When Search Meets Recommendation: Learning Disentangled Search Representation for Recommendation

Zihua Si
Gaoling School of Artificial Intelligence
Renmin University of China
Beijing, China
zihua_si@ruc.edu.cn

Zhongxiang Sun
Gaoling School of Artificial Intelligence
Renmin University of China
Beijing, China
sunzhongxiang@ruc.edu.cn

Xiao Zhang[*]
Jun Xu[*]
Gaoling School of Artificial Intelligence
Renmin University of China
Beijing, China
{zhangx89,junxu}@ruc.edu.cn

Xiaoxue Zang
Yang Song
Kuaishou Technology Co., Ltd.
Beijing, China
{zangxiaoxue,yangsong}@kuaishou.com

Kun Gai
Unaffiliated
Beijing, China
gai.kun@qq.com

Ji-Rong Wen
Gaoling School of Artificial Intelligence
Renmin University of China
Beijing, China
jrwen@ruc.edu.cn

ABSTRACT

Modern online service providers such as online shopping platforms often provide both search and recommendation (**S&R**) services to meet different user needs. Rarely has there been any effective means of incorporating user behavior data from both S&R services. Most existing approaches either simply treat S&R behaviors separately, or jointly optimize them by aggregating data from both services, ignoring the fact that user intents in S&R can be distinctively different. In our paper, we propose a **S**earch-**E**nhanced framework for the **S**equential **Rec**ommendation (**SESRec**) that leverages users' search interests for recommendation, by disentangling similar and dissimilar representations within S&R behaviors. Specifically, SESRec first aligns query and item embeddings based on users' query-item interactions for the computations of their similarities. Two transformer encoders are used to learn the contextual representations of S&R behaviors independently. Then a contrastive learning task is designed to supervise the disentanglement of similar and dissimilar representations from behavior sequences of S&R. Finally, we extract user interests by the attention mechanism from three perspectives, *i.e.,* the contextual representations, the two separated behaviors containing similar and dissimilar interests. Extensive experiments on both industrial and public datasets demonstrate that SESRec consistently outperforms state-of-the-art models. Empirical studies further validate that SESRec successfully disentangle similar and dissimilar user interests from their S&R behaviors.

[*]Corresponding authors. Work partially done at Engineering Research Center of Next-Generation Intelligent Search and Recommendation, Ministry of Education.
Work done when Zihua Si and Zhongxiang Sun were interns at Kuaishou.

CCS CONCEPTS

• **Information systems** → **Recommender systems**.

KEYWORDS

Recommendation; Search; Contrastive Learning; Disentanglement Learning

ACM Reference Format:
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 '23), July 23–27, 2023, Taipei, Taiwan.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3539618.3591786

1 INTRODUCTION

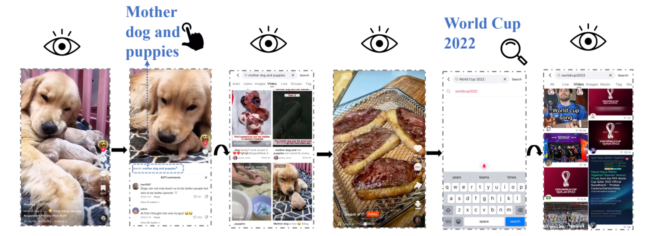
Recommender systems and search engines have been widely deployed in online platforms to help users alleviate information overload. Currently, with the vast increase of data on the Internet, solely using one of the recommender systems or the search engines cannot meet users' information needs. Hence, many social media platforms, *e.g.,* YouTube and TikTok, provide both search and recommendation (**S&R**) services for users to obtain information. As users express their diverse interests in both scenarios, it is feasible to enhance the recommendation system by jointly modeling the behaviors of both, and the core challenge is how to effectively leverage users' search interests for capturing accurate recommendation interests[1].

Early studies [28, 29] have demonstrated that jointly optimizing the S&R models benefits both performances. Recently, several works [5, 17, 24, 27, 31] have been proposed to boost the

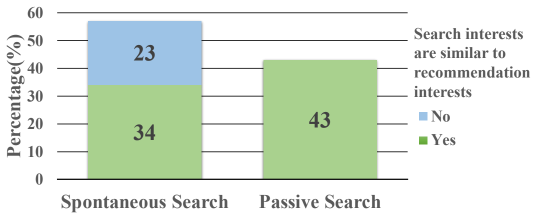
[1]In this paper, we use **recommendation interests** to refer to user interests captured by the recommendation system, and **search interests** to refer to users' interests revealed in their search history.

recommendation using search data. As such, devising a search-enhanced framework is a promising research area in the recommendation field. Due to that recommendation with search data is still a nascent research area in both academia and industry, recent works [5, 24, 27, 31] usually only incorporate users' S&R behaviors by feeding them into one encoder to mine users' interests.

Despite their effectiveness, most previous works ignore the differences between users' interests in S&R behaviors by modeling them without considering their correlations. However, in real-world applications, search behaviors may strengthen or be complementary to the interests revealed in the recommendation behaviors. For example, Figure 1(a) illustrates partial behavior histories of a user in the short-video scenario. While users browsing the items/video suggested by the recommendation system, they may spontaneously start searching by typing queries, which usually differ from the video content in the recommendation feed. We refer to such case as spontaneous search. In contrast, users may also start searching by clicking on the suggested query that is related to the current item/video being played, which we denote as passive search. To verify the universality of this phenomenon, we conducted data analysis from the real-world data collected from the Kuaishou[2] app, shown in Figure 1(b). The data analysis is based on behaviors of millions of users. For each search behavior, if the categories of the items exist in the set of categories of the items interacted by this user in the past seven days, this search behavior is similar to recent recommendation behaviors, and otherwise dissimilar. The similar search behaviors reflect users' strong interests overlapped in the recommendation behaviors and should be strengthened. The dissimilar behaviors may be undiscovered interests, which are probably newly emerging and unfulfilled in the recommendation feed. As a result, it is critical to disentangle the similar and dissimilar representations between S&R behaviors.



(a) An example of user S&R behaviors.



(b) Distribution of spontaneous and passive search w.r.t. recommendation interests.

Figure 1: S&R behaviors in the short-video scenario. (a) After watching a video about dogs, the user chooses to click on the suggested query (passive search) to explore more. Later, after watching a food video, the user searches "world cup 2022", a spontaneous search unrelated to the watched video. (b) Statistics of search behaviors collected from the Kuaishou app. 57% of the search behaviors are spontaneous and 43% are passive. 23% of the spontaneous searches have dissimilar interests to the recommendation interests.

To address such a problem, we devise a search-enhanced framework for sequential recommendation, namely SESRec, to learn the disentangled search representation for the recommendation. In details, in order to disentangle the similar and dissimilar interests between two behaviors, we propose to decompose each history sequence into two sub-sequences that respectively represent the similar and dissimilar interests so that we can extract user interests from multiple aspects. To learn the similarity between two behaviors, we first align query-item embeddings with an InfoNCE loss based on users' query-item interactions. Then two separate encoders are utilized to model S&R behaviors and generate their contextual representations. Due to the lack of the labels denoting the similarity of interests between the obtained contextual representations, we propose to leverage self-supervision to guide learning the similar and dissimilar interests. Specifically, we exploit the co-attention mechanism to learn the correlation between S&R's contextual representations. Based on the co-attention score, for both of the contextual representations, we not only aggregate them to generate anchors which are considered to maintain the shared interests of S&R, but also partition them into two sub-sequences, which are considered to represent the similar and dissimilar behaviors between S&R (respectively referred to as the positives and the negatives). Then following contrastive learning, we define a triplet loss to push the anchors closer to the positives than the negatives. Finally, we employ the attention mechanism to extract user interests from three aspects, *i.e.,* the contextual representations, the positives, and the negatives of S&R. In this way, the disentangled interests of S&R behaviors enhance the prediction for the next interaction.

The contributions of this paper are summarized as follows:

• To the best of our knowledge, it is the first time that users' S&R interests are jointly considered and disentangled into similar and dissimilar representations for user modeling. We pioneer the practice of learning disentangled search representations for the recommendation.

• We propose a search-enhanced framework for the sequential recommendation. By jointly considering users' S&R behaviors, we extract users' interests from multiple aspects by decomposing both S&R behaviors into two parts based on their co-attention scores: one for behaviors containing similar interests and the other for behaviors containing dissimilar interests. Moreover, we also utilize self-supervised learning to guide the decomposition.

• We conduct extensive experiments on two datasets. The experimental results validate the effectiveness of our proposed SESRec. In particular, SESRec outperforms traditional sequential models which do not leverage search data, as well as other search-aware models, which neglect the correlation between users' interests in S&R behaviors.

[2]https://www.kuaishou.com/en

2 RELATED WORK

Recommendation with Search Data. In both academia and industry, research that enhances recommendation using search data is relatively rare. Only a few works involve in this area. Zamani and Croft [28, 29] assume that S&R models could potentially benefit from each other by jointly training with both S&R data. Yao et al. [27] design an approach called USER that mines user interests from the integrated user behavior sequences and accomplishes these two tasks in a unified way. NRHUB [24] exploits heterogeneous user behaviors, *i.e.,* webpage browsing and search queries, to enhance the recommendation model. IV4REC [17] and IV4REC+ [18] leverages search queries as instrumental variables to reconstruct user and item embeddings and boost recommendation performance in a causal learning manner. Query-SeqRec [5] is a query-aware model, which incorporates issued queries and browsing items to capture users' interests and intents. SRJGraph [31] is a GNN-based method which incorporates queries into user-item interaction edges as attributes. In this work, we also develop a framework to learn disentangled search representation for recommendation.

Sequential Recommendation. Sequential recommendation methods mine user interests by modeling sequential relationships of user behaviors. An early work [6] first utilizes the GRU mechanism to model user preferences. And attention mechanisms are introduced to capture sequential patterns, such as STAMP [12]. There are works using CNN architectures, *e.g.,* Caser [20] treats the historical item sequence as an "image" and adopts a CNN for user modeling. For other neural network architectures, several models employ GNN [2, 26] which construct a graph for historical sequences. Currently, lots of models leverage the transformer architecture, *e.g.,* SASRec [8] and BERT4Rec [19]. Several works [33, 34] devise an attention mechanism to adaptively learn user interests from behaviors. FMLP-Rec [36] is a state-of-the-art that leverages an all-MLP architecture with learnable filters for sequential recommendation. Unlike these works, this work incorporates users' search activities into the sequential recommendation task.

Contrastive Learning for Recommendation. With the successful development of contrastive learning, this technique has been widely adopted in recommendation [7, 13, 32, 35]. As for sequential recommendation, Zhou et al. [35] first devise auxiliary self-supervised objectives to enhance data representations via pretraining methods. Ma et al. [13] propose a sequence-to-sequence training strategy by performing self-supervised learning in the latent space. Recently, several works [30, 32] are proposed for learning users' diverse interests. For example, Zhang et al. [30] propose a contrastive learning framework to disentangle long and short-term interests for recommendation with self-supervision. In this work, we propose to disentangle the similar and dissimilar interests between S&R behaviors with self-supervision signals.

3 PROBLEM FORMULATION

Assume that the sets of users, items, and queries are denoted by $\mathcal{U}$, $\mathcal{I}$, and $\mathcal{Q}$ respectively, where $u \in \mathcal{U}$ denotes a user, $i \in \mathcal{I}$ denotes an item, and $q \in \mathcal{Q}$ denotes a query. For the recommendation history, a user u has a context of chronologically ordered item interactions: $S_i^u = [i_1, i_2, \dots, i_{T_r}]$, where S_i^u is u's interacted item sequence, T_r is the number of items that user u has interacted till timestamp t, and i_k is the k-th interacted item. For the search history, a user u has a context of chronologically ordered issued queries: $S_q^u = [q_1, q_2, \dots, q_{T_s}]$, where S_q^u is u's issued query sequence, T_s is the number of queries issued before t, and q_k is the k-th issued query. When using the search service, user u also clicks items after issuing queries: $S_c^u = \left[i_{q_1}^{(1)}, i_{q_1}^{(2)}, i_{q_2}^{(1)}, \dots, i_{T_s}^{(1)}, i_{T_s}^{(2)}, i_{T_s}^{(3)}\right]$, where S_c^u is the u's clicked item sequence corresponding to S_q^u, and $i_{q_k}^{(j)}$ is the j-th clicked item under query q_k. The number of the clicked items corresponding to each query can be different and at minimum 0.

Based on the above notations, we define the task of sequential recommendation with search data. Given the contextual sequences S_i^u, S_q^u and S_c^u of a user's recommendation and search histories, the sequential recommendation with search data task aims to predict the next item that the user is likely to interact with at timestamp $t+1$, *i.e.,* $P(i_{t+1} \mid S_i^u, S_q^u, S_c^u)$. Note that it differs from the conventional sequential recommendation task, which predicts $P(i_{t+1} \mid S_i^u)$.

4 OUR APPROACH: SESREC

In this section, we elaborate on the proposed SESRec.

4.1 Overview

The overview of SESRec is illustrated in Figure 2. First, we embed the sparse input data into dense representations. Then, we leverage the transformer layers [23] to learn the contextual representations of historical behaviors. For disentangling interests, we separate two behavior sequences into sub-sequences that respectively represent similar and dissimilar interests. And we aggregate behavior sequences into vectors to represent user interests w.r.t. the candidate item. Finally, we concatenate all the vectors together to get the overall representation vector, which is followed by an Multilayer Perceptron (MLP) to generate the ultimate prediction.

Specifically, we design several components to disentangle user interests with self-supervision and aggregate user interests from all aspects. These designed components are shown in Figure 2 within colored boxes. We align query and item representations into the same semantic space with the InfoNCE loss. Then we separate S&R behavior sequences into sub-sequences respectively. We leverage self-supervision signals to guide the separation based on the triplet loss. Last, we introduce an interest extraction module that aggregates the original sequences and constructed sub-sequences to form aggregated, similar and dissimilar interest representations of both behaviors.

4.2 Encoding Sequential Behaviors

4.2.1 Embedding Layer. We maintain separate look-up tables for IDs and attributes of users, items, and queries. As for users and items, given a user (an item), we concatenate his (its) ID and attribute embeddings to form a user (item) representation: $\mathbf{e}^u = \mathbf{e}^{\mathrm{ID}_u} \| \mathbf{e}^{a_1} \| \cdots \| \mathbf{e}^{a_n}$ ($\mathbf{e}^i = \mathbf{e}^{\mathrm{ID}_i} \| \mathbf{e}^{b_1} \| \cdots \| \mathbf{e}^{b_m}$), where $\|$ denotes concatenation, $a_1, \dots, a_n$ and $b_1, \dots, b_m$ denote the attributes of users and items, respectively. As for queries, each query q contains several terms $(w_1, w_2, \dots, w_{|q|})$. We obtain the query embedding by concatenating query ID embedding and the mean pooling of term embeddings: $\mathbf{e}^q = \mathbf{e}^{\mathrm{ID}_q} \| \mathrm{MEAN}(\mathbf{e}^{w_1}, \mathbf{e}^{w_2}, \dots, \mathbf{e}^{w_{|q|}})$, where $\mathbf{e}^{\mathrm{ID}_q}$ is

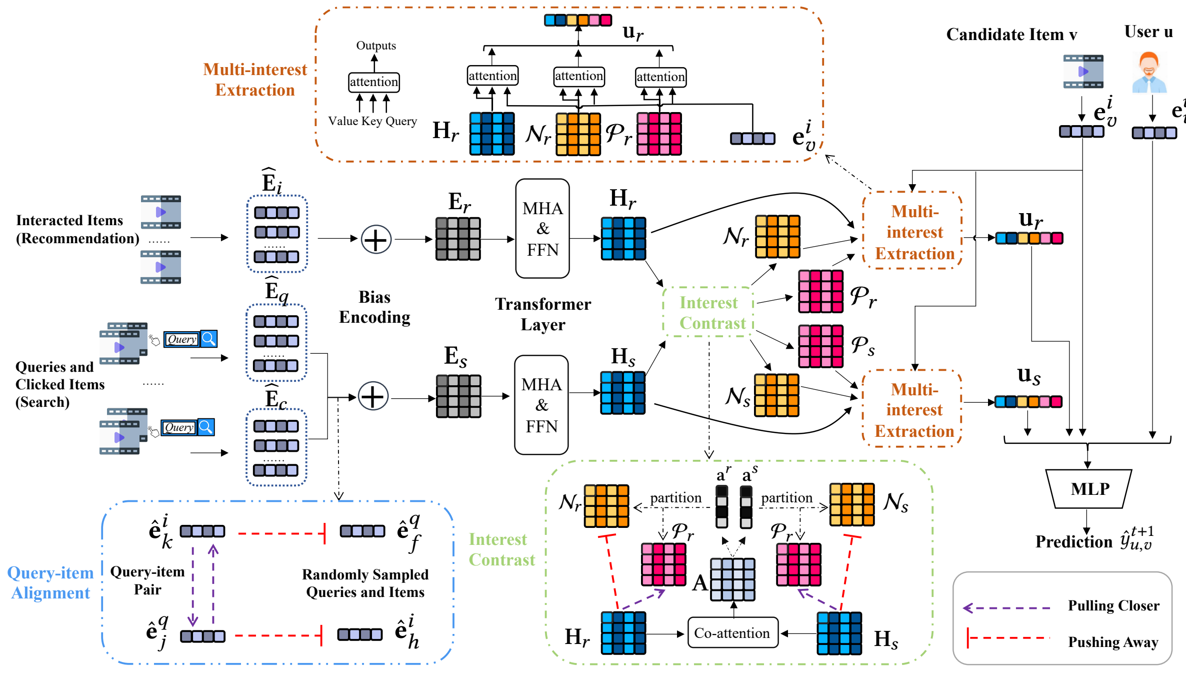


Figure 2: The architecture of SESRec. From left to right is the process of modeling S&R histories. On the far right is the process of ultimate prediction. The three colored modules with dashed lines conduct interest disentanglement.

the query ID embedding, $\mathbf{e}^{w_k}$ is the ID embedding of the k-th term, and MEAN() denotes mean pooling. Many queries occur repeatedly in search data, so query ID is informative. And most queries consist of less than five terms and lack strong sequential patterns. So the average pooling operation, following the bag-of-words paradigm, is effective and efficient.

For a user u, given the context S_i^u, S_q^u, and S_c^u, we obtain embedding matrices of historically interacted items, issued queries, and clicked items, denoted as $\mathbf{E}_i = [\mathbf{e}_1^i, \mathbf{e}_2^i, \ldots, \mathbf{e}_{T_r}^i]^\intercal \in \mathbb{R}^{T_r \times d_i}$, $\mathbf{E}_q = [\mathbf{e}_1^q, \mathbf{e}_2^q, \ldots, \mathbf{e}_{T_s}^q]^\intercal \in \mathbb{R}^{T_s \times d_q}$, and $\mathbf{E}_c = [\mathbf{e}_1^i, \mathbf{e}_2^i, \ldots, \mathbf{e}_{|S_c^u|}^i]^\intercal \in \mathbb{R}^{|S_c^u| \times d_i}$, respectively, where d_i and d_q are dimensions of item and query embeddings. Besides, we incorporate learnable position encoding matrices to model the sequential order of behaviors, denoted as $\mathbf{P}_s \in \mathbb{R}^{T_s \times d}$ and $\mathbf{P}_r \in \mathbb{R}^{T_r \times d}$, respectively, where d is dimension. As for search behaviors, we also adopt type embedding for queries of different sources, *e.g.,* user-typed queries, user-specific historical queries, and queries related to the current item. The search type embedding matrix is denoted by $\mathbf{M}_s \in \mathbb{R}^{k \times d}$, where k is the total number of all possible search sources.

Because it is challenging to model user interests with query and item representations in unaligned vector spaces, we transform the item and query embeddings into latent vector spaces with the same dimension. The transformed embedding matrices of interacted items, issued queries, and clicked items are calculated as:

$$\widehat{\mathbf{E}}_i = \mathbf{E}_i \mathbf{W}_i, \quad \widehat{\mathbf{E}}_q = \mathbf{E}_q \mathbf{W}_q, \quad \widehat{\mathbf{E}}_c = \mathbf{E}_c \mathbf{W}_i, \tag{1}$$

where $\widehat{\mathbf{E}}_i \in \mathbb{R}^{T_r \times d}$, $\widehat{\mathbf{E}}_q \in \mathbb{R}^{T_s \times d}$ and $\widehat{\mathbf{E}}_c \in \mathbb{R}^{|S_c^u| \times d}$ are transformed matrices, $\mathbf{W}_i \in \mathbb{R}^{d_i \times d}$ and $\mathbf{W}_q \in \mathbb{R}^{d_q \times d}$ are trainable parameters for linear projection.

4.2.2 *Bias Encoding.* We incorporate position encodings for S&R behaviors to make use of the order relations of the sequence. The recommendation sequence matrix $\mathbf{E}_r \in \mathbb{R}^{T_r \times d}$ is obtained by summing the interacted item matrix and the position matrix:

$$\mathbf{E}_r = \widehat{\mathbf{E}}_i + \mathbf{P}_r \tag{2}$$

As for search behaviors, we additionally introduce type encodings along with position encodings. The search sequence matrix $\mathbf{E}_s \in \mathbb{R}^{T_s \times d}$ is defined as:

$$\mathbf{E}_s = \widehat{\mathbf{E}}_q + \widetilde{\mathbf{E}}_c + \mathbf{P}_s + \widehat{\mathbf{M}}_s \tag{3}$$

where $\widetilde{\mathbf{E}}_c \in \mathbb{R}^{T_s \times d}$ is the matrix of the mean pooling of all clicked items' matrices under each query, and $\widehat{\mathbf{M}}_s \in \mathbb{R}^{T_s \times d}$ is the type matrix.

To obtain $\widetilde{\mathbf{E}}_c$, we first group clicked items $\widehat{\mathbf{E}}_c$ by the issued queries, *i.e.,* several items clicked by the same query are divided into the same group. Then we apply a mean pooling operation on each group to get the matrix $\widetilde{\mathbf{E}}_c \in \mathbb{R}^{T_s \times d}$.

The type matrix $\widehat{\mathbf{M}}_s$ is defined as a sequence of type embeddings for each query in the search history, where each element $\mathbf{m}_j \in \mathbb{R}^d$ in $\widehat{\mathbf{M}}_s$ is obtained from the look-up table $\mathbf{M}_s$. We add the type matrix to model the correlation between search behaviors and search sources.

Considering clicked items contain identical users' interests as their corresponding queries, we fuse the issued query sequence and clicked item sequence to form a unified search sequence by adding them up in Equation (3).

4.2.3 Transformer Layer. To learn an enhanced contextual representation for each element in a given sequence, we use the transformer layer [23] to capture the relations between each element with other elements in the S&R sequences. The transformer layer generally consists of two sub-layers, *i.e.,* a multi-head self-attention layer and a point-wise feed-forward network. We apply the transformer layers for S&R sequences, respectively:

$$\mathbf{F}_s = \mathrm{MHA}_s(\mathbf{E}_s), \quad \mathbf{F}_r = \mathrm{MHA}_r(\mathbf{E}_r), \tag{4}$$

$$\mathbf{H}_s = \mathrm{FFN}_s(\mathbf{F}_s), \quad \mathbf{H}_r = \mathrm{FFN}_r(\mathbf{F}_r), \tag{5}$$

where $\mathbf{H}_s \in \mathbb{R}^{T_s \times d}$ and $\mathbf{H}_r \in \mathbb{R}^{T_r \times d}$ denote enhanced matrices of S&R sequences respectively, the multi-head self-attention is abbreviated to "MHA", and the two-layer feed-forward network is abbreviated to "FFN".

4.3 Self-supervised Interest Disentanglement

As mentioned before, user interests between S&R behaviors have overlaps and differences. Since there does not exist any annotated label of user interests, we leverage contrastive learning techniques to disentangle the S&R behaviors with self-supervision and then extract user interests from three aspects, *i.e.,* the aggregated behaviors, the two separated behaviors containing similar and dissimilar interests.

4.3.1 Query-item Alignment. It is challenging for the behavior encoders to jointly learn user interests from S&R behaviors that have unaligned embeddings. Also, it is unfeasible to disentangle user interests from S&R behaviors without knowing the semantic similarities between queries and items. Thus, we align the embeddings of queries and items as follows before further extracting user interests from them.

Because items and queries have different forms of features, their original embeddings are unaligned in different vector spaces. As shown in Equation (1), we first transform the item and query embeddings into a latent vector space. Then, inspired by works [10, 16] for multi-model learning, we leverage a contrastive learning loss to teach the model which queries and items are similar or different. Given issued query and clicked item sequence matrices $\widehat{\mathbf{E}}_q = [\hat{\mathbf{e}}^q_1, \hat{\mathbf{e}}^q_2, \dots, \hat{\mathbf{e}}^q_{T_s}]^\intercal \in \mathbb{R}^{T_s \times d}$ and $\widehat{\mathbf{E}}_c = [\hat{\mathbf{e}}^i_1, \hat{\mathbf{e}}^i_2, \dots, \hat{\mathbf{e}}^i_{|S^u_c|}]^\intercal \in \mathbb{R}^{|S^u_c| \times d}$, we minimize the sum of two InfoNCE [21] losses: one for query-to-item alignment

$$\mathcal{L}^{u,t}_{\mathrm{A_{q2i}}} = -\sum_{j=1}^{T_s} \sum_{k=1}^{|q_j|} \log \frac{\exp(s(\hat{\mathbf{e}}^q_j, \hat{\mathbf{e}}^i_k)/\tau)}{\sum_{h \in \mathcal{I}_{\mathrm{neg}}} \exp(s(\hat{\mathbf{e}}^q_j, \hat{\mathbf{e}}^i_h)/\tau)}, \tag{6}$$

and the other for item-to-query alignment

$$\mathcal{L}^{u,t}_{\mathrm{A_{i2q}}} = -\sum_{j=1}^{T_s} \sum_{k=1}^{|q_j|} \log \frac{\exp(s(\hat{\mathbf{e}}^q_j, \hat{\mathbf{e}}^i_k)/\tau)}{\sum_{f \in \mathcal{Q}_{\mathrm{neg}}} \exp(s(\hat{\mathbf{e}}^q_f, \hat{\mathbf{e}}^i_k)/\tau)}, \tag{7}$$

where τ is a learnable temperature parameter, $|q_j|$ denotes the number of clicked items of query q_j which satisfies $\sum_{j=1}^{T_s} |q_j| = |S^u_c|$, $\mathcal{I}_{\mathrm{neg}}$ and $\mathcal{Q}_{\mathrm{neg}}$ denote the sets of randomly sampled items and queries respectively, and s is a similarity function. The function s is defined as: $s(\mathbf{p}, \mathbf{q}) = \tanh(\mathbf{p}^\intercal \mathbf{W}_{\mathrm{A}} \mathbf{q})$, where tanh denotes the activation function and the introduction of $\mathbf{W}_{\mathrm{A}} \in \mathbb{R}^{d \times d}$ ensures the query-item correlation estimation can be different with the criterion used in the ultimate prediction. Finally, the query-item alignment loss is obtained by:

$$\mathcal{L}^{u,t}_{\mathrm{ali}} = \frac{1}{2}(\mathcal{L}^{u,t}_{\mathrm{A_{q2i}}} + \mathcal{L}^{u,t}_{\mathrm{A_{i2q}}}), \tag{8}$$

4.3.2 Interest Contrast. To conduct interest disentanglement, we employ a contrastive learning mechanism to distinguish similar and dissimilar interests between the contextual representations of behaviors $\mathbf{H}_s$ and $\mathbf{H}_r$.

After the transformer layers, given the matrices $\mathbf{H}_s$ and $\mathbf{H}_r$, we construct a co-dependant representation matrix of both behaviors, which generates the similarity scores of two sequences. Inspired by recent works [15, 25] for question answering, we leverage the co-attention technique. We first compute an affinity matrix $\mathbf{A} \in \mathbb{R}^{T_s \times T_r}$ as follows:

$$\mathbf{A} = \tanh(\mathbf{H}_s \mathbf{W}_l (\mathbf{H}_r)^{\mathrm{T}}), \tag{9}$$

where $\mathbf{W}_l \in \mathbb{R}^{d \times d}$ is a learnable weight matrix. The affinity matrix $\mathbf{A}$ contains affinity scores corresponding to all pairs of recommendation behaviors and search behaviors. We multiply the affinity matrix $\mathbf{A}$ and the search matrix $\mathbf{H}_s$ (or the recommendation matrix $\mathbf{H}_r$), and then normalize the multiplication results to get similarity scores for each element in one sequence across all the elements in the other sequence:

$$\mathbf{a}^s = \mathrm{softmax}(\mathbf{W}_r \mathbf{H}_r^{\mathrm{T}} \mathbf{A}^{\mathrm{T}}), \quad \mathbf{a}^r = \mathrm{softmax}(\mathbf{W}_s \mathbf{H}_s^{\mathrm{T}} \mathbf{A}), \tag{10}$$

where $\mathbf{a}^r \in \mathbb{R}^{T_r}$ and $\mathbf{a}^s \in \mathbb{R}^{T_s}$ are similarity scores, $\mathbf{W}_r, \mathbf{W}_s \in \mathbb{R}^{1 \times d}$ are trainable parameters for linear projection.

Next, we exploit a triplet loss to self-supervise the disentanglement of similar and dissimilar interests between two behaviors. Given similarity scores $\mathbf{a}^r$ and $\mathbf{a}^s$, elements in $\mathbf{H}_s$ and $\mathbf{H}_r$ with higher scores can be interpreted as representative ones for similar interests, while elements with lower scores can be interpreted as representative ones for dissimilar interests. Let $\mathcal{P}$ ($\mathcal{N}$) denote the set of elements containing similar (dissimilar) interests of S&R behaviors. As such, we perform hard selection to separate S&R sequences into two subsequences as follows:

$$\mathcal{P}_s = \{\mathbf{h}^s_j \mid \mathbf{a}^s_j > \gamma_s\}, \quad \mathcal{N}_s = \{\mathbf{h}^s_j \mid \mathbf{a}^s_j \le \gamma_s\}, \tag{11}$$

$$\mathcal{P}_r = \{\mathbf{h}^r_j \mid \mathbf{a}^r_j > \gamma_r\}, \quad \mathcal{N}_r = \{\mathbf{h}^r_j \mid \mathbf{a}^r_j \le \gamma_r\}, \tag{12}$$

where $\mathbf{h}^s_j, \mathbf{h}^r_j \in \mathbb{R}^d$ are the j-th vectors in matrices $\mathbf{H}_s$ and $\mathbf{H}_r$, $\mathbf{a}^s_j$ and $\mathbf{a}^r_j$ are similarity scores for $\mathbf{h}^s_j$ and $\mathbf{h}^r_j$ respectively, γ_r and γ_s are selection thresholds. Since $\mathbf{a}^s$ and $\mathbf{a}^r$ are normalized after softmax, we empirically set the thresholds γ_r and γ_s to the uniform values $\frac{1}{T_r}$ and $\frac{1}{T_s}$. The positives with similarity scores larger than the thresholds can be interpreted as similar interests with above-average similarities. The negatives, as the counterparts of positives, are with below-average similarities.

Then we design the anchors, positives and negatives of the triplet loss. To guide learning the disentanglement, we utilize the original sequences $\mathbf{H}_r$ and $\mathbf{H}_s$ to form anchors, and leverage the separated subsequences $\mathcal{P}_s, \mathcal{P}_r$ and $\mathcal{N}_s, \mathcal{N}_r$ to serve as positives and negatives. The anchors, positives, and negatives can be calculated as:

$$\mathbf{i}^A_s = \sum_{j=1}^{T_s} \mathbf{a}^s_j \mathbf{h}^s_j, \quad \mathbf{i}^P_s = \mathrm{MEAN}(\mathcal{P}_s), \quad \mathbf{i}^N_s = \mathrm{MEAN}(\mathcal{N}_s), \tag{13}$$

$$\mathbf{i}_r^A = \sum_{j=1}^{T_r} \mathbf{a}_j^r \mathbf{h}_j^r, \quad \mathbf{i}_r^P = \text{MEAN}(\mathcal{P}_r), \quad \mathbf{i}_r^N = \text{MEAN}(\mathcal{N}_r), \tag{14}$$

where $\mathbf{i}_s^A, i_r^A \in \mathbb{R}^d$ are anchors, $\mathbf{i}_s^P, \mathbf{i}_r^P \in \mathbb{R}^d$ are positives, $\mathbf{i}_s^N, \mathbf{i}_r^N \in \mathbb{R}^d$ are negatives. Then we perform contrastive learning, which requires the anchors to be similar with positives, and to be different from negatives. Based on these vectors, We implement triplet losses for S&R behaviors, respectively. Formally, the loss function is computed as follows:

$$\mathcal{L}_{\text{tri}}(a, p, n) = \max\{d(a, p) - d(a, n) + m, 0\}, \tag{15}$$

where d denotes distance function which is implemented as euclidean distance, m denotes a positive margin value, a, p and n denote anchors, positives and negatives, respectively. Finally, the interest contrast loss can be obtained by summing up two triplet losses, one for recommendation behaviors, the other for search behaviors:

$$\mathcal{L}_{\text{con}}^{u,t} = \mathcal{L}_{\text{tri}}(\mathbf{i}_r^A, \mathbf{i}_r^P, \mathbf{i}_r^N) + \mathcal{L}_{\text{tri}}(\mathbf{i}_s^A, \mathbf{i}_s^P, \mathbf{i}_s^N), \tag{16}$$

Remark. In most cases, users use S&R services at different frequencies. The lengths and update frequencies of two behaviors are different since they are collected from different services. That is why we employ triplet losses for the two behaviors, respectively. We update the model parameters of each behavior with its own constructed interest representations, which ensures the consistency of model training. Besides, considering that similar and dissimilar interests usually overlap with each other to some extent, there is no clear distinction between them. The triplet loss performs pairwise comparisons, which reduces the differences between similar things and increases the differences between different things. That is why we use the triplet loss instead of other contrastive loss functions, *e.g.,* InfoNCE [21], which imposes too strong punishment on the similarity between positives and negatives.

4.3.3 Multi-interest Extraction. Based on the original behaviors and separated behaviors containing similar and dissimilar interests, we extract user interests from three aspects, *i.e.,* aggregated, similar, and dissimilar interests. Given a candidate item v, we utilize an attention mechanism to reallocate the user interests w.r.t. the candidate item. For recommendation behaviors, interests can be extracted from three aspects as follows:

$$\mathbf{u}_r^{\text{all}} = \sum_{j=1}^{T_r} a_j^{\text{all}} \mathbf{h}_j^r, \quad a_j^{\text{all}} = \frac{\exp((\mathbf{h}_j^r)^{\mathrm{T}} \mathbf{W}_d \mathbf{e}_v^i)}{\sum_{k=1}^{T_r} \exp((\mathbf{h}_k^r)^{\mathrm{T}} \mathbf{W}_d \mathbf{e}_v^i)}, \tag{17}$$

$$\mathbf{u}_r^{\text{sim}} = \sum_{\mathbf{h}_j^r \in \mathcal{P}_r} a_j^{\text{sim}} \mathbf{h}_j^r, \quad a_j^{\text{sim}} = \frac{\exp((\mathbf{h}_j^r)^{\mathrm{T}} \mathbf{W}_d \mathbf{e}_v^i)}{\sum_{\mathbf{h}_k^r \in \mathcal{P}_r} \exp((\mathbf{h}_k^r)^{\mathrm{T}} \mathbf{W}_d \mathbf{e}_v^i)}, \tag{18}$$

$$\mathbf{u}_r^{\text{diff}} = \sum_{\mathbf{h}_j^r \in \mathcal{N}_r} a_j^{\text{diff}} \mathbf{h}_j^r, \quad a_j^{\text{diff}} = \frac{\exp((\mathbf{h}_j^r)^{\mathrm{T}} \mathbf{W}_d \mathbf{e}_v^i)}{\sum_{\mathbf{h}_k^r \in \mathcal{N}_r} \exp((\mathbf{h}_k^r)^{\mathrm{T}} \mathbf{W}_d \mathbf{e}_v^i)}, \tag{19}$$

where $\mathbf{u}_r^{\text{all}}, \mathbf{u}_r^{\text{sim}}, \mathbf{u}_r^{\text{diff}} \in \mathbb{R}^d$ are representative vectors for aggregated, similar, and dissimilar interests, $\mathbf{W}_d$ is the trainable parameters to model correlation between recommendation behaviors and the candidate item. By concatenating these three vectors, we can get the representation of recommendation interests:

Table 1: Statistics of datasets used in this paper. 'S' and 'R' denote search and recommendation, respectively.

Dataset	Users	Items	Queries	Actions-S	Actions-R
Kuaishou	35,721	822,832	398,924	922,531	11,381,172
Amazon	68,223	61,934	4,298	934,664	989,618

$$\mathbf{u}_r = \mathbf{u}_r^{\text{all}} \| \mathbf{u}_r^{\text{sim}} \| \mathbf{u}_r^{\text{diff}}, \tag{20}$$

where $\mathbf{u}_r \in \mathbb{R}^{3d}$. Similarly, we can obtain the representation of search interests in the same way, *i.e.,* $\mathbf{u}_s \in \mathbb{R}^{3d}$.

4.4 Prediction and Model Training

4.4.1 Prediction. To predict the interaction, we utilize the widely adopted two-layer MLP [33, 34] to model feature interaction and make predictions. Given a user u and an item v at timestamp $t+1$, the prediction score can be calculated as follows:

$$\hat{y}_{u,v}^{t+1} = \text{MLP}(\mathbf{u}_r \| \mathbf{u}_s \| \mathbf{e}_v^i \| \mathbf{e}_u^u), \tag{21}$$

where $\hat{y}_{u,v}^{t+1}$ denotes the prediction score, $\mathbf{e}_v^i$ and $\mathbf{e}_u^u$ are embeddings of the item v and the user u, respectively.

4.4.2 Model Training. Following the existing works' settings [33, 34], we adopt the negative log-likelihood function to supervise the final prediction:

$$\mathcal{L}_{\text{rec}}^{u,t} = -\frac{1}{N} \sum_{v \in O} y_{u,v}^{t+1} \log(\hat{y}_{u,v}^{t+1}) + (1 - y_{u,v}^{t+1}) \log(1 - \hat{y}_{u,v}^{t+1}), \tag{22}$$

where O is the set composed of training pairs of one positive item and $N-1$ negative items. In order to apply additional self-supervised signals about query-item alignment and interest disentanglement, we train our model in an end-to-end manner under a multi-task learning schema. The overall loss function is formulated as:

$$\mathcal{L} = \sum_{u=1}^{|\mathcal{U}|} \sum_{t=1}^{T_u} (\mathcal{L}_{\text{rec}}^{u,t} + \alpha \mathcal{L}_{\text{ali}}^{u,t} + \beta \mathcal{L}_{\text{con}}^{u,t}) + \lambda ||\Theta||_2. \tag{23}$$

where $|\mathcal{U}|$ is the number of users, T_u denotes the timestamp of the user u's latest interaction, α and β are hyper-parameters for additional tasks, and $\lambda||\Theta||_2$ denotes the L_2 regularization to avoid over-fitting.

5 EXPERIMENT

5.1 Experimental Setup

5.1.1 Dataset. SESRec needs user S&R behavior logs simultaneously. In the following experiments, we evaluated models on two datasets: one is collected from logs of a short-video app, and the other is based on a widely used public Amazon dataset [4, 14]. Table 1 reports statistics of both datasets.

Kuaishou Dataset: This dataset is constructed based on behavior logs of 35,721 users who elected to use both S&R services on the short-video app named Kuaishou over one month in 2022. The historical S&R behaviors have been recorded. For dataset preprocessing, following the common practice in [8, 19, 36], we group interaction records by users, sort them by timestamp ascendingly and filter unpopular items and users with fewer than five interaction records.

Table 2: Overall performance comparisons on both datasets. The best and the second-best performance methods are denoted in bold and underlined fonts respectively. * means improvements over the second-best methods are significant (*p*-value < 0.01).

Dataset		Kuaishou						Amazon (Kindle Store)					
Category	Method	NDCG@5	NDCG@10	HIT@1	HIT@5	HIT@10	MRR	NDCG@5	NDCG@10	HIT@1	HIT@5	HIT@10	MRR
Sequential	STAMP	0.2544	0.2981	0.1413	0.3616	0.4970	0.2569	0.2612	0.3103	0.1336	0.3833	0.5352	0.2608
	DIN	0.2969	0.3418	0.1792	0.4092	0.5484	0.2976	0.2999	0.3495	0.1591	0.4340	0.5871	0.2942
	GRU4Rec	0.3247	0.3688	0.1890	0.4517	0.5881	0.3180	0.3099	0.3662	0.1479	0.4648	0.6388	0.2993
	SASRec	0.3252	0.3693	0.1904	0.4501	0.5864	0.3187	0.3822	0.4312	0.2187	0.5324	0.6838	0.3675
	DIEN	0.3217	0.3704	0.1914	0.4463	0.5969	0.3192	0.3336	0.3803	0.1871	0.4706	0.6150	0.3246
	FMLP-Rec	0.3354	0.3787	0.1953	0.4651	0.5988	0.3270	0.4073	0.4550	0.2349	0.5651	**0.7121**	0.3883
Search-aware	NRHUB	0.2964	0.3431	0.1665	0.4199	0.5647	0.2933	0.2744	0.3265	0.1329	0.4099	0.5708	0.2704
	JSR	0.3015	0.3513	0.1738	0.4241	0.5783	0.3004	0.3221	0.3722	0.2057	0.4386	0.5937	0.3224
	IV4REC	0.3114	0.3591	0.1877	0.4282	0.5761	0.3116	0.3473	0.3960	0.1853	0.4985	0.6258	0.3331
	Query-SeqRec	0.3117	0.3581	0.1740	0.4412	0.5844	0.3055	0.3692	0.4142	0.2187	0.5083	0.6470	0.3572
	SRJGraph	0.3297	0.3762	0.2046	0.4479	0.5917	0.3277	0.3670	0.4043	**0.2760**	0.4898	0.6242	0.3708
	SESRec	**0.3541***	**0.4054***	**0.2173***	**0.4848***	**0.6436***	**0.3490***	**0.4224***	**0.4663***	0.2580	**0.5723***	0.7074	**0.4046***

Amazon Dataset[3]: To the best of our knowledge, there doesn't exist a public dataset that contains both S&R behaviors. Following a standard approach for product search [1], we enhance a recommendation dataset, the Amazon dataset [4, 14], by generating search behaviors. We adopt the "Kindle Store" subset of the five-core Amazon dataset that covers data in which all users and items have at least five reviews. The detailed generation process of search behaviors[4] can be found in [1]. Please note that this automatically constructed search dataset has been widely used by product search community [1, 3, 22, 29]. Following [3], we randomly select one query for items with multiple constructed queries to model the sequential behaviors.

Following previous works [8, 19, 36], we adopt the *leave-one-out* strategy to split both datasets. For each user, we hold out the most recent action for testing, the second most recent action for validation, and all the remaining actions for training. Besides, to ensure data quality, we filter interactions in which the user doesn't have historical S&R history simultaneously.

5.1.2 *Evaluation Metrics.*

Following [19, 36], we employ several widely used ranking metrics, including *Hit Ratio* (HIT), *Normalized Discounted Cumulative Gain* (NDCG), and *Mean Reciprocal Rank* (MRR). We report HIT with $k = 1, 5, 10$ and NDCG with $k = 5, 10$. We pair the ground-truth item with 99 randomly sampled items that the user has never interacted with. For all metrics, we calculate them according to the ranking of items and report average results.

5.1.3 *Baseline Models.*

In this work, we compare SESRec with state-of-the-art methods.

For sequential recommendation methods without leveraging search data, we include following ***sequential*** models: (1) **STAMP** [12]: It captures users' general interests from the long-term memory and short-term memory; (2) **DIN** [34]: It uses an attention mechanism to model user interest from historical behaviors w.r.t. a target item; (3) **GRU4Rec** [6]: It is the first work to apply RNN to session-based recommendation with a ranking based loss; (4) **SASRec** [8]: It is a unidirectional transformer-based sequential model, which uses self-attention to capture sequential preferences; (5) **DIEN** [33]: It enhances DIN by combining attention with GRU units to take interests evolution into consideration; (6) **FMLPRec** [36]: It is an all-MLP model with learnable filters which can adaptively attenuate the noise information in historical sequences.

For methods using search data, we include following ***search-aware*** models: (7) **NRHUB** [24]: It is a news recommendation model leveraging heterogeneous user behaviors; (8) **JSR** [28]: It is a general framework which optimizes a joint loss. We implement it following [17] to ensure a fair comparison with other sequential models; (9) **IV4REC** [17]: It is a model-agnostic framework exploiting search queries as instrumental variables to enhance the recommendation model. Following the original paper, we apply this framework over DIN; (10) **Query-SeqRec** [5]: It is a query-aware sequential model which incorporates queries into user behaviors using a transformer-based model. (11) **SRJGraph** [31]: It is a GNN-based model which exploits a heterogeneous graph to model the user-item and user-query-item interactions for S&R.

5.1.4 *Implementation Details.*

All hyper-parameters of baselines are searched following suggestions from the original papers. For all models, the maximum sequence length of recommendation (search) history is set to 150 (25) on the Kuaishou dataset and 15 (15) on the Amazon dataset. d_i, d_q and d are set as 48, 64 and 48 (32, 32, and 32) on the Kuaishou dataset (Amazon dataset). For the fair competition, we deploy the same setting of item embeddings on all models. For query embeddings, we also randomly initialize term embeddings for all search-aware models. The batch size is set as 256. The hyper-parameters α and β are set as 0.1 and 0.001, respectively. The margin m is set as 0.1. We use the Adam [9] with a learning rate of 0.001, and adopt early-stopped training to avoid over-fitting. More details can be found in the open source codes[5].

5.2 Overall Performance

Table 2 reports the recommendation results on the two datasets. We have the following observations:

• Search-aware models do not always bring performance gains. SRJGraph is the SOTA approach that mines both S&R behaviors. However, the SOTA sequential model FMLP-Rec can obtain compatible or even better performance than SRJGraph. Besides, Query-SeqRec shares a similar architecture as SASRec but achieves slightly poorer performance than SASRec. These phenomenons indicate

[3]The Amazon review dataset can be found at http://jmcauley.ucsd.edu/data/amazon/.
[4]The constructed search data is available at https://github.com/QingyaoAi/Amazon-Product-Search-Datasets.

[5]https://github.com/Ethan00Si/SESREC-SIGIR-2023

Table 3: Ablation studies by progressively adding proposed modules to the base model. MIE is short for the multi-interest extraction module.

Model	N@5	N@10	H@1	H@5	H@10	MRR
Base	0.3394	0.3770	0.2027	0.4618	0.6094	0.3294
$+\mathcal{L}_{\text{ali}}^{u,t}$	0.3464	0.3982	0.2106	0.4762	0.6308	0.3421
$+\mathcal{L}_{\text{con}}^{u,t}$	0.3507	0.4021	0.2139	0.4812	0.6406	0.3459
+MIE	**0.3541**	**0.4054**	**0.2173**	**0.4848**	**0.6436**	**0.3490**

that blindly incorporating S&R histories may be insufficient to capture users' diverse interests because users' interests in essence are entangled.

• Compared to baseline models, SESRec achieves the best performance on both datasets in most cases, significantly outperforming the SOTA sequential and search-aware methods FMLP-Rec and SRJGraph by a large margin (paired t-test at p-value < 0.01). The relative improvements over conventional sequential models reveal that leveraging search behaviors can boost recommendation models. Furthermore, the substantial performance gains over search-aware models validate the effectiveness of interest disentanglement in S&R behaviors.

• Comparing SESRec on two datasets, SESRec achieves fewer relative improvements on the Amazon dataset. Considering that search data of the Amazon dataset was automatically constructed, many items share the same queries, and the number of queries is sparse compared with the number of items, as shown in Table 1. Thus, the query-item alignment and interest contrast modules play a minor role in boosting recommendation performance on this dataset.

5.3 Detailed Empirical Analysis

In this section, we conducted more detailed experiments on the real-world Kuaishou dataset, providing in-depth analyses of how and why SESRec achieved state-of-the-art performance.

5.3.1 Ablation Study. SESRec consists of several key components, including alignment for queries and items, disentanglement for user S&R interests with self-supervised signals, and the multi-interest extraction module. To investigate how different components affect the performance of SESRec, we conducted ablation studies by progressively adding three components to the base model. We added these modules one by one because each module depends on previous modules. The base model solely processes S&R behaviors with transformer layers and the interest extraction module of aggregated interests. Table 3 shows the results on the Kuaishou dataset. Next, we give a detailed discussion about each component:

• $\mathcal{L}_{\text{ali}}^{u,t}$: denotes the loss function of query-item alignment, which guarantees that model captures the correlation between queries and items. We observed that adding $\mathcal{L}_{\text{ali}}^{u,t}$ leads to consistent performance gain. The results demonstrate that understanding the interactions between queries and items is beneficial to jointly model S&R behaviors.

• $\mathcal{L}_{\text{con}}^{u,t}$: refers to the loss function of interest contrast, which is designed to disentangle similar and dissimilar interests between S&R behaviors. The interest disentanglement leads to performance improvement, which indicates the necessity of disentanglement.

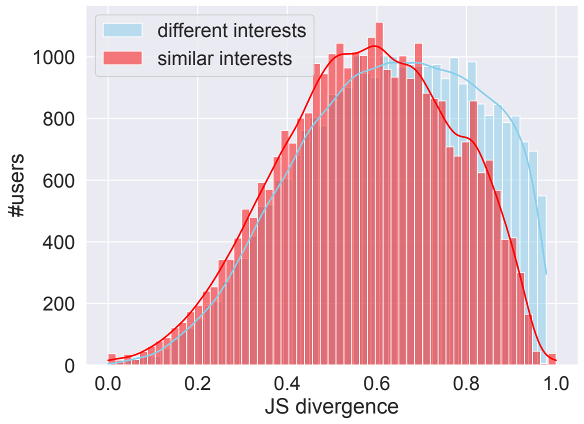


Figure 3: Visualization of similarity between similar and dissimilar interests in S&R behaviors for all users based on a histogram. We use the JS divergence of item category distributions to estimate similarities between S&R behaviors. Similar interests are more similar than dissimilar interests with smaller values of JS divergence.

We attribute the improvement to the fact that $\mathcal{L}_{\text{con}}^{u,t}$ helps the model capture more accurate representations of user interests.

• MIE: is short for the multi-interest extraction module, which is designed to extract interests from three perspectives, *i.e.,* aggregated, similar, and dissimilar interests. We found that MIE contributes to the final prediction, validating the importance of MIE. Though $\mathcal{L}_{\text{con}}^{u,t}$ has distinguished interests into similar and dissimilar ones, the model cannot explicitly leverage similar or dissimilar interests for prediction without MIE.

5.3.2 Analysis of Disentangled Interests. The core operation of interest disentanglement is the separation of similar and dissimilar interests. To illustrate whether the interests are disentangled, we visualize and interpret the similarity between constructed sets of similar and dissimilar interests in Figure 3. Behaviors $\mathcal{P}_s, \mathcal{P}_r$ containing similar interests and $\mathcal{N}_s, \mathcal{N}_r$ containing dissimilar interests are separated in Equation (11) and (12). For each user, we calculated the Jensen–Shannon (JS) divergence [11] between sets with similar interests, *i.e.,* $D_{JS}(\mathcal{P}_s \| \mathcal{P}_r)$, which is colored in red in Figure 3. The same calculation also was done for sets with dissimilar interests, *i.e.,* $D_{JS}(\mathcal{N}_s \| \mathcal{N}_r)$, which is colored in blue in Figure 3. As discussed in section 1, we can use the categories of items S_i^u (interacted in recommendation history) and S_c^u (clicked in search history) to estimate the similarities between two behaviors. The calculation of JS divergence is based on the distributions of item categories corresponding to $\mathcal{P}_s, \mathcal{P}_r$ and $\mathcal{N}_s, \mathcal{N}_r$ for each user. Figure 3 illustrates the results for all users. We observed that similar interests tend to have smaller values of JS divergence than dissimilar interests, where red data has more counts smaller than 0.6 compared with blue data. This phenomenon indicates $\mathcal{P}_s$ and $\mathcal{P}_r$ are more similar than $\mathcal{N}_s$ and $\mathcal{N}_r$, verifying the capability of SESRec to disentangle user interests.

5.3.3 Analysis of γ_s, γ_r in Interest Disentanglement. The separation of similar and dissimilar interests depends on the positive/negative

Table 4: The analysis of the positive/negative selection thresholds γ_r, γ_s in interest disentanglement, as defined in Equation (11) and (12).

γ_r, γ_s	N@5	N@10	H@1	H@5	H@10	MRR
1/16	0.3429	0.3927	0.2125	0.4681	0.6224	0.3429
1/8	0.3449	0.3940	0.2148	0.4691	0.6211	0.3415
Median	0.3510	0.4035	0.2135	0.4828	**0.6453**	0.3462
Mean	**0.3541**	**0.4054**	**0.2173**	**0.4848**	0.6436	**0.3490**

behaviors selection, as defined in Equation (11) and (12). We investigated the impacts of different positive/negative selection thresholds γ_r, γ_s. In practice, we utilize an adaptive way, *i.e.,* setting thresholds as mean values ($\gamma_r = \frac{1}{T_r}, \gamma_s = \frac{1}{T_r}$), to choose the positives and negatives. To show how γ_r, γ_s affect the performance of SESRec, we conducted experiments where γ_r, γ_s are set as constants, *e.g.,* $\frac{1}{8}$ and $\frac{1}{16}$, and we also investigated another adaptive way that sets γ_r, γ_s as the median values of given similarity scores. Table 4 presents the results with different settings of γ_r, γ_s. We can observe that constant settings ($\gamma_r, \gamma_s = \frac{1}{8}$ or $\frac{1}{16}$) yield inferior performances compared with the two adaptive strategies. We postulate that the constant settings can not handle behaviors with different lengths consistently because longer behaviors lead to smaller mean values of scores normalized by softmax. We also find that the adopted mean value strategy achieves better performance than the median value strategy in most cases. The median value strategy separates behavior sequences into two parts of the same length. However, the similar and dissimilar parts of users' behaviors do not satisfy this distribution in most cases. That is why the adopted mean value strategy achieves the best performance.

5.3.4 *Effect of Query-item Alignment.*

We conducted experiments to explore how the query-item alignment facilitates representation learning and whether SESRec understands the similarity of queries and items. Toward this end, we tested the relevance between queries and their clicked items. For search behaviors, we split queries and their clicked items into pairs, where each pair consists of a query and its corresponding item. And we obtained their embeddings learned by SESRec and a variation of SESRec, which removes the query-item alignment loss $\mathcal{L}_{\text{ali}}^{u,t}$. We calculated the cosine similarity of each query-item pair based on the embeddings and plotted the distribution of similarity scores in Figure 4. From the results, we can observe that embeddings learned by SESRec have smaller similarity scores than those learned without $\mathcal{L}_{\text{ali}}^{u,t}$. These results indicate that the query-item alignment module ensures that queries are closed to their corresponding items in correlation.

5.3.5 *Impact of Hyper-parameters.*

Since we design two additional tasks, hyper-parameters α and β are introduced to balance the objectives in the final loss function, as defined in Equation (23). To investigate the impacts of these hyper-parameters, we conducted experiments with varying α and β respectively. When varying one parameter, the other is set as a constant, where 1e-1 for α and 1e-3 for β. From the results in Figure 5, we found that the performance peaks when α is 1e-1 and β is 1e-3. With a further increase of hyper-parameters, the recommendation performances become worse. We attribute it to the fact that the recommendation prediction task becomes less important with larger α and β, which verifies the necessity of hyper-parameters to balance different tasks in the multi-task learning schema.

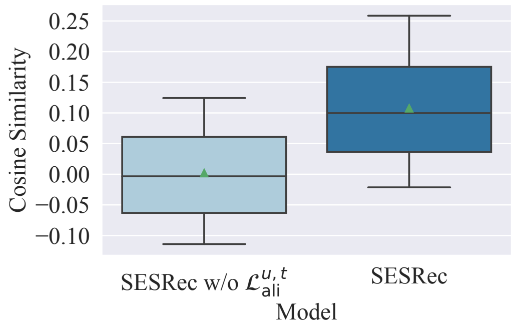
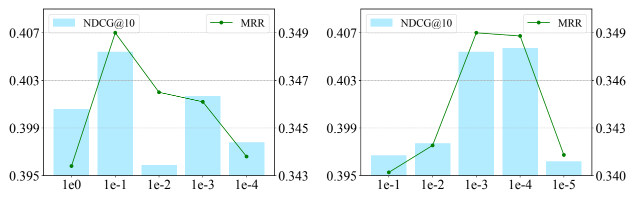


Figure 4: Distribution of cosine similarity between representations of queries and corresponding items based on box plots. Rectangles denote mean values. With query-item alignment loss $\mathcal{L}_{\text{ali}}^{u,t}$, embeddings of query-item pairs are more similar with higher cosine values.



(a) Performance of different α. **(b) Performance of different β.**

Figure 5: Effects of hyper-parameters α and β in terms of NDCG@10 and MRR.

6 CONCLUSION

In this paper, we propose to learn disentangled search representation for recommendation with a search-enhanced framework, namely SESRec. SESRec exploits the query-item interactions to help the recommendation model to learn better representations of queries and items. With the help of self-supervision, SESRec disentangles the similar and dissimilar representations between users' search and recommendation behaviors to capture users' interests from multiple aspects. Besides, SESRec provides an end-to-end multi-task learning framework for estimating the parameters. Extensive experiments on industrial and public datasets demonstrate that SESRec consistently outperforms state-of-the-art baselines. Moreover, we empirically validate that SESRec successfully learn the disentangled representations of user interests.

ACKNOWLEDGMENTS

This work was funded by the National Key R&D Program of China (2019YFE0198200), Beijing Outstanding Young Scientist Program NO. BJJWZYJH012019100020098, Major Innovation & Planning Interdisciplinary Platform for the "Double-First Class" Initiative, and Public Computing Cloud, Renmin University of China. The work was partially done at Beijing Key Laboratory of Big Data Management and Analysis Methods.

REFERENCES

[1] Qingyao Ai, Yongfeng Zhang, Keping Bi, Xu Chen, and W. Bruce Croft. 2017. Learning a Hierarchical Embedding Model for Personalized Product Search *(SIGIR '17)*. Association for Computing Machinery, New York, NY, USA, 645–654. https://doi.org/10.1145/3077136.3080813

[2] 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.

[3] Yangyang Guo, Zhiyong Cheng, Liqiang Nie, Yinglong Wang, Jun Ma, and Mohan Kankanhalli. 2019. Attentive Long Short-Term Preference Modeling for Personalized Product Search. *ACM Trans. Inf. Syst.* 37, 2, Article 19 (jan 2019), 27 pages. https://doi.org/10.1145/3295822

[4] 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* (Montréal, Québec, Canada) *(WWW '16)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE, 507–517. https://doi.org/10.1145/2872427.2883037

[5] 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. https://doi.org/10.1145/3511808.3557677

[6] 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.). http://arxiv.org/abs/1511.06939

[7] Yupeng Hou, Binbin Hu, Zhiqiang Zhang, and Wayne Xin Zhao. 2022. CORE: Simple and Effective Session-based Recommendation within Consistent Representation Space. In *SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, July 11 - 15, 2022*. ACM, 1796–1801. https://doi.org/10.1145/3477495.3531955

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

[9] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. http://arxiv.org/abs/1412.6980

[10] Junnan Li, Ramprasaath Selvaraju, Akhilesh Gotmare, Shafiq Joty, Caiming Xiong, and Steven Chu Hong Hoi. 2021. Align before fuse: Vision and language representation learning with momentum distillation. *Advances in neural information processing systems* 34 (2021), 9694–9705.

[11] Jianhua Lin. 1991. Divergence measures based on the Shannon entropy. *IEEE Transactions on Information theory* 37, 1 (1991), 145–151.

[12] Qiao Liu, Yifu Zeng, Refuoe Mokhosi, and Haibin Zhang. 2018. STAMP: Short-Term Attention/Memory Priority Model for Session-Based Recommendation. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery &amp; Data Mining* (London, United Kingdom) *(KDD '18)*. Association for Computing Machinery, New York, NY, USA, 1831–1839. https://doi.org/10.1145/3219819.3219950

[13] Jianxin Ma, Chang Zhou, Hongxia Yang, Peng Cui, Xin Wang, and Wenwu Zhu. 2020. Disentangled Self-Supervision in Sequential Recommenders. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery &amp; Data Mining* (Virtual Event, CA, USA) *(KDD '20)*. Association for Computing Machinery, New York, NY, USA, 483–491. https://doi.org/10.1145/3394486.3403091

[14] 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* (Santiago, Chile) *(SIGIR '15)*. Association for Computing Machinery, New York, NY, USA, 43–52. https://doi.org/10.1145/2766462.2767755

[15] Duy-Kien Nguyen and Takayuki Okatani. 2018. Improved Fusion of Visual and Language Representations by Dense Symmetric Co-Attention for Visual Question Answering. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*. Computer Vision Foundation / IEEE Computer Society, 6087–6096. https://doi.org/10.1109/CVPR.2018.00637

[16] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning Transferable Visual Models From Natural Language Supervision. In *ICML (Proceedings of Machine Learning Research)*, Vol. 139. PMLR, 8748–8763.

[17] 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. https://doi.org/10.1145/3485447.3511951

[18] Zihua Si, Zhongxiang Sun, Xiao Zhang, Jun Xu, Yang Song, Xiaoxue Zang, and Ji-Rong Wen. 2023. Enhancing Recommendation with Search Data in a Causal Learning Manner. *ACM Transactions on Information Systems* (Feb 2023). https://doi.org/10.1145/3582425

[19] 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. https://doi.org/10.1145/3357384.3357895

[20] Jiaxi Tang and Ke Wang. 2018. Personalized Top-N Sequential Recommendation via Convolutional Sequence Embedding. In *ACM International Conference on Web Search and Data Mining*.

[21] Aäron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation Learning with Contrastive Predictive Coding. *CoRR* abs/1807.03748 (2018). arXiv:1807.03748 http://arxiv.org/abs/1807.03748

[22] Christophe Van Gysel, Maarten de Rijke, and Evangelos Kanoulas. 2016. Learning Latent Vector Spaces for Product Search. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management* (Indianapolis, Indiana, USA) *(CIKM '16)*. Association for Computing Machinery, New York, NY, USA, 165–174. https://doi.org/10.1145/2983323.2983702

[23] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. *CoRR* abs/1706.03762 (2017). arXiv:1706.03762 http://arxiv.org/abs/1706.03762

[24] 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. https://doi.org/10.18653/v1/D19-1493

[25] Caiming Xiong, Victor Zhong, and Richard Socher. 2017. Dynamic Coattention Networks For Question Answering. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net. https://openreview.net/forum?id=rJeKjwvclx

[26] Chengfeng Xu, Pengpeng Zhao, Yanchi Liu, Victor S. Sheng, Jiajie Xu, Fuzhen Zhuang, Junhua Fang, and Xiaofang Zhou. 2019. Graph Contextualized Self-Attention Network for Session-Based Recommendation. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence* (Macao, China) *(IJCAI'19)*. AAAI Press, 3940–3946.

[27] 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]&amp; Knowledge Management* (Virtual Event, Queensland, Australia) *(CIKM '21)*. Association for Computing Machinery, New York, NY, USA, 2373–2382. https://doi.org/10.1145/3459637.3482489

[28] 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.

[29] 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. https://doi.org/10.1145/3336191.3371818

[30] Shengyu Zhang, Lingxiao Yang, Dong Yao, Yujie Lu, Fuli Feng, Zhou Zhao, Tat-seng Chua, and Fei Wu. 2022. Re4: Learning to Re-Contrast, Re-Attend, Re-Construct for Multi-Interest Recommendation. In *Proceedings of the ACM Web Conference 2022* (Virtual Event, Lyon, France) *(WWW '22)*. Association for Computing Machinery, New York, NY, USA, 2216–2226. https://doi.org/10.1145/3485447.3512094

[31] 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. https://doi.org/10.1145/3488560.3498414

[32] Yu Zheng, Chen Gao, Jianxin Chang, Yanan Niu, Yang Song, Depeng Jin, and Yong Li. 2022. Disentangling Long and Short-Term Interests for Recommendation. In *Proceedings of the ACM Web Conference 2022* (Virtual Event, Lyon, France) *(WWW '22)*. Association for Computing Machinery, New York, NY, USA, 2256–2267. https://doi.org/10.1145/3485447.3512098

[33] 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.

[34] 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 (London, United Kingdom) *(KDD '18)*. Association for Computing Machinery, New York, NY, USA, 1059–1068. https://doi.org/10.1145/3219819.3219823

[35] Kun Zhou, Hui Wang, Wayne Xin Zhao, Yutao Zhu, Sirui Wang, Fuzheng Zhang, Zhongyuan Wang, and Ji-Rong Wen. 2020. S3-Rec: Self-Supervised Learning for Sequential Recommendation with Mutual Information Maximization. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*. ACM, 1893–1902.

[36] 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. https://doi.org/10.1145/3485447.3512111