce position embeddings $\mathbf{P}_s \in \mathbb{R}^{N_s \times d}$ and $\mathbf{P}_r \in \mathbb{R}^{N_r \times d}$ for the S&R histories, respectively. The final representations of the S&R histories are computed as follows:

$$\widehat{\mathbf{E}}_s = \mathbf{E}_s + \mathbf{P}_s, \quad \widehat{\mathbf{E}}_r = \mathbf{E}_r + \mathbf{P}_r.$$

To further model the contextual representations of user S&R histories, we encode them separately using two Transformer [44] encoders, each consisting of a Multi-Head Self-Attention (MSA) layer followed by a Feed-Forward Network (FFN). The historical embeddings serve as the query, key, and value in the MSA mechanism. The encoding process is formulated as:

$$\mathbf{H}_s = \text{FFN}_s(\text{MSA}_s(\widehat{\mathbf{E}}_s, \widehat{\mathbf{E}}_s, \widehat{\mathbf{E}}_s)), \; \mathbf{H}_r = \text{FFN}_r(\text{MSA}_r(\widehat{\mathbf{E}}_r, \widehat{\mathbf{E}}_r, \widehat{\mathbf{E}}_r)),$$

where $\mathbf{H}_s \in \mathbb{R}^{N_s \times d}$ and $\mathbf{H}_r \in \mathbb{R}^{N_r \times d}$ denote the contextualized embeddings of the S&R histories, respectively.

$\mathbf{H}_s$ and $\mathbf{H}_r$ encode the user's interests reflected in their S&R histories, respectively. In contrast, the code embeddings $\mathbf{E}_{\tilde{s}}$ and $\mathbf{E}_{\tilde{r}}$ derived in Eq. (10) capture user preferences enhanced by collaborative relationships among users. We fuse these two types of representations to obtain enriched representations of the user's

S&R histories. Specifically, we employ Multi-Head Cross-Attention (MCA), where the history representations serve as queries, and the corresponding code embeddings act as keys and values. The fusion process is computed as follows:

$$\begin{aligned} \mathbf{F}_s &= \mathrm{MSA}_s(\mathbf{H}_s, \mathbf{H}_s, \mathbf{H}_s),\ \mathbf{W}_s = \mathrm{FFN}_s(\mathrm{MCA}_s(\mathbf{F}_s, \mathbf{E}_{\tilde{s}}, \mathbf{E}_{\tilde{s}})), \\ \mathbf{F}_r &= \mathrm{MSA}_r(\mathbf{H}_r, \mathbf{H}_r, \mathbf{H}_r),\ \mathbf{W}_r = \mathrm{FFN}_r(\mathrm{MCA}_r(\mathbf{F}_r, \mathbf{E}_{\tilde{r}}, \mathbf{E}_{\tilde{r}})), \end{aligned} \tag{11}$$

where $\mathbf{W}_s \in \mathbb{R}^{N_s \times d}$ and $\mathbf{W}_r \in \mathbb{R}^{N_r \times d}$ denote the final contextually enriched representations for the S&R histories, respectively.

To enable more effective fusion, we first align the history embeddings with the code embeddings. Taking the search history as an example, we compute the mean of the search history embeddings and the search code sequence embeddings to obtain $\mathbf{h}_s = \mathrm{MEAN}(\mathbf{H}_s) \in \mathbb{R}^d$ and $\mathbf{e}_{\tilde{s}} = \mathrm{MEAN}(\mathbf{E}_{\tilde{s}}) \in \mathbb{R}^d$, respectively. Then we employ contrastive learning and computer the following loss:

$$\begin{aligned} \mathcal{L}_{\text{S-CL}} = -\Bigg[&\log \frac{\exp(\mathrm{sim}(\mathbf{h}_s, \mathbf{e}_{\tilde{s}})/\tau_3)}{\sum_{\mathbf{e}_{\tilde{s}}^- \in \mathcal{E}_{\tilde{s}}^{\text{neg}}} \exp(\mathrm{sim}(\mathbf{h}_s, \mathbf{e}_{\tilde{s}}^-)/\tau_3)} \\ &+ \log \frac{\exp(\mathrm{sim}(\mathbf{h}_s, \mathbf{e}_{\tilde{s}})/\tau_3)}{\sum_{\mathbf{h}_s^- \in \mathcal{H}_s^{\text{neg}}} \exp(\mathrm{sim}(\mathbf{h}_s^-, \mathbf{e}_{\tilde{s}})/\tau_3)} \Bigg], \end{aligned} \tag{12}$$

where τ_3 is a learnable temperature coefficient, $\mathcal{E}_{\tilde{s}}^{\text{neg}}$ and $\mathcal{H}_s^{\text{neg}}$ are in-batch negative samples. Similarly, we can get the contrastive loss $\mathcal{L}_{\text{R-CL}}$ for recommendation history and code sequence. The total contrastive loss is formulated as follows:

$$\mathcal{L}_{\text{His-CL}} = \mathcal{L}_{\text{S-CL}} + \mathcal{L}_{\text{R-CL}}. \tag{13}$$

After obtaining the representations of the user's S&R histories, $\mathbf{W}_s$ and $\mathbf{W}_r$, we perform history pooling based on the similarity between each historical behavior and the next candidate item. Specifically, we apply a Self-Attention (SA) mechanism as follows:

$$\mathbf{w}_s = \mathrm{SA}(\mathbf{e}_{i_{T+1}}, \mathbf{W}_s, \mathbf{W}_s), \quad \mathbf{w}_r = \mathrm{SA}(\mathbf{e}_{i_{T+1}}, \mathbf{W}_r, \mathbf{W}_r), \tag{14}$$

where $\mathbf{w}_s, \mathbf{w}_r \in \mathbb{R}^d$ are the aggregated representations of the S&R histories, respectively. Here, the embedding of the next candidate item $\mathbf{e}_{i_{T+1}}$ serves as the query, while the S&R history representations act as the key and value for the attention computation.

4.3 Model Prediction and Training

4.3.1 Prediction. Finally, we concatenate the user's S&R representations, $\mathbf{e}_u^s$ and $\mathbf{e}_u^r$, obtained from Eq. (8), along with the historical representations $\mathbf{w}_s$ and $\mathbf{w}_r$ derived from Eq. (14), and the embedding of the next candidate item $\mathbf{e}_{i_{T+1}}$. The concatenated vector is then fed into a multi-layer perceptron (MLP) to predict the user's preference for the next item:

$$\hat{y}_{u,i_{T+1}} = \mathrm{MLP}(\mathrm{CONCAT}(\mathbf{e}_u^s, \mathbf{e}_u^r, \mathbf{w}_s, \mathbf{w}_r, \mathbf{e}_{i_{T+1}})), \tag{15}$$

where $\hat{y}_{u,i_{T+1}}$ is the predicted preference score. $\mathrm{CONCAT}(\cdot)$ denotes the concatenation operation.

4.3.2 Training. Following previous works [34, 38, 73], we adopt the binary cross-entropy loss to optimize our recommendation model:

$$\mathcal{L}_{\text{rec}} = -\frac{1}{|\mathcal{D}|} \sum_{(u,i_{T+1}) \in \mathcal{D}} y_{u,i_{T+1}} \log(\hat{y}_{u,i_{T+1}}) + (1 - y_{u,i_{T+1}}) \log(1 - \hat{y}_{u,i_{T+1}}), \tag{16}$$

Table 1: Dataset statistics used in this paper. "S" and "R" represent search and recommendation, respectively.

Dataset	#Users	#Items	#Queries	#Interaction-S	#Interaction-R
CDs	75,258	64,443	671	852,889	1,097,592
Electronics	192,403	63,001	982	1,280,465	1,689,188
Qilin	15,482	1,983,938	44,820	969,866	1,438,435

where $\mathcal{D}$ denotes the set of user-item interaction pairs used for training. Here, $y_{u,i_{T+1}} \in \{0, 1\}$ is the ground-truth label indicating whether user u interacts with item i_{T+1}.

Finally, the overall training loss of our model combines the recommendation loss defined in Eq. (16) with two auxiliary contrastive losses introduced in Eq. (9) and Eq. (13). Additionally, we incorporate an L_2 regularization term to prevent overfitting. The total loss is formulated as:

$$\mathcal{L}_{\text{Total}} = \mathcal{L}_{\text{rec}} + \lambda_{\text{U-CL}} \mathcal{L}_{\text{U-CL}} + \lambda_{\text{His-CL}} \mathcal{L}_{\text{His-CL}} + \lambda_{\text{Reg}} ||\Theta||^2, \tag{17}$$

where $\lambda_{\text{U-CL}}$, $\lambda_{\text{His-CL}}$, and λ_{Reg} are hyper-parameters that control the contributions of the user-level alignment loss, history-level alignment loss, and regularization term, respectively. Here, $||\Theta||^2$ represents the L_2 norm of the model parameters Θ, which helps to regularize the model and improve its generalization ability.

4.4 Discussion

Computational Efficiency and Complexity. Our method leverages LLM inference solely for summarizing user preferences, which can be performed offline. As a result, the online serving phase only requires the recommendation model, ensuring high efficiency. Regarding the user-code graph, take $\mathcal{G}_s$ (described in § 4.1.3) as an example. It consists of $|\mathcal{U}| + |\widetilde{\mathcal{S}}|$ nodes, where $|\mathcal{U}|$ denotes the number of users and $|\widetilde{\mathcal{S}}|$ the number of search codes. The number of codes $|\widetilde{\mathcal{S}}|$ is at most $L \times N_c$, where L is the number of codebooks used in RQ-VAE (§ 4.1.2) and N_c is the size of each codebook. In practice, both $L \ll |\mathcal{U}|$ and $N_c \ll |\mathcal{U}|$, so the resulting graph remains lightweight. This structure is significantly more efficient than prior graph-based recommendation models [13, 47, 53], where the node count is $|\mathcal{U}| + |\mathcal{I}|$, with $|\mathcal{I}|$ (the number of items) typically much larger than both L and N_c ($|\mathcal{I}| \gg L$ and $|\mathcal{I}| \gg N_c$).

Comparison with Existing Methods. In contrast to existing search-enhanced recommendation models, our method explicitly targets the challenge of sparse search interactions by constructing the user-code graphs. Through message passing on the graphs, user embeddings with rich search interactions are leveraged to enhance those of users with sparse behaviors. This design leads to more substantial performance gains, especially for users with limited search histories.

Compared to GNN-based graph recommendation models [13, 47, 53], which commonly construct user-item graphs to capture collaborative filtering signals, our approach instead builds the user-code graphs aimed at enhancing user representations. This graph structure facilitates more effective user-user information sharing via shared discrete codes, offering a novel perspective on graph-based representation learning.

Table 2: Overall recommendation performance comparison of different methods on all datasets. H@k and N@k denote HR@k and NDCG@k, respectively. The best and second-best results are highlighted in bold and underlined fonts, respectively. "*" denotes that the improvement over the second-best method is statistically significant (t-test, p-value < 0.05).

Category	Methods	CDs					Electronics					Qilin				
		H@1	H@5	H@10	N@5	N@10	H@1	H@5	H@10	N@5	N@10	H@1	H@5	H@10	N@5	N@10
Recommendation	LightGCN	0.0911	0.3285	0.4963	0.2103	0.2645	0.0277	0.1025	0.1707	0.0648	0.0867	0.0624	0.2436	0.3820	0.1530	0.1976
	SGL	0.1431	0.3660	0.4929	0.2579	0.2988	0.0304	0.1081	0.1816	0.0690	0.0926	0.0780	0.2566	0.3854	0.1684	0.2099
	SimGCL	0.1651	0.3874	0.5141	0.2799	0.3207	0.0395	0.1261	0.2061	0.0827	0.1084	0.0751	0.2607	0.3889	0.1684	0.2097
	GRU4Rec	0.1360	0.4191	0.5854	0.2806	0.3344	0.0579	0.2005	0.3144	0.1295	0.1661	0.1295	0.3443	0.4855	0.2395	0.2850
	SASRec	0.1621	0.4134	0.5638	0.2907	0.3393	0.1019	0.2188	0.3121	0.1608	0.1907	0.1334	0.3449	0.4768	0.2411	0.2836
	BERT4Rec	0.1692	0.4226	0.5750	0.2993	0.3485	0.1031	0.2242	0.3224	0.1642	0.1957	0.1319	0.3472	0.4832	0.2426	0.2864
	CL4SRec	0.1834	0.4570	0.6084	0.3240	0.3730	0.1052	0.2308	0.3299	0.1684	0.2003	0.1363	0.3489	0.4848	0.2455	0.2894
	KAR	0.1922	0.4937	0.6394	0.3483	0.3955	0.0958	0.2513	0.3629	0.1750	0.2108	0.1140	0.3200	0.4551	0.2188	0.2625
	LLM-ESR	0.2079	0.5104	0.6610	0.3648	0.4136	0.1055	0.2560	0.3672	0.1817	0.2175	0.1422	0.3599	0.4932	0.2532	0.2963
Search Enhanced Recommendation	NRHUB	0.1454	0.4243	0.5825	0.2885	0.3397	0.0533	0.1820	0.2889	0.1179	0.1522	0.1389	0.3543	0.4829	0.2499	0.2913
	Query-SeqRec	0.1832	0.4537	0.6066	0.3219	0.3713	0.1009	0.2219	0.3205	0.1619	0.1935	0.1299	0.3473	0.4824	0.2412	0.2847
	JSR	0.1808	0.4346	0.5807	0.3113	0.3586	0.1090	0.2289	0.3246	0.1694	0.2001	0.1445	0.3711	0.5077	0.2608	0.3048
	USER	0.1904	0.4929	0.6465	0.3465	0.3963	0.0672	0.2146	0.3270	0.1415	0.1776	0.1549	0.3820	0.5199	0.2715	0.3161
	SESRec	0.2019	0.5059	0.6494	0.3595	0.4060	0.0790	0.2403	0.3572	0.1606	0.1982	0.1535	0.3694	0.4904	0.2647	0.3037
	UnifiedSSR	0.2079	0.4928	0.6359	0.3549	0.4012	0.1066	0.2343	0.3320	0.1711	0.2025	0.1412	0.3595	0.4957	0.2532	0.2972
	UniSAR	0.2219	0.5249	0.6712	0.3797	0.4271	0.0996	0.2633	0.3757	0.1829	0.2191	0.1616	0.3835	0.5099	0.2767	0.3174
	GSERec	**0.2505***	**0.5459***	**0.6825***	**0.4045***	**0.4487***	**0.1205***	**0.2739***	**0.3788***	**0.1989***	**0.2327***	**0.1812***	**0.4062***	**0.5335***	**0.2988***	**0.3400***

5 Experiments

We conducted extensive experiments to evaluate the performance of GSERec. The code is available[1].

5.1 Experimental Setup

5.1.1 Dataset. Since GSERec relies on both users' S&R interaction data, as well as the textual information of items, we conduct experiments on the following publicly available datasets. The statistics of these datasets are summarized in Table 1.

Amazon[2] [12, 21]: We adopted a widely used semi-synthetic dataset. Following previous studies [1, 2, 34, 38], we generated synthetic search behaviors based on an existing recommendation dataset. We used the "CDs and Vinyl" and "Electronics" subsets and selected their five-core versions, ensuring that each user and item has at least five interactions. [3]

Qilin [4]: The dataset is collected from Xiaohongshu[4], a well-known lifestyle search engine in China with over 300 million monthly active users. It contains user behavior data from both S&R scenarios, as well as multimodal information for all items. In this work, we utilize only the textual information.

Following previous works [34, 38, 39], we applied the leave-one-out strategy to split all the dataset into training, validation, and test sets.

5.1.2 Baselines. We compare GSERec with two categories of baselines to comprehensively evaluate its effectiveness: (1) *Recommendation*: **LightGCN** [13]; **SGL** [47]; **SimGCL** [53]; **GRU4Rec** [15]; **SASRec** [16]; **BERT4Rec** [39]; **CL4SRec** [51]; **KAR** [48]; **LLM-ESR** [20]. (2) *Search enhanced recommendation*: **NRHUB** [46]; **Query-SeqRec** [14]; **JSR** [55]; **USER** [52]; **SESRec** [38]; **UnifiedSSR** [50]; **UniSAR** [34].

5.1.3 Evaluation. Following previous studies [34, 38, 39], we adopt *Hit Ratio* (HR) and *Normalized Discounted Cumulative Gain* (NDCG) as our evaluation metrics. We report HR at top $\{1, 5, 10\}$ ranks and NDCG at top $\{5, 10\}$ ranks. Following the standard evaluation protocol [16, 34, 38], each ground-truth item is paired with 99 randomly sampled negative items with which the user has no prior interactions to form the candidate list.

5.1.4 Implementation Details. User-Code Graph Construction (§ 4.1): We use the LLM DeepSeek-R1-Distill-Qwen-7B[5] [9] to summarize user S&R preferences, which are then embedded via BGE-M3[6] [5]. The RQ-VAE (§ 4.1.2) uses $L = 4$ codebooks with $N_c = 256$ codes each and code dimension $d_l = 32$. We set $\lambda_{\text{RQ}} = 1.0$ (Eq, (6)) and train the quantization model for 500 epochs using Adam [17] with a batch size of 1024 and learning rate 1e-3. The temperature τ_1 (Eq. (1)) is 0.1, and the contrastive loss weight $\lambda_{\text{RQ-CL}}$ (Eq. (6)) is tuned from {1e-4, 1e-3, 1e-2, 1e-1, 1}.

Search Enhanced Recommendation Modeling (§ 4.2): Each baseline was tuned per dataset based on the original paper settings. For our model, embedding dimension d is 64 for CDs/Electronics and 32 for Qilin. The max history length is 20 for CDs/Qilin and 10 for Electronics. We use 2 LightGCN layers ($K = 2$). Temperatures τ_2 (Eq. (9)) and τ_3 (Eq. (12)) are set to 0.1. Loss weights $\lambda_{\text{U-CL}}$ and $\lambda_{\text{His-CL}}$ (Eq. (17)) are tuned over {1e-4, 1e-3, 1e-2, 1e-1, 1}. All models are trained for up to 100 epochs with Adam [17], using a batch size of 1024 and early stopping. The learning rate is searched from {1e-3, 1e-4, 1e-5}, and λ_{Reg} (Eq. (17)) is tuned over {1e-5, 1e-6, 1e-7}.

5.2 Overall Performance

Table 2 presents the recommendation results on three datasets. From the results, we can observe the following:

• Firstly, it can be observed that compared to existing recommendation or search-enhanced recommendation models, GSERec achieves state-of-the-art results. This validates the effectiveness of GSERec in alleviating data sparsity by constructing the user-code graphs

[1] https://github.com/TengShi-RUC/GSERec

[2] https://cseweb.ucsd.edu/~jmcauley/datasets/amazon/links.html, https://github.com/QingyaoAi/Amazon-Product-Search-Datasets.

[3] Due to the lack of textual information for approximately 70% of items in the "Kindle Store" subset used in previous works [34, 38], we instead used the "CDs and Vinyl" and "Electronics" subsets, where fewer than 1% of items lack text.

[4] https://www.xiaohongshu.com.

[5] https://huggingface.co/deepseek-ai/DeepSeek-R1-Distill-Qwen-7B.

[6] https://huggingface.co/BAAI/bge-m3.

Table 3: Ablation study conducted on the Qilin dataset, where "w/o" indicates that the corresponding module has been removed from GSERec. "U-C Graph" denotes the user-code graph.

Model	H@1	H@5	H@10	N@5	N@10
GSERec	**0.1812**	**0.4062**	**0.5335**	**0.2988**	**0.3400**
w/o $\mathcal{L}_{\text{RQ-CL}}$ (Eq. (1))	0.1665	0.3975	0.5272	0.2862	0.3281
w/o U-C Graph	0.1428	0.3647	0.4858	0.2576	0.2967
w/o $\mathcal{L}_{\text{U-CL}}$ (Eq. (9))	0.1632	0.3922	0.5230	0.2819	0.3242
w/o $\mathcal{L}_{\text{His-CL}}$ (Eq. (13))	0.1654	0.3935	0.5215	0.2835	0.3248
w/o MCA	0.1763	0.4032	0.5295	0.2933	0.3358

and performing message passing, thereby enhancing the representations of users with sparse search interactions using information from users with richer search behaviors.

- Secondly, we observe that search-enhanced recommendation models, such as GSERec and UniSAR, generally outperform traditional recommendation methods. However, models like NRHUB perform worse than traditional baselines in some cases, indicating that simply incorporating search features does not necessarily lead to improved performance. This highlights the need for dedicated designs to effectively learn representations of search features.
- Thirdly, we also observe that graph-based models such as LightGCN underperform compared to sequential recommendation models like SASRec, highlighting the importance of leveraging users' historical behaviors. Meanwhile, models leveraging LLMs, including GSERec, KAR, and LLM-ESR, achieve significant improvements over traditional recommendation methods, highlighting the effectiveness of incorporating LLMs into recommendation tasks.
- Finally, we compare the performance of the baselines and our model across user groups with varying numbers of search interactions, as shown in Figure 2. Due to space limitations, we report results on the Qilin dataset, comparing the traditional recommendation model SASRec, the search-enhanced model UniSAR, and our proposed model GSERec. As observed, UniSAR achieves larger improvements for users with rich search interactions, while GSERec further outperforms it for users with sparse search interactions. This further confirms the effectiveness of GSERec in alleviating the sparsity of search interactions.

5.3 Ablation Study

Due to space limitations, we conduct ablation studies on Qilin, the dataset containing real user S&R interactions, to evaluate the effectiveness of each module in GSERec. The results are shown in Table 3.

5.3.1 Effectiveness of User Alignment in Preference Quantization. In § 4.1.2, we leverage contrastive learning to align the latent embeddings of users' S&R preferences, promoting the capture of user-level similarity. This alignment is enforced via the loss term $\mathcal{L}_{\text{RQ-CL}}$ defined in Eq. (1). As demonstrated by the "w/o $\mathcal{L}_{\text{RQ-CL}}$" setting in Table 3, removing this loss leads to a significant performance drop, underscoring its critical role. By ensuring better alignment before quantization, the generated codes more effectively reflect inter-user similarities, thereby improving downstream tasks.

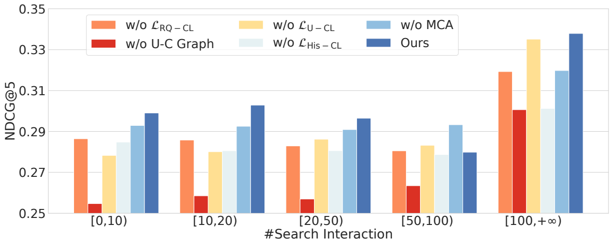


Figure 5: Ablation study across user groups with varying numbers of search interactions

5.3.2 Effectiveness of User-Code Graphs. To enrich the representations of users with sparse search interactions, we construct user-code graphs using discrete codes (§ 4.1.3) and apply message passing (§ 4.2.2) to propagate information from users with richer behaviors. To assess its effectiveness, we ablate the user-code graph, allowing the model to rely solely on users' S&R histories for prediction. As shown by the "w/o U-C Graph" setting in Table 3, performance drops significantly, confirming the utility of leveraging rich user interactions to enhance sparse-user representations.

Furthermore, to facilitate more effective message passing, we introduce a user alignment loss $\mathcal{L}_{\text{U-CL}}$ in Eq. (9) to align user embeddings and better capture cross-user similarity. Removing this loss ("w/o $\mathcal{L}_{\text{U-CL}}$") leads to notable degradation in performance, demonstrating the importance of embedding alignment in improving information propagation and similarity modeling.

5.3.3 Effectiveness of Code and History Fusion. In § 4.2.3, we align and fuse the enhanced user S&R code embeddings ($\mathbf{E}_{\tilde{s}}$ and $\mathbf{E}_{\tilde{r}}$) with the corresponding user S&R histories. We separately evaluate the contributions of the alignment and fusion components.

For alignment, we introduce the contrastive loss $\mathcal{L}_{\text{His-CL}}$ in Eq. (13) to align the code sequences with users' historical behaviors. As shown by the "w/o $\mathcal{L}_{\text{His-CL}}$" setting in Table 3, removing this loss leads to noticeable performance degradation, underscoring the importance of aligning the two types of embeddings into a shared semantic space prior to fusion.

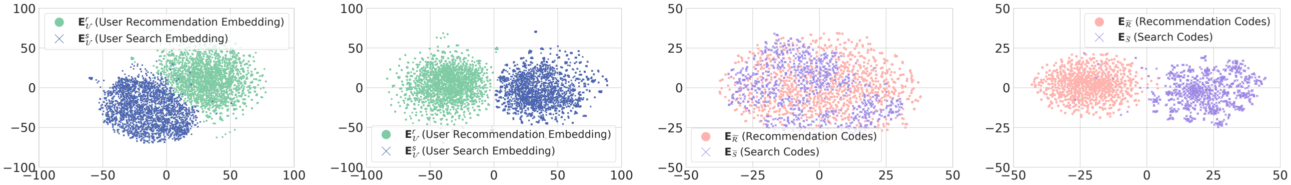
For the fusion step, as defined in Eq. (11), we employ Multi-head Cross Attention (MCA) to integrate the code sequences with the historical behavior representations. As indicated by the "w/o MCA" setting in Table3, the absence of MCA results in reduced performance, validating the effectiveness of incorporating enhanced code embeddings to enrich the modeling of S&R histories.

5.4 Experimental Analysis

We further conduct experimental analysis on the Qilin dataset to investigate the contributions of different modules.

5.4.1 Ablation study across user groups with varying numbers of search interactions. To further investigate the effectiveness of each module in addressing the search sparsity issue, we conduct ablation studies across user groups with different levels of search interaction sparsity, as shown in Figure 5.

We observe that removing the user-code graph module ("w/o U-C graph" in Figure 5) causes a more substantial performance degradation for users with sparse search interactions. This indicates that message passing on the user-code graph can transfer useful



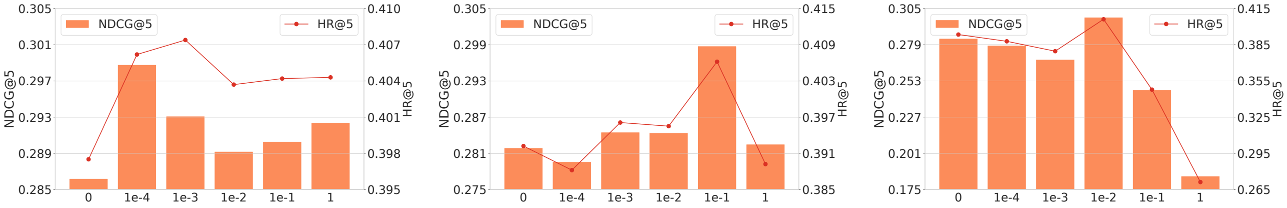
(a) $\mathbf{E}^s_{\mathcal{U}}$ and $\mathbf{E}^r_{\mathcal{U}}$ w/o $\mathcal{L}_{\text{U-CL}}$.

(b) $\mathbf{E}^s_{\mathcal{U}}$ and $\mathbf{E}^r_{\mathcal{U}}$ w/ $\mathcal{L}_{\text{U-CL}}$.

(c) $\mathbf{E}_{\widetilde{\mathcal{S}}}$ and $\mathbf{E}_{\widetilde{\mathcal{R}}}$ w/o $\mathcal{L}_{\text{U-CL}}$.

(d) $\mathbf{E}_{\widetilde{\mathcal{S}}}$ and $\mathbf{E}_{\widetilde{\mathcal{R}}}$ w/ $\mathcal{L}_{\text{U-CL}}$.

Figure 6: The t-SNE visualization of user S&R embeddings $\mathbf{E}^s_{\mathcal{U}}$ and $\mathbf{E}^r_{\mathcal{U}}$, as well as code embeddings $\mathbf{E}_{\widetilde{\mathcal{S}}}$ and $\mathbf{E}_{\widetilde{\mathcal{R}}}$ (§ 4.2.2), with and without the user alignment loss $\mathcal{L}_{\text{U-CL}}$ (Eq. (9)). "w/o" and "w/" denote results without and with the alignment loss, respectively.



(a) Performance of different $\lambda_{\text{RQ-CL}}$ (Eq. (6))

(b) Performance of different $\lambda_{\text{U-CL}}$ (Eq. (17))

(c) Performance of different $\lambda_{\text{His-CL}}$ (Eq. (17))

Figure 7: Impact of hyper-parameters $\lambda_{\text{RQ-CL}}$, $\lambda_{\text{U-CL}}$, and $\lambda_{\text{His-CL}}$ on model performance, evaluated by NDCG@5 and HR@5.

information from users with rich search histories to those with sparse ones, thereby improving their representation quality.

Furthermore, removing the user alignment loss $\mathcal{L}_{\text{U-CL}}$ (Eq. (9)) also leads to a more pronounced performance drop for sparse-search users. This highlights the critical role of the alignment loss in capturing user similarity, which enhances the effectiveness of message passing on the user-code graph. Moreover, this loss contributes to the learning of more discriminative user embeddings.

For the other modules, we generally observe that removing any of them leads to performance degradation across most user groups. This further validates the effectiveness and necessity of each component in the overall model.

5.4.2 Embedding Visualization.

To gain deeper insights into the representations learned through message passing on the user-code graph (§ 4.2.2), we visualize the user and code embeddings for both S&R. Specifically, we analyze the user embeddings $\mathbf{E}^s_{\mathcal{U}}$ and $\mathbf{E}^r_{\mathcal{U}}$, along with the corresponding code embeddings $\mathbf{E}_{\widetilde{\mathcal{S}}}$ and $\mathbf{E}_{\widetilde{\mathcal{R}}}$. We employ t-SNE [43] to project the high-dimensional embeddings into a two-dimensional space, as shown in Figure 6.

To assess the effectiveness of the user alignment loss, we compare the embedding distributions with and without the user contrastive loss $\mathcal{L}_{\text{U-CL}}$ (Eq. (9)). Without this loss, the embeddings for S&R are highly entangled, which can lead to redundant information being propagated through the graphs $\mathcal{G}_s$ and $\mathcal{G}_r$. In contrast, when the alignment loss is applied, the embeddings become more clearly separated, allowing the two graphs to model distinct user behavior patterns. This separation contributes to more effective message passing and ultimately leads to improved recommendation performance.

5.4.3 Impact of Hyper-parameters.

We analyze the influence of the alignment loss weights $\lambda_{\text{RQ-CL}}$, $\lambda_{\text{U-CL}}$, and $\lambda_{\text{His-CL}}$—corresponding to $\mathcal{L}_{\text{RQ-CL}}$ (Eq. (6)), $\mathcal{L}_{\text{U-CL}}$ (Eq. (17)), and $\mathcal{L}_{\text{His-CL}}$ (Eq. (17))—on the final recommendation performance. Results are shown in Figure 7. During each analysis, the other two weights are fixed to their optimal values: $\lambda_{\text{RQ-CL}}$ = 1e-4, $\lambda_{\text{U-CL}}$ = 1e-1, and $\lambda_{\text{His-CL}}$ = 1e-2.

We find that a non-zero $\lambda_{\text{RQ-CL}}$ consistently improves performance, highlighting the benefit of aligning user S&R preference embeddings before quantization to better capture user similarity. For $\lambda_{\text{U-CL}}$, non-zero values occasionally degrade performance, suggesting that this loss requires careful tuning to effectively model user similarity in the user-code graph. For $\lambda_{\text{His-CL}}$, overly large values significantly harm performance, likely due to overemphasis on aligning code sequences with user histories at the expense of the main recommendation loss $\mathcal{L}_{\text{rec}}$ (Eq. (17)). Thus, this weight must be appropriately balanced to ensure optimal performance.

6 Conclusion

In this paper, we propose GSERec to address data sparsity in search-enhanced recommendation by leveraging users with rich search interactions to improve representations for users with sparse behaviors. We first use a LLM to summarize each user's S&R preferences, which are then encoded and discretized via vector quantization. Users are connected to their codes, forming the user-code graphs where shared codes link similar users. Message passing on this graph enables knowledge transfer from rich to sparse users. We further introduce contrastive losses to enhance user similarity modeling. The refined user and code embeddings are finally integrated with user histories for prediction. Experiments on three real-world datasets show that GSERec consistently outperforms baselines, especially for users with sparse search activity.

Acknowledgments

This work was funded by the National Natural Science Foundation of China (No. 62472426), Beijing Key Laboratory of Research on Large Models and Intelligent Governance, fund for building world-class universities (disciplines) of Renmin University of China.

GenAI Usage Disclosure

During the writing of this paper, we used generative AI tools (e.g., ChatGPT) solely for the purpose of improving language clarity and grammar. No parts of the manuscript were generated directly by AI; all content, including ideas, experimental designs, results, and discussions, were conceived and written by the authors. The use of AI was limited to minor linguistic refinement and did not contribute to the creation of scientific content or analysis.

References

[1] Qingyao Ai, Daniel N Hill, SVN Vishwanathan, and W Bruce Croft. 2019. A zero attention model for personalized product search. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. 379–388.

[2] Qingyao Ai, Yongfeng Zhang, Keping Bi, Xu Chen, and W Bruce Croft. 2017. Learning a hierarchical embedding model for personalized product search. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 645–654.

[3] Xuheng Cai, Chao Huang, Lianghao Xia, and Xubin Ren. 2023. LightGCL: Simple Yet Effective Graph Contrastive Learning for Recommendation. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*.

[4] Jia Chen, Qian Dong, Haitao Li, Xiaohui He, Yan Gao, Shaosheng Cao, Yi Wu, Ping Yang, Chen Xu, Yao Hu, et al. 2025. Qilin: A Multimodal Information Retrieval Dataset with APP-level User Sessions. *arXiv preprint arXiv:2503.00501* (2025).

[5] Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024. BGE M3-Embedding: Multi-Lingual, Multi-Functionality, Multi-Granularity Text Embeddings Through Self-Knowledge Distillation. arXiv:2402.03216 [cs.CL]

[6] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*. PMLR, 1597–1607.

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

[8] Sunhao Dai, Ninglu Shao, Jieming Zhu, Xiao Zhang, Zhenhua Dong, Jun Xu, Quanyu Dai, and Ji-Rong Wen. 2024. Modeling user attention in music recommendation. In *2024 IEEE 40th International Conference on Data Engineering (ICDE)*. IEEE, 761–774.

[9] DeepSeek-AI. 2025. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. arXiv:2501.12948 [cs.CL] https://arxiv.org/abs/2501.12948

[10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*.

[11] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 9729–9738.

[12] Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *proceedings of the 25th international conference on world wide web*. 507–517.

[13] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. 2020. Lightgcn: Simplifying and powering graph convolution network for recommendation. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*. 639–648.

[14] Zhankui He, Handong Zhao, Zhaowen Wang, Zhe Lin, Ajinkya Kale, and Julian Mcauley. 2022. Query-Aware Sequential Recommendation. In *Proceedings of the 31st ACM International Conference on Information &amp; Knowledge Management* (Atlanta, GA, USA) *(CIKM '22)*. Association for Computing Machinery, New York, NY, USA, 4019–4023.

[15] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2016. Session-based Recommendations with Recurrent Neural Networks. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, Yoshua Bengio and Yann LeCun (Eds.).

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

[17] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[18] Zihan Lin, Changxin Tian, Yupeng Hou, and Wayne Xin Zhao. 2022. Improving graph collaborative filtering with neighborhood-enriched contrastive learning. In *Proceedings of the ACM web conference 2022*. 2320–2329.

[19] Qijiong Liu, Hengchang Hu, Jiahao Wu, Jieming Zhu, Min-Yen Kan, and Xiao-Ming Wu. 2024. Discrete semantic tokenization for deep ctr prediction. In *Companion Proceedings of the ACM Web Conference 2024*. 919–922.

[20] Qidong Liu, Xian Wu, Yejing Wang, Zijian Zhang, Feng Tian, Yefeng Zheng, and Xiangyu Zhao. 2024. Llm-esr: Large language models enhancement for long-tailed sequential recommendation. *Advances in Neural Information Processing Systems* 37 (2024), 26701–26727.

[21] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. 2015. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*. 43–52.

[22] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748* (2018).

[23] Gustavo Penha, Ali Vardasbi, Enrico Palumbo, Marco De Nadai, and Hugues Bouchard. 2024. Bridging Search and Recommendation in Generative Retrieval: Does One Task Help the Other?. In *Proceedings of the 18th ACM Conference on Recommender Systems*. 340–349.

[24] Weicong Qin, Zelin Cao, Weijie Yu, Zihua Si, Sirui Chen, and Jun Xu. 2024. Explicitly integrating judgment prediction with legal document retrieval: a law-guided generative approach. In *Proceedings of the 47th international ACM SIGIR conference on research and development in information retrieval*. 2210–2220.

[25] Weicong Qin, Yi Xu, Weijie Yu, Chenglei Shen, Ming He, Jianping Fan, Xiao Zhang, and Jun Xu. 2025. MAPS: Motivation-Aware Personalized Search via LLM-Driven Consultation Alignment. *arXiv preprint arXiv:2503.01711* (2025).

[26] Weicong Qin, Yi Xu, Weijie Yu, Chenglei Shen, Xiao Zhang, Ming He, Jianping Fan, and Jun Xu. 2024. Enhancing sequential recommendations through multi-perspective reflections and iteration. *arXiv preprint arXiv:2409.06377* (2024).

[27] Weicong Qin, Yi Xu, Weijie Yu, Teng Shi, Chenglei Shen, Ming He, Jianping Fan, Xiao Zhang, and Jun Xu. 2025. Similarity= Value? Consultation Value Assessment and Alignment for Personalized Search. *arXiv preprint arXiv:2506.14437* (2025).

[28] Weicong Qin, Weijie Yu, Kepu Zhang, Haiyuan Zhao, Jun Xu, and Ji-Rong Wen. 2025. Uncertainty-aware evidential learning for legal case retrieval with noisy correspondence. *Information Sciences* 702 (2025), 121915.

[29] 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. *Advances in Neural Information Processing Systems* 36 (2023), 10299–10315.

[30] 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 Web Conference 2024*. 3464–3475.

[31] Chenglei Shen, Xiao Zhang, Teng Shi, Changshuo Zhang, Guofu Xie, and Jun Xu. 2024. A survey of controllable learning: Methods and applications in information retrieval. *arXiv preprint arXiv:2407.06083* (2024).

[32] Chenglei Shen, Jiahao Zhao, Xiao Zhang, Weijie Yu, Ming He, and Jianping Fan. 2024. Generating Model Parameters for Controlling: Parameter Diffusion for Controllable Multi-Task Recommendation. *arXiv preprint arXiv:2410.10639* (2024).

[33] Teng Shi, Weicong Qin, Weijie Yu, Xiao Zhang, Ming He, Jianping Fan, and Jun Xu. 2025. Bridging Search and Recommendation through Latent Cross Reasoning. *arXiv preprint arXiv:2508.04152* (2025).

[34] Teng Shi, Zihua Si, Jun Xu, Xiao Zhang, Xiaoxue Zang, Kai Zheng, Dewei Leng, Yanan Niu, and Yang Song. 2024. UniSAR: Modeling User Transition Behaviors between Search and Recommendation. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1029–1039.

[35] Teng Shi, Jun Xu, Xiao Zhang, Xiaoxue Zang, Kai Zheng, Yang Song, and Han Li. 2025. Retrieval Augmented Generation with Collaborative Filtering for Personalized Text Generation. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1294–1304.

[36] Teng Shi, Jun Xu, Xiao Zhang, Xiaoxue Zang, Kai Zheng, Yang Song, and Enyun Yu. 2025. Unified Generative Search and Recommendation. *arXiv preprint arXiv:2504.05730* (2025).

[37] Zihua Si, Xueran Han, Xiao Zhang, Jun Xu, Yue Yin, Yang Song, and Ji-Rong Wen. 2022. A Model-Agnostic Causal Learning Framework for Recommendation Using Search Data. In *Proceedings of the ACM Web Conference 2022* (Virtual Event, Lyon, France) *(WWW '22)*. Association for Computing Machinery, New York, NY, USA, 224–233.

[38] Zihua Si, Zhongxiang Sun, Xiao Zhang, Jun Xu, Xiaoxue Zang, Yang Song, Kun Gai, and Ji-Rong Wen. 2023. When Search Meets Recommendation: Learning Disentangled Search Representation for Recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2023, Taipei, Taiwan, July 23-27, 2023*. ACM, 1313–1323.

[39] 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* (Beijing, China) *(CIKM '19)*. ACM, New York, NY, USA, 1441–1450.

[40] Zhongxiang Sun, Kepu Zhang, Weijie Yu, Haoyu Wang, and Jun Xu. 2024. Logic rules as explanations for legal case retrieval. *arXiv preprint arXiv:2403.01457* (2024).

[41] Jiakai Tang, Sunhao Dai, Teng Shi, Jun Xu, Xu Chen, Wen Chen, Wu Jian, and Yuning Jiang. 2025. Think before recommend: Unleashing the latent reasoning power for sequential recommendation. *arXiv preprint arXiv:2503.22675* (2025).

[42] Jiakai Tang, Sunhao Dai, Zexu Sun, Xu Chen, Jun Xu, Wenhui Yu, Lantao Hu, Peng Jiang, and Han Li. 2024. Towards robust recommendation via decision boundary-aware graph contrastive learning. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 2854–2865.

[43] Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of machine learning research* 9, 11 (2008).

[44] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[45] Yuening Wang, Man Chen, Yaochen Hu, Wei Guo, Yingxue Zhang, Huifeng Guo, Yong Liu, and Mark Coates. 2024. Enhancing Click-through Rate Prediction in Recommendation Domain with Search Query Representation. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*. 2462–2471.

[46] Chuhan Wu, Fangzhao Wu, Mingxiao An, Tao Qi, Jianqiang Huang, Yongfeng Huang, and Xing Xie. 2019. Neural News Recommendation with Heterogeneous User Behavior. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics, Hong Kong, China, 4874–4883.

[47] Jiancan Wu, Xiang Wang, Fuli Feng, Xiangnan He, Liang Chen, Jianxun Lian, and Xing Xie. 2021. Self-supervised graph learning for recommendation. In *Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval*. 726–735.

[48] 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.

[49] Shitao Xiao, Zheng Liu, Peitian Zhang, Niklas Muennighoff, Defu Lian, and Jian-Yun Nie. 2024. C-pack: Packed resources for general chinese embeddings. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 641–649.

[50] Jiayi Xie, Shang Liu, Gao Cong, and Zhenzhong Chen. 2024. UnifiedSSR: A Unified Framework of Sequential Search and Recommendation. In *Proceedings of the ACM on Web Conference 2024*. 3410–3419.

[51] Xu Xie, Fei Sun, Zhaoyang Liu, Shiwen Wu, Jinyang Gao, Jiandong Zhang, Bolin Ding, and Bin Cui. 2022. Contrastive learning for sequential recommendation. In *2022 IEEE 38th international conference on data engineering (ICDE)*. IEEE, 1259–1273.

[52] Jing Yao, Zhicheng Dou, Ruobing Xie, Yanxiong Lu, Zhiping Wang, and Ji-Rong Wen. 2021. USER: A Unified Information Search and Recommendation Model Based on Integrated Behavior Sequence. In *Proceedings of the 30th ACM International Conference on Information J&amp; Knowledge Management* (Virtual Event, Queensland, Australia) *(CIKM '21)*. Association for Computing Machinery, New York, NY, USA, 2373–2382.

[53] Junliang Yu, Hongzhi Yin, Xin Xia, Tong Chen, Lizhen Cui, and Quoc Viet Hung Nguyen. 2022. Are graph augmentations necessary? simple graph contrastive learning for recommendation. In *Proceedings of the 45th international ACM SIGIR conference on research and development in information retrieval*. 1294–1303.

[54] Zhenrui Yue, Yueqi Wang, Zhankui He, Huimin Zeng, Julian McAuley, and Dong Wang. 2024. Linear recurrent units for sequential recommendation. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*. 930–938.

[55] Hamed Zamani and W. Bruce Croft. 2018. Joint Modeling and Optimization of Search and Recommendation. In *Proceedings of the First Biennial Conference on Design of Experimental Search & Information Retrieval Systems, Bertinoro, Italy, August 28-31, 2018 (CEUR Workshop Proceedings, Vol. 2167)*. CEUR-WS.org, 36–41.

[56] Hamed Zamani and W. Bruce Croft. 2020. Learning a Joint Search and Recommendation Model from User-Item Interactions. In *Proceedings of the 13th International Conference on Web Search and Data Mining* (Houston, TX, USA) *(WSDM '20)*. Association for Computing Machinery, New York, NY, USA, 717–725.

[57] Neil Zeghidour, Alejandro Luebs, Ahmed Omran, Jan Skoglund, and Marco Tagliasacchi. 2021. Soundstream: An end-to-end neural audio codec. *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 30 (2021), 495–507.

[58] Changshuo Zhang, Sirui Chen, Xiao Zhang, Sunhao Dai, Weijie Yu, and Jun Xu. 2024. Reinforcing Long-Term Performance in Recommender Systems with User-Oriented Exploration Policy. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1850–1860.

[59] Changshuo Zhang, Teng Shi, Xiao Zhang, Qi Liu, Ruobing Xie, Jun Xu, and Ji-Rong Wen. 2024. Modeling domain and feedback transitions for cross-domain sequential recommendation. *arXiv preprint arXiv:2408.08209* (2024).

[60] Changshuo Zhang, Teng Shi, Xiao Zhang, Yanping Zheng, Ruobing Xie, Qi Liu, Jun Xu, and Ji-Rong Wen. 2024. QAGCF: Graph Collaborative Filtering for Q&A Recommendation. *arXiv preprint arXiv:2406.04828* (2024).

[61] Changshuo Zhang, Xiao Zhang, Teng Shi, Jun Xu, and Ji-Rong Wen. 2025. Test-Time Alignment for Tracking User Interest Shifts in Sequential Recommendation. *arXiv preprint arXiv:2504.01489* (2025).

[62] Kepu Zhang, Teng Shi, Sunhao Dai, Xiao Zhang, Yinfeng Li, Jing Lu, Xiaoxue Zang, Yang Song, and Jun Xu. 2024. SAQRec: Aligning Recommender Systems to User Satisfaction via Questionnaire Feedback. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*. 3165–3175.

[63] Kepu Zhang, Zhongxiang Sun, Xiao Zhang, Xiaoxue Zang, Kai Zheng, Yang Song, and Jun Xu. 2025. Trigger3: Refining Query Correction via Adaptive Model Selector. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 39. 13260–13268.

[64] Kepu Zhang, Weijie Yu, Sunhao Dai, and Jun Xu. 2024. Citalaw: Enhancing llm with citations in legal domain. *arXiv preprint arXiv:2412.14556* (2024).

[65] Kepu Zhang, Weijie Yu, Zhongxiang Sun, and Jun Xu. 2025. Syler: A framework for explicit syllogistic legal reasoning in large language models. *arXiv preprint arXiv:2504.04042* (2025).

[66] Xiao Zhang, Teng Shi, Jun Xu, Zhenhua Dong, and Ji-Rong Wen. 2024. Model-agnostic causal embedding learning for counterfactually group-fair recommendation. *IEEE Transactions on Knowledge and Data Engineering* (2024).

[67] Yuting Zhang, Yiqing Wu, Ruidong Han, Ying Sun, Yongchun Zhu, Xiang Li, Wei Lin, Fuzhen Zhuang, Zhulin An, and Yongjun Xu. 2024. Unified Dual-Intent Translation for Jont Modeling of Search and Recommendation. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 6291–6300.

[68] Jujia Zhao, Wenjie Wang, Chen Xu, Xiuying Chen, Zhaochun Ren, and Suzan Verberne. 2025. Unifying Search and Recommendation: A Generative Paradigm Inspired by Information Theory. *arXiv preprint arXiv:2504.06714* (2025).

[69] Kai Zhao, Yukun Zheng, Tao Zhuang, Xiang Li, and Xiaoyi Zeng. 2022. Joint Learning of E-Commerce Search and Recommendation with a Unified Graph Neural Network. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining* (Virtual Event, AZ, USA) *(WSDM '22)*. Association for Computing Machinery, New York, NY, USA, 1461–1469.

[70] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223* (2023).

[71] Bowen Zheng, Yupeng Hou, Hongyu Lu, Yu Chen, Wayne Xin Zhao, Ming Chen, and Ji-Rong Wen. 2024. Adapting large language models by integrating collaborative semantics for recommendation. In *2024 IEEE 40th International Conference on Data Engineering (ICDE)*. IEEE, 1435–1448.

[72] Guorui Zhou, Na Mou, Ying Fan, Qi Pi, Weijie Bian, Chang Zhou, Xiaoqiang Zhu, and Kun Gai. 2019. Deep interest evolution network for click-through rate prediction. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 33. 5941–5948.

[73] Guorui Zhou, Xiaoqiang Zhu, Chenru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. 2018. Deep Interest Network for Click-Through Rate Prediction. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '18)*. 1059–1068.

[74] 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* (Virtual Event, Lyon, France) *(WWW '22)*. Association for Computing Machinery, New York, NY, USA, 2388–2399.