

Prompt-based Multi-interest Learning Method for Sequential Recommendation

Xue Dong, Xuemeng Song, *Senior Member, IEEE*, Tongliang Liu, *Senior Member, IEEE*, and Weili Guan

Abstract—Multi-interest learning method for sequential recommendation aims to predict the next item according to user multi-faceted interests given the user historical interactions. Existing methods mainly consist of a multi-interest extractor that embeds the multiple user interests based on the user interactions, and a multi-interest aggregator that aggregates the learned multi-interest embeddings to derive the final user embedding, used for predicting the user rating to an item. Despite their effectiveness, existing methods have two key limitations: 1) they directly feed the user interactions into the multi-interest extractor and aggregator, while ignoring their different learning objectives, and 2) they merely consider the centrality of the user interactions to embed multiple interests of the user, while overlooking their dispersion. To tackle these limitations, we propose a prompt-based multi-interest learning method (PoMRec), where specific prompts are inserted into user interactions, making them adaptive to the extractor and aggregator. Moreover, we utilize both the mean and variance embeddings of user interactions to embed the user multiple interests for the comprehensively user interest learning. We conduct extensive experiments on three public datasets, and the results verify that our proposed PoMRec outperforms the state-of-the-art multi-interest learning methods.

Index Terms—Sequential recommendation, multi-interest learning method, prompt tuning.

1 INTRODUCTION

RECOMMENDER systems have become increasingly prevalent in real-world applications, which target on recommending items for users based on their interests. The core of the recommender systems is to learn the user and item embeddings, and predict the user rating to the item with the distance between their embeddings. Traditional recommendation methods [1], [2], [3] treat the user historical interactions as a set, ignoring their sequential information. Intuitively, a user may purchase a phone case after buying a cellphone. Therefore, many researches [4], [5], [6], [7] have formalized the sequential recommendation task, where user interactions are treated as an ordered sequence to predict the next item that the user will be interested in. Currently, the mainstream sequential recommendation methods focus on devising various neural networks, such as the recurrent neural networks [4], [5] and Transformer [8], to encode the user interaction sequence into a single embedding to represent the user interests.

However, user interests are diverse and multi-faceted. For example, a girl may be simultaneously interested in jewelry, handbags, and make-ups. In such case, a single interest embedding could be hard to accurately capture all the user's diverse interests. Hence, several multi-interest learning methods [9], [10], [11], [12] for the sequential recommendation have been proposed to learn multiple

interest embeddings for each user. Generally, these methods involve two key modules. 1) The *multi-interest extractor* that aims to derive multiple interest embeddings for one user to capture his/her multi-faceted interests. And 2) the *multi-interest aggregator* that aggregates the learned multi-interest embeddings to one user embedding. Then the user rating to one item can be predicted by the dot product between the user and item embeddings. Despite their effectiveness, existing multi-interest learning methods for the sequential recommendation still have the following limitations:

- *Fail to adapt the user interactions to different learning objectives of the multi-interest extractor and aggregator.* Since both the user multi-interest embeddings and their aggregation weights can be referred to the user interactions, existing methods directly take the user interactions as the inputs to the multi-interest extractor and aggregator. In fact, the extractor focuses on the contents of user interactions to embed the user multiple interests, while the aggregator emphasizes analyzing the distribution of user interactions over multiple interests to derive the aggregation weights. Therefore, existing methods fail to make the inputted user interactions adaptive to different learning objectives of the two modules, whereby may derive the precise results.
- *Overlook the dispersion of user interactions during the user multi-interest learning.* Existing approaches mainly derive the user multi-interest embeddings by selecting a representative embedding of the user interaction embeddings, e.g., the weighted summation. his representative embedding only capture the centrality of user interactions. However, the user interactions can be dispersed and it is insufficient that existing methods learn the user multi-interest embeddings with only their centrality when the user interactions become more dispersed.

• X. Dong is with the School of Software, Tsinghua University, Beijing 100084, China. E-mail: dongxue.sdu@gmail.com.

• X. Song is with the School of Computer Science and Technology, Shandong University, Qingdao 266237, China. E-mail: sxmustc@gmail.com.

• T. Liu is with Sydney AI Centre, the University of Sydney, 6 Cleveland St, Darlington, NSW 2008, Australia. Email: tongliang.liu@sydney.edu.au.

• W. Guan is with the Faculty of Information Technology, Monash University (Clayton Campus), Australia. E-mail: honeyguan@gmail.com.

X. Song is the corresponding author.

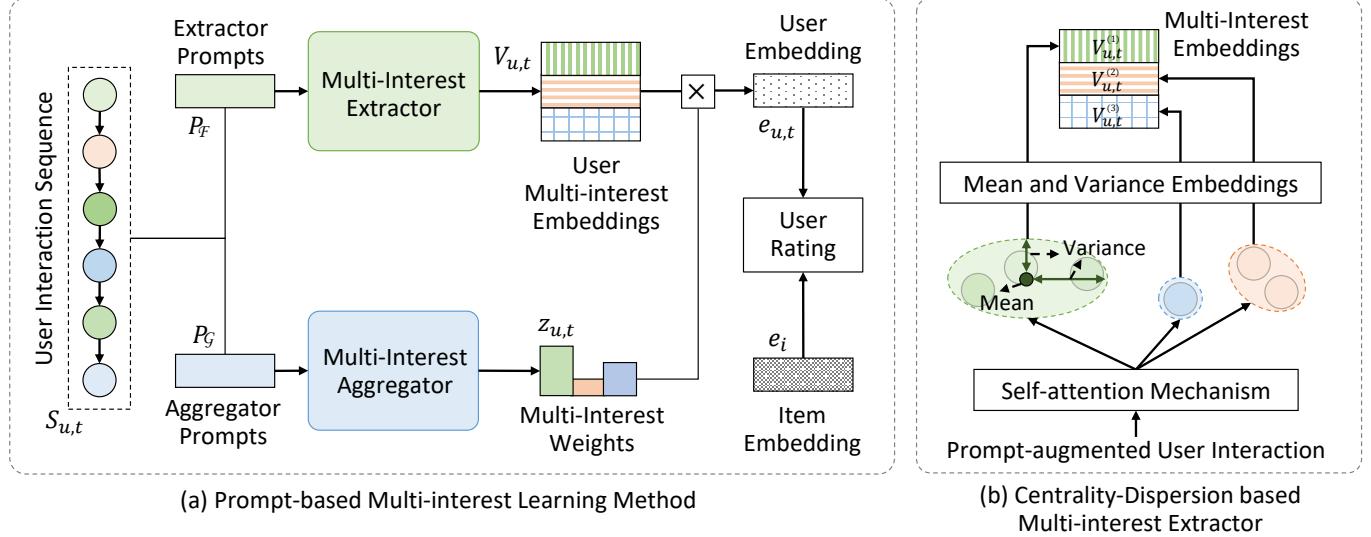


Fig. 1. Overview of the proposed prompt-based multi-interest learning method (PoMRec) for sequential recommendation (a). We insert certain prompt embeddings at the beginning of the user interaction sequence to make the model know whether it should focus on the contents of user interactions to derived the multi-interest embeddings or the preference over multiple interest to predict the aggregation weights. Besides, we propose a centrality-dispersion based multi-interest extractor (b) that derives the multi-interests embeddings based on both the centrality and dispersion of user interactions. Here we provide an example of the user that has three interests.

To address the aforementioned limitations, in this paper, we propose a **Prompt-based Multi-interest** learning method for the **sequential Recommendation**, termed as PoMRec, as shown in Figure 1. As same as existing multi-interest learning methods, PoMRec consists a multi-interest extractor that learns the user multi-interest embeddings, and a multi-interest aggregator that learns the weights to fuse the multi-interest embeddings. Differently, in order to adapt the inputted user interactions to different learning objectives of the extractor and aggregator, we take inspiration from the soft prompts introduced in the prompt tuning [13], [14] that can make the original inputs adaptive to different downstream tasks. As can be seen in Figure 1 (a), we first introduce multiple learnable prompt embeddings for the multi-interest extractor and aggregator, respectively. Then the original inputs, i.e., the user interactions, augmented by the prompt embeddings, are fed into the two modules. With the help of the prompts, the model will know whether it should focus on the contents of user interactions to derived the multi-interest embeddings or their distribution over multiple interests to derive the aggregation weights. Thereafter, the outputted user multi-interest embeddings are aggregated by the weights as the final user embedding to predict the user rating to a given item. Besides, to embed the user multiple interests in a more comprehensive manner, we propose a centrality-dispersion based multi-interest extractor, as shown in Figure 1 (b), which attempts to learn the multi-interest embeddings considering both the centrality and dispersion of the user interactions. We first adopt the self-attention mechanism to softly cluster the prompt-augmented user interactions into several groups. Then we calculate both the mean and variance embeddings of the interaction embeddings in each group, and incorporate them as the final user interest embedding. We conduct extensive experiments on three public datasets: ML-

1M, Beauty, and Movie & TV. The experimental results have demonstrated the effectiveness of the proposed PoMRec.

The main contributions can be summarized as follows:

- We propose a prompt-based multi-interest learning method for the sequential recommendation (PoMRec) that introduces the learnable prompt embeddings into the inputted user interactions to make them adaptive to the different learning objectives of the multi-interest extractor and aggregator.
- Different from existing methods that only consider the centrality of user interactions when embedding the user multi-interests, we propose a centrality-dispersion based multi-interest extractor that further takes the dispersion of user interactions as a supplement for seeking a better representation of the user multiple interests.
- Extensive experiments on the three public datasets have demonstrated the effectiveness of the proposed method. We have released our codes to facilitate other researchers in <https://github.com/hello-dx/PoMRec>.

2 RELATED WORK

In this section, we first introduce the single-interest and multi-interest learning methods in Subsection 2.1 and Subsection 2.2, respectively. Besides, we review the prompt tuning methods in Subsection 2.3.

2.1 Single-interest Learning Method

The sequential recommendation aims to predict the next item that the user might be interested in based on the user historical interaction sequence [4], [6], [15], [16]. Early researches [4], [6], [17] focus on encoding the user interaction sequence into a single embedding to represent the user single-interest. For example, Cheng et al. [17] adopted Markov chains to capture the correlation among items. With the development of deep neural networks,

several studies employ the sequential modeling techniques, e.g., the Gate Recurrent Unit [4] and Long Short-Term Memory [18], to encode the user interaction sequence into a single embedding to predict the next item. Recently, the attention mechanism [6], [19], [20] has shown promising potential to capture context-aware interests. However, user interests are diverse and multi-faceted. The aforementioned methods that utilize a single embeddings is hard to capture all the user’s diverse interests. Besides, existing methods only leverage one embedding to represent the centrality of user interactions, which might be insufficient to capture the user interests when the user interactions are dispersed.

2.2 Multi-interest Learning Method

As user interests are diverse, multi-interest learning has been proposed for the sequential recommendation, which aims to learn multiple embeddings for one user to capture the user multi-interests [9], [10], [21], [22], [23]. Existing methods generally involve a multi-interest extractor to learn the multiple interests of the user, and a multi-interest aggregator to fuse the multiple interests for generating the final recommendation. In particular, as for the extractor, MIND [9] uses the dynamic capsule routing method to group user interactions and obtain multiple user embeddings. Cen et al. [10] adopted the self-attention mechanism to generate multiple embeddings from the inputted user interactions. As for the aggregator, several approaches utilize the simple greedy inference strategy [10], [24] that utilizes the best matching interest embedding (among all interest embeddings) to rank the item. Other recent approaches [11], [21] predict a weight vector to aggregate the user multi-interest embeddings into the user embedding, which shows greater performance. For example, Wang et al. [11] utilized the Gate Recurrent Unit (GRU) to encode user interactions into one embedding and mapped it to the weight vector by the multi-layer perceptron.

Nevertheless, these methods directly feed the user interactions into the multi-interest extractor and aggregator, without making the inputs adaptive to the different learning objectives. In this paper, we introduce specific prompts into user interactions in order to make the downstream model known whether it should focus on the contents of user interactions or the user preference over multiple interests.

2.3 Prompt Tuning

The prompt learning is firstly proposed to overcome the gap between the pre-training and fine-tuning [14], [25]. It focuses on adding prompts to the downstream tasks of the pre-training models to improve the performance of the downstream tasks without the fine-tuning step. Early approaches [26], [27] mostly incorporate manually generated discrete prompts to guide the model. Later, since the manually generated prompts are both time-consuming and trivial, other researches [28], [29], [30] turn to automatically search discrete prompts for specific tasks. Nevertheless, these methods largely depend on the quality of the generated prompts. Some recent approaches have begun to utilize the continuous learnable embeddings as the prompts [13], [31], [32] achieving the state-of-the-art performance.

TABLE 1
Summary of the Main Notations.

Notation	Explanation
\mathcal{U}, \mathcal{I}	Sets of users and items, respectively.
S_u	User u ’s interaction sequence.
$S_{u,t}$	User u ’s interaction sequence truncated at time t .
K	Predefined number of user interests.
d	Embedding size.
$e_i \in \mathbb{R}^d$	Embedding of the item i .
$e_{u,t} \in \mathbb{R}^d$	Embedding of the user u at time t .
$H_{u,t} \in \mathbb{R}^{d \times M}$	Interaction embeddings $H_{u,t} = [e_{u,t-M}, \dots, e_{u,t}]$ of user u contains most recent M items in $S_{u,t}$.
N_p	Number of the prompt embeddings.
$P_F \in \mathbb{R}^{N_p \times d}$	Prompts for multi-interest extractor.
$P_G \in \mathbb{R}^{N_p \times d}$	Prompts for multi-interest aggregator.
$V_{u,t} \in \mathbb{R}^{d \times K}$	Multi-interest embeddings of the user u , where the column vector $V_u^{(k)}$ is the k -th interest embedding.
$z_{u,t} \in \mathbb{R}^K$	Interest weights of the user u at time t .
$y_{u,t}^i$	Predicted rating of the user u to item i at time t .
λ	Trade-off parameter between the centrality and dispersion of user interactions.

Inspired by them, in this paper, we introduce the prompt embeddings into the multi-interest interest learning to make the inputs adaptive to the downstream multi-interest extractor and aggregator. Our approach is the first attempt to add the prompt embeddings into the multi-interest learning for the sequential recommendation.

3 METHODOLOGY

We propose a prompt-based multi-interest learning method for sequential recommendation (PoMRec). In this section, we first brief the problem definition of the sequential recommendation in Subsection 3.1. We then detail the proposed PoMRec in Subsection 3.2, followed by the model complexity analysis in Subsection 3.3.

3.1 Problem Definition

To improve the readability, we declare the notations used in this paper. We use the squiggled letters (e.g., \mathcal{X}) to represent sets. The bold capital letters (e.g., \mathbf{X}) and bold lowercase letters (e.g., \mathbf{x}) represent matrices and vectors, respectively. Let the nonbold letters (e.g., x) denote scalars.

Suppose that there is a set of users \mathcal{U} , and a set of items \mathcal{I} . Each user $u \in \mathcal{U}$ is associated with a sequence of all his/her historical interactions sorted by their corresponding interacted timestamps, denoted as $S_u = [i_{u,1}, i_{u,2}, \dots, i_{u,N_u}]$, where $i_{u,t}$ is the interacted item at time step t and N_u is the length of the list. Different from traditional recommendation that represents the user with a given user ID, the sequential recommendation resort to a sequence of item IDs that the user historically interacted. In particular, as for a given user, the goal of the sequential recommendation takes the user interaction sequence $S_{u,t}$ truncated as the time step t as the input and predicts the next item $i_{u,t+1}$ that the user will be interested in be. The notations used in this paper are summarized in Table 1.

3.2 Prompt-based Multi-interest Learning Method for the Sequential Recommendation (PoMRec)

In this subsection, we first outline the proposed PoMRec method. Then we provide an implementation of the multi-interest extractor and aggregator in PoMRec, respectively, followed by the model optimization.

3.2.1 Overall Framework

Given the user interaction sequence $S_{u,t}$, PoMRec has a multi-interest extractor that derives the user multi-interest embeddings and a multi-interest aggregator that learns the weights to fuse the multi-interest embeddings. Different from previous methods, PoMRec inserts certain prompts into the inputted user interaction sequence $S_{u,t}$ before feeding it into the multi-interest extractor and aggregator, which makes it adaptive to different learning objectives of the two modules.

Inputs. Accordingly, the inputs of PoMRec consist of two parts: the original user interactions and newly introduced prompts for the multi-interest extractor and aggregator.

- To represent the original user interactions, following most recommendation methods [3], [10], [11], we represent each item $i \in \mathcal{I}$ with a d -dimensional learnable embedding $e_i \in \mathbb{R}^d$. Then the user historical interaction sequence $S_{u,t}$ can be transformed into an embedding sequence $\mathbf{H}_{u,t} = [e_{u,t-M+1}, \dots, e_{u,t-1}, e_{u,t}]$, which consists of the most recent interacted M items at the time step t . We term the embedding sequence $\mathbf{H}_{u,t}$ as the user interaction embeddings. Notably, if the sequence length is less than M , we repeatedly add a zero embedding to the beginning of the sequence until the length becomes M . In addition, we also add the trainable positional embeddings to each item embedding in $\mathbf{H}_{u,t}$ in order to make use of the sequential information.
- We then introduce the prompts for the multi-interest extractor and aggregator. Following the soft prompts in prompt tuning methods [13], [31], we introduce the to-be-learned prompt embeddings $\mathbf{P}_{\mathcal{F}} = [p_{\mathcal{F}}^1, \dots, p_{\mathcal{F}}^{N_p}]$ for the extractor, and $\mathbf{P}_{\mathcal{G}} = [p_{\mathcal{G}}^1, \dots, p_{\mathcal{G}}^{N_p}]$ for the aggregator, both of which consist of N_p randomly initialized d -dimensional embeddings. The prompt embeddings $\mathbf{P}_{\mathcal{F}}$ and $\mathbf{P}_{\mathcal{G}}$ are expected to capture the specific learning objectives of the extractor and aggregator, respectively. Notably, we use more than one prompt embedding to increase the adaptive ability of the inputs [33].

Ultimately, the prompt embeddings can be as the identifier to prompt the model to determine whether it should focus on the contents or the distribution over interests of user interactions. Formally, we insert the prompt embeddings at the beginning of the interaction embeddings $\mathbf{H}_{u,t}$ as the inputs for the extractor and aggregator as follows,

$$\begin{cases} \mathbf{H}_{u,t}^{\mathcal{F}} = [\mathbf{P}_{\mathcal{F}}, \mathbf{H}_{u,t}], \\ \mathbf{H}_{u,t}^{\mathcal{G}} = [\mathbf{P}_{\mathcal{G}}, \mathbf{H}_{u,t}], \end{cases} \quad (1)$$

where $\mathbf{H}_{u,t}^{\mathcal{F}} \in \mathbb{R}^{(N_p+M) \times d}$ and $\mathbf{H}_{u,t}^{\mathcal{G}} \in \mathbb{R}^{(N_p+M) \times d}$ are the prompt-augmented interaction embeddings inputted into the multi-interest extractor and aggregator, respectively.

Centrality-Dispersion based Multi-interest Extractor. Considering that the user's current interests can be inferred

by his/her recent interactions, this extractor \mathcal{F} will output the user multi-interest embeddings based on the prompt-augmented interaction embeddings $\mathbf{H}_{u,t}^{\mathcal{F}}$. Formally, suppose that each user has K interests. The user multi-interest embeddings $\mathbf{V}_{u,t}$ can be derived as follows,

$$\mathbf{V}_{u,t} = \mathcal{F}(\mathbf{H}_{u,t}^{\mathcal{F}} | \Theta_{\mathcal{F}}), \quad (2)$$

where $\mathbf{V}_{u,t} \in \mathbb{R}^{d \times K}$ refer to the total K interest embeddings of the user u at the time step t . $\Theta_{\mathcal{F}}$ is the set of parameters in the extractor \mathcal{F} . The detailed implementation of the extractor can refer to Subsection 3.2.2.

Attention-based Multi-interest Aggregator. The aggregator \mathcal{G} focuses on aggregating the learned multi-interest embeddings $\mathbf{V}_{u,t}$ to one embedding for predicting the final user rating. In particular, the aggregation weights of different user interests can be traced from the user interactions. For example, if the user have purchased the outdoor equipment, he/she is not likely to prefer the same items in the next time. Therefore, we feed the prompt-augmented interaction embeddings $\mathbf{H}_{u,t}^{\mathcal{G}}$ into the aggregator \mathcal{G} to learn the weights of multiple user interests. Then the final user embedding $e_{u,t} \in \mathbb{R}^d$ at the time step t can be aggregated by the learned weights as follows,

$$e_{u,t} = \mathcal{G}(\mathbf{H}_{u,t}^{\mathcal{G}}, \mathbf{V}_{u,t} | \Theta_{\mathcal{G}}), \quad (3)$$

where $\Theta_{\mathcal{G}}$ is the set of parameters in the aggregator \mathcal{G} . The detailed implementation can refer to Subsection 3.2.3.

Output. Ultimately, we use the commonly-used dot product between the user and item embeddings as the rating of the user u to a given item i as follows,

$$y_{u,t}^i = \mathbf{e}_{u,t}^T \mathbf{e}_i, \quad (4)$$

where $y_{u,t}^i$ is the rating of the user u to the item i at the time step t . Accordingly, based on the predicted user ratings, the candidate items can be ranked and top items are recommended to the user.

3.2.2 Centrality-Dispersion Based Multi-interest Extractor

The basic idea of the centrality-dispersion based multi-interest extractor is to first represent the centrality of each interest in the user interactions and then calculate the dispersion according to the centrality as the supplement. To be more specific, following studies [10], [11], we adopt the self-attention mechanism to learn the centrality representation matrix $\mathbf{M}_{u,t} \in \mathbb{R}^{d \times K}$ of the user multi-interests as follows,

$$\begin{cases} \mathbf{M}_{u,t} = (\mathbf{H}_{u,t}^{\mathcal{F}})^T \mathbf{A}_{u,t}^{\mathcal{F}}, \\ \mathbf{A}_{u,t}^{\mathcal{F}} = \text{softmax}(\mathbf{W}_{\mathcal{F}}^2 \tanh(\mathbf{W}_{\mathcal{F}}^1 \mathbf{H}_{u,t}^{\mathcal{F}}))^T, \end{cases} \quad (5)$$

where $\mathbf{A}_{u,t}^{\mathcal{F}} \in \mathbb{R}^{(N_p+M) \times K}$ is an attention matrix, which represents the probability of each user interaction belonging to each interest of the user u . Intuitively, the attention matrix is able to softly classify the user interactions into K groups. Accordingly, the k -th column of the centrality representation matrix, i.e., $\mathbf{M}_{u,t}^{(k)}$ is derived by the weighted summation of the interaction embeddings, which represents the centrality of the k -th interests of the user u . The function softmax is to enforce the summation of the elements in the k -th column vector of the attention matrix to equal to 1. $\mathbf{W}_{\mathcal{F}}^1 \in \mathbb{R}^{d' \times d}$ and $\mathbf{W}_{\mathcal{F}}^2 \in \mathbb{R}^{K \times d'}$ are the trainable

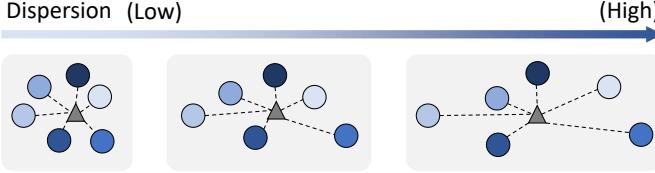


Fig. 2. Illustration of the user interactions with different dispersion. The blue points represent the user interacted items, and the grey triangles are their corresponding centrality representation. Intuitively, along with the dispersion of user interactions from low to high, the reliability of the centrality for representing the user interactions decreases.

parameters. \tanh is the activation function. It is worth noting that we can also utilize other approaches to learn the centrality representation matrix $M_{u,t}$.

Existing approaches directly utilize the centrality representation matrix $M_{u,t}$ as the user multi-interest embeddings. Nevertheless, as shown in Figure 2, along with the user interactions becoming increasingly dispersed, the reliability of the centrality representation to represent the user interactions tends to decrease. To tackle this issue, in this work, we first calculate a dispersion representation matrix based on the centrality representation matrix, and then derive the user multi-interest embeddings by their combination. Technically, we define the multi-interest embeddings $V_{u,t} \in \mathbb{R}^{d \times K}$ of the user u as follows,

$$\begin{cases} V_{u,t} = M_{u,t} + \lambda \Sigma_{u,t}, \\ \Sigma_{u,t}^{(k)} = \text{sqrt}\left(((H_{u,t}^F)^T A_{u,t}^{F(k)} - (M_{u,t}^{(k)})^2\right), k = 1, \dots, K, \end{cases} \quad (6)$$

where $\Sigma_{u,t} \in \mathbb{R}^{d \times K}$ is the dispersion representation matrix, calculated by the dispersion of user interactions from each centrality. λ is a hyper-parameter to adjust the trade-off between the centrality and dispersion of the user interactions in learning the user multi-interests. The superscript 2 of a matrix denotes the element-wise multiplication of the matrix. $M_{u,t}^{(k)}$ is the k -th column vector of the matrix $M_{u,t}$.

3.2.3 Attention-based Multi-interest Aggregator

The aggregator \mathcal{G} first predicts a weight vector that captures the weights of user multiple interests, and then aggregates the user multi-interest embeddings $V_{u,t}$ to the final user embedding. To fulfill this, we first aggregate these embeddings into a single embedding, and then map the embedding into a K -dimensional vector as the weight vector.

Technically, due to the impressive ability of self-attention mechanism in learning the attention weight of the embedding aggregation, we adopt it to aggregate the prompt-augmented interaction embeddings $H_{u,t}^F$ to derive a summarized embedding $v_{u,t} \in \mathbb{R}^d$ as follows,

$$\begin{cases} a_{u,t}^G = \text{softmax}\left(W_G^2 \tanh(W_G^1 H_{u,t}^G)\right)^T, \\ v_{u,t} = H_{u,t}^G a_{u,t}^G, \end{cases} \quad (7)$$

where $a_{u,t}^G \in \mathbb{R}^{N_p+M}$ contains the attention weights of the prompt-augmented interaction embeddings to derive the summarized embedding. $W_G^1 \in \mathbb{R}^{d' \times d}$ and $W_G^2 \in \mathbb{R}^{1 \times d'}$ are the trainable parameters.

We then adopt the multi-layer perceptron to map the summarized embedding to the weight vector as follows,

$$z_{u,t} = \underbrace{\cdots W_M^2 \tanh(W_M^1 v_{u,t} + b_M^1) + b_M^2}_{L} \cdots, \quad (8)$$

where $z_{u,t} \in \mathbb{R}^K$ denotes the weights of multiple interests of the user u at the time step t . L is the layer of the multi-layer perceptron, which is set to 2 empirically. $W_M^1 \in \mathbb{R}^{d' \times d}$, $b_M^1 \in \mathbb{R}^{d'}$, $W_M^2 \in \mathbb{R}^{d' \times K}$ and $b_M^2 \in \mathbb{R}^K$ are the trainable parameters.

Accordingly, we aggregate the learned user multi-interest embeddings $V_{u,t}$ based on the weight vector $z_{u,t}$ to derive the user embedding as follows,

$$e_{u,t} = V_{u,t} z_{u,t}, \quad (9)$$

where $e_{u,t} \in \mathbb{R}^d$ captures the user multi-interests at the time step t , which is used to predict the final user rating for the recommendation.

3.2.4 Model Optimization

The goal of the sequential recommendation is to predict the next item given the user interaction sequence at one time step. Accordingly, following Bayesian personalized ranking mechanism [34], we build the following training set,

$$\mathcal{D} = \{(S_{u,t}, i, j) \mid u \in \mathcal{U}, t = 1, \dots, N_u - 1, i = i_{u,t+1} \in S_u, j \in \mathcal{I} \setminus S_u\} \quad (10)$$

where $S_{u,t}$ is the user interaction sequence truncated at time step t , $i = i_{u,t+1}$ is the target next item in S_u , and $j \in \mathcal{I} \setminus S_u$ is a negative item that randomly sampled from items that the user has not interacted. The training triplet $(S_{u,t}, i, j)$ indicates that the user u prefers item i compared to the item j at the time step t . We then adopt the pair-wise objective function to ensure the rating of the user u to the positive item i is larger than the negative item j as follows,

$$\mathcal{L} = \min_{\Theta} - \sum_{\mathcal{D}} \log(\text{sigmoid}(y_{u,t}^i - y_{u,t}^j)), \quad (11)$$

where $y_{u,t}^i$ and $y_{u,t}^j$ are the ratings of the user u to the item i and j at the time step t , which can be derived in Eqn (4). sigmoid is the sigmoid activation function. Θ is the set of all parameters in the method, consisting of the item embeddings $\{e_i\}_{i \in \mathcal{I}}$, the prompt embeddings P_F and P_G , and parameters in the multi-interest extractor Θ_F and target-interest predictor Θ_G . The detailed training process of the proposed PoMRec is summarized in Algorithm 1.

3.3 Model Complexity Analysis

In PoMRec, the main time-consuming steps appear in the user multi-interest extraction and multi-interest weight prediction, i.e., step 6 and step 7 in Algorithm 1.

- Step 6 in Algorithm 1 learns the user multi-interest embeddings, and it has computational complexity $\mathcal{O}((dd' + d'K)(M + N_p))$ to derive the attention matrix A_{Fu} . Based on the attention matrix, we then calculate the mean and variance representations of the user multi-interests, both of which have the computational complexity $\mathcal{O}(dK(M + N_p))$. Accordingly, the user multi-interest embeddings can be calculated through Eqn. (6) with the computational complexity $\mathcal{O}(d(M + N_p))$.

Algorithm 1 Training Process of the Proposed PoMRec.

Input: The set of users \mathcal{U} and set of items \mathcal{I} . The historical interactions S_u of each $u \in \mathcal{U}$. The training set \mathcal{D} . The hyper-parameters K , N_p , and λ .

Output: The model parameters Θ .

- 1: Randomly initialize the embeddings $\{e_i\}_{i \in \mathcal{I}}$, \mathbf{P}_F , \mathbf{P}_G and parameters Θ_F , Θ_G .
- 2: Shuffle the training triplets $(S_{u,t}, i, j)$ in \mathcal{D} .
- 3: **while** not converged **do**
- 4: Draw a mini-batch of training triplet $(S_{u,t}, i, j)$.
- 5: Construct the prompt-augmented user interaction embeddings $\mathbf{H}_{u,t}^F$ and $\mathbf{H}_{u,t}^G$.
- 6: Calculate the user multi-interest embeddings: $\mathbf{V}_{u,t} = \mathcal{F}(\mathbf{H}_{u,t}^F | \Theta_F)$.
- 7: Calculate the final user embedding: $\mathbf{e}_{u,t} = \mathcal{G}(\mathbf{H}_{u,t}^G, \mathbf{V}_{u,t} | \Theta_G)$.
- 8: Calculate the user ratings to the item i and j : $y_{u,t}^i = \mathbf{e}_{u,t}^T \mathbf{e}_i, y_{u,t}^j = \mathbf{e}_{u,t}^T \mathbf{e}_j$.
- 9: Update the parameters of PoMRec: $\Theta \leftarrow \Theta - \eta \frac{\partial \mathcal{L}}{\partial \Theta}$
- 10: **end while**
- 11: **return** Model parameters Θ .

- Step 7 in Algorithm 1 learns the weight vector and it first derives the attention matrix \mathbf{A}_{Gu} , which has the computational complexity $\mathcal{O}(dd'(M + N_p))$. The weight vector then can be derived by the multi-layer perceptron taking the computational complexity $\mathcal{O}(dd' + d'K)$.

Therefore, the overall complexity for evaluating PoMRec is $\mathcal{O}(d'dK(M + N_p))$. Notably, compared with existing multi-interest learning methods, the proposed PoMRec only additionally introduces N_p prompt embeddings for multi-interest extraction and multi-interest weight prediction, respectively, which burdens little to the space complexity.

4 EXPERIMENTS

In this section, we first detail the experimental settings in Subsection 4.1, and then present the experiment results to answer the following research questions:

- RQ1: Does the proposed PoMRec outperform existing state-of-the-art methods?
- RQ2: How effective are the different proposals of PoMRec?
- RQ3: How effective is PoMRec deployed on existing multi-interest learning methods?
- RQ4: How do hyper-parameters affect model performance?
- RQ5: Do the learned multi-interest embeddings capture the user multi-interests?

4.1 Experimental Settings

Datasets. To evaluate our proposed PoMRec in the task of the top- N item recommendation, we conducted extensive experiments on the following three public datasets: **ML-1M** [35], **Bueaty** [36] and **Movies & TV** [36], which have various scales and sparsities. All these datasets contain the users' ratings on items, and each rating is annotated with its time step. For the fair comparison, we closely followed the data pre-processing of the study [11]. The statistics of the datasets are summarized in Table 2.

TABLE 2
Statistics of datasets.

Dataset	#User $ \mathcal{U} $	#Item $ \mathcal{I} $	#Interaction $\sum_u N_u$	Density $\frac{\sum_u N_u}{ \mathcal{U} \times \mathcal{I} }$
ML-1M	6,040	3,706	1,000,209	4.47%
Beauty	22,363	12,101	198,502	0.07%
Movies & TV	123,960	50,052	8,765,568	0.14%

Evaluation Protocols. We adopt the leave-one-out evaluation to evaluate model performance, which is widely-used in previous studies [6], [11]. In particular, for each user in the dataset, the most recent interaction is used for testing, the second most recent interaction is used for validation, and the remaining interactions are for training. In the validation and testing, the candidate items consist of one ground-truth item and 999 randomly sampled items that the user has not interacted with. This sampled metric is shown to be comparable to using the whole set of items as candidate items [11], [37], while largely saving the computational time. We adopted Recall@ N and NDCG@ N to evaluate the effectiveness of our proposed PoMRec in the top- N recommendation. By default, we set $N = 5, 10, 20$.

Implementation Configuration. The hyper-parameters in our proposed method consist of the embedding size d , the length M of the embedding sequence $\mathbf{H}_{u,t}$, the number of user interests K , the number of prompt embeddings N_p , and the trade-off parameter λ . In particular, we set the embedding size d to 64 and length M to 20 following existing studies [10], [11], [24]. We tuned these hyper-parameters from $\{1, 2, 3, 4, 5\}$, and results are discussed in Subsection 4.5. We optimized the proposed PoMRec with Adam optimizer [38]. We set the mini-batch sizes to 256 for all the datasets. We trained the network with 200 epochs with the early stop strategy, and selected the best model according to the performance of the validation set. Typically, 200 epochs are sufficient for PoMRec to converge.

4.2 Performance Comparison (RQ1)

To quantitatively examine the effectiveness of the proposed PoMRec, we compared it with the following multi-interest learning methods for the sequential recommendation:

- **GRU4Rec** [4] is the first sequential recommendation method that utilizes GRU to model the user interactions into one user interest embedding.
- **MIND** [9] uses the dynamic routing mechanism in capsule networks to group the user interacted items into multiple clusters and obtain multiple user interest embeddings for recommendation. To generate the final recommendation, it utilizes the maximal matching score between the items and the user interests as the final rating of the user to the item, which is referred to the greedy inference strategy.
- **ComiRec** [10] uses multi-head attention mechanisms to learn multiple embeddings for each user to capture their diverse interests. It also utilizes the greedy inference strategy to generate the final recommendation.
- **MINER** [12] introduces a novel poly attention scheme to learn the user multi-interest embeddings. It also lever-

TABLE 3

Experimental results of the performance comparison. The best and second best results are in bold and underlined, respectively. The parameter numbers of the methods are listed in the “Param.” column. The superscript * denotes that the results of the method are referred to the study [11].

Dataset	Method	Recall@N			NDCG@N			Param. (Million)
		N=5	N=10	N=20	N=5	N=10	N=20	
ML-1M	GRU4Rec*	0.2730	0.3964	0.5323	0.1875	0.2273	0.2616	0.3
	MIND*	0.1863	0.2881	0.4152	0.1229	0.1558	0.1877	0.2
	ComiRec*	0.2513	0.3659	0.4937	0.1708	0.2078	0.2400	0.2
	MINER	0.2758	0.3901	0.3901	0.1878	0.2246	0.2574	0.2
	TiMiRec*	<u>0.3091</u>	<u>0.4310</u>	<u>0.5625</u>	<u>0.2136</u>	<u>0.2529</u>	<u>0.2861</u>	0.5
	PoMRec	0.3151	0.4422	0.5752	0.2188	0.2598	0.2933	0.2
Relative Improvement		1.94%	2.60%	2.26%	2.43%	2.73%	2.52%	-
Beauty	GRU4Rec*	0.1072	0.1552	0.2107	0.0719	0.0873	0.1013	0.9
	MIND*	0.1193	0.1727	0.2492	0.0809	0.0981	0.1173	0.8
	ComiRec*	0.1257	0.1832	0.2543	0.0852	0.1038	0.1217	0.8
	MINER	0.1224	0.1740	0.2408	0.0841	0.1008	0.1176	0.8
	TiMiRec*	<u>0.1437</u>	<u>0.2006</u>	<u>0.2645</u>	<u>0.1006</u>	<u>0.1118</u>	<u>0.1350</u>	1.6
	PoMRec	0.1456	0.2031	0.2713	0.1010	0.1195	0.1367	0.8
Relative Improvement		1.32%	1.25%	2.57%	0.40%	6.89%	1.26%	-
Movies & TV	GRU4Rec	0.1928	0.2811	0.3871	0.1299	0.1584	0.1851	3.3
	MIND	0.2394	0.3290	0.4282	0.1655	0.1954	0.2205	3.2
	ComiRec	0.2411	0.3291	0.4275	0.1687	0.1970	0.2219	3.2
	MINER	0.2467	0.3398	0.4426	0.1702	0.2002	0.2262	3.2
	TiMiRec	<u>0.2506</u>	<u>0.3412</u>	<u>0.4386</u>	<u>0.1755</u>	<u>0.2048</u>	<u>0.2294</u>	6.4
	PoMRec	0.2747	0.3684	0.4714	0.1944	0.2246	0.2507	3.2
Relative Improvement		9.62%	7.97%	7.48%	10.77%	9.29%	8.67%	-

ages a disagreement regularization to improve the poly attention, which enlarges the distance among different interest embeddings during training. The learned multi-interest embeddings are fed into the mean aggregation to derive the user embedding.

- **TiMiRec** [11] applies the self-attention mechanism to the multi-interest extractor to learn the user’s multi-interests and adopts the GRU to learn the weight vector from the user interactions. This method is a two-stage method, which first pre-trains the multi-interest extractor with the greedy inference strategy and then fine-tunes the interest weight predictor to learn the user embedding for predicting the recommended items.

Table 3 shows the comparison results of all baselines and the proposed PoMRec, where the best results are in bold and the second-best results are underlined. Besides, we also listed the total parameters of each method in Table 3. From Table 3, we have the following observations:

- 1) Our proposed PoMRec achieves the best performance with respect to all metrics on both datasets. This demonstrates the effectiveness of our proposed method. The reasons behind this may be as follows: (i) by adding the specific prompts of learning objectives, the inputs of the user interactions are adaptive to the multi-interest extractor and aggregator, thus that the multi-interest embeddings and their weights can be better learned; and (ii) we utilize both centrality and dispersion of the user

interactions to learn the user multi-interest embeddings, which comprehensively models the user multi-interests.

- 2) The proposed PoMRec not only achieves the better performance but also needs significantly fewer parameters compared to the best baseline, i.e., TiMiRec. This is because that TiMiRec allocates two embeddings for each item, with one embedding used for the multi-interest extractor and another used for the multi-interest aggregator. In contrast, the proposed PoMRec only allocates one embedding for each item, and inserts specific prompt embeddings into the item embeddings in the user interaction sequence to make them adaptive to both the extractor and aggregator.
- 3) Multi-interest learning methods MINER, TiMiRec, and PoMRec perform better than the multi-interest learning methods MIND and ComiRec. This may be due to that although MIND and ComiRec learn the user multi-interests, they merely generate recommendations based on the best matching interest of the target item. This suggests that it is helpful to predict the next items that the user might be interested in by leveraging all the learned user interests. Moreover, MINER performs worse than TiMiRec and our proposed PoMRec. This may be because that MINER only utilizes the mean aggregation of the learned user multi-interest embeddings to retrieve items, while TiMiRec and PoMRec utilize networks to predict interest weights to aggregate the learned user multi-interest embeddings.

TABLE 4

Experimental results of the ablation study on PoMRec. “+Prompt” denotes that we use the prompt embeddings to augment the inputted user interactions. “+Disp.” denotes that we utilize both the centrality and dispersion of user interaction to learn the multi-interest embeddings.

Dataset	Variant	Recall@N		NDCG@N	
		N=5	N=10	N=5	N=10
ML-1M	Base	0.3068	0.4248	0.2101	0.2484
	+Prompt	0.3129	0.4333	0.2148	0.2536
	+Disp.	0.3108	0.4320	0.2151	0.2544
	PoMRec	0.3151	0.4422	0.2188	0.2598
Beauty	Base	0.1193	0.1726	0.0823	0.0996
	+Prompt	0.1299	0.1838	0.0888	0.1061
	+Disp.	0.1293	0.1811	0.0889	0.1057
	PoMRec	0.1456	0.2031	0.1010	0.1195
Movies & TV	Base	0.2429	0.3295	0.1704	0.1983
	+Prompt	0.2498	0.3406	0.1738	0.2031
	+Disp.	0.2591	0.3536	0.1801	0.2106
	PoMRec	0.2747	0.3684	0.1944	0.2246

- 4) Note that introducing the prompt embeddings could involve additional parameters, but can be negligible compared to the whole model parameters. Specifically, the model parameters largely depends on the number of items. There are 3,706, 12,101, and 50,052 items in ML-1M, Beauty and Movies & TV datasets, respectively. However, there are at most 10 prompt embeddings introduced, i.e., there are two tasks in our method and we assign at most 5 embeddings for each task. Therefore, the new involved parameters only account for of 0.26%, 0.06%, and 0.02% of original model parameters in ML-1M, Beauty and Movies & TV datasets, respectively.

4.3 Ablation Study (RQ2)

In this subsection, we evaluated the proposed centrality-dispersion based multi-interest extractor and the prompt-based multi-interest learning method. In particular, we designed the following variants of PoMRec:

- **Base.** We removed both the CD and BP in the proposed PoMRec. Specifically, we directly fed the user interaction embeddings $\mathbf{H}_{u,t}$ without the prompts embeddings to both the multi-interest extractor and aggregator. Besides, in the multi-interest extractor, we only utilized the centrality representation as the user multi-interest embeddings, i.e., $\mathbf{V}_{u,t} = \mathbf{M}_{u,t}$, without the dispersion representation $\Sigma_{u,t}$.
- **+Prompt.** This variant adds prompt-based multi-interest learning method into the Base variant. Specifically, we fed the prompt-augmented user interaction embeddings $\mathbf{H}_{u,t}^F$ and $\mathbf{H}_{u,t}^G$ into the multi-interest extractor and aggregator, respectively.
- **+Disp.** Compared to the Base variant, this variant adopts the proposed centrality-dispersion based multi-interest extractor to derive the user multi-interest embeddings, i.e., $\mathbf{V}_{u,t} = \mathbf{M}_{u,t} + \lambda \Sigma_{u,t}$.

The experimental results of the ablation study are displayed in Table 4. Without losing generality, we reported

TABLE 5
Deployments of the PoMRec in existing multi-interest learning methods.

Dataset	Variant	Recall@N		NDCG@N	
		N=5	N=10	N=5	N=10
ML-1M	MINER	0.2758	0.3901	0.1878	0.2246
	PoM-MINER	0.2975	0.4147	0.2066	0.2441
	Impro.	7.87%	6.31%	10.01%	8.68%
	TiMiRec	0.3091	0.4310	0.2136	0.2529
Movie & TV	PoM-TiMiRec	0.3222	0.4344	0.2228	0.2580
	Impro.	4.24%	0.79%	4.31%	2.02%
	MINER	0.2467	0.3398	0.1702	0.2048
	PoM-MINER	0.2652	0.3596	0.1870	0.2176
& TV	Impro.	7.50%	5.83%	9.87%	6.25%
	TiMiRec	0.2506	0.3412	0.1755	0.2048
	PoM-TiMiRec	0.2735	0.3674	0.1933	0.2236
	Impro.	9.14%	7.68%	10.14%	9.18%

the Recall@5, Recall@10, NDCG@5, and NDCG@10 of the methods. As can be seen, firstly, the variant +Prompt outperforms the variant Base, when incorporating the prompt embeddings. This demonstrates that it is helpful to add specific prompts for the user interaction embeddings to guide their learning objectives in the multi-interest extractor and aggregator. Secondly, the variant +Disp. outperforms the variant Base, which proves that it is beneficial to learn the user multi-interest embedding from both the centrality and dispersion of the user interactions.

4.4 Deployment on Other Methods (RQ3)

The two proposals can also be deployed to other multi-interest learning methods that involve the multi-interest extractor and aggregator. In this part, we deployed our method to two state-of-the-art multi-interest learning method, i.e., MINER [12] and TiMiRec [11]. In particular, we directly adopted the model architectures of the multi-interest extractor and aggregator in MINER and TiMiRec. Differently, we added the learnable prompt embeddings into the inputs of the two modules and learned the user multi-interest embeddings with both the mean and variance embeddings of user interests, as same as our proposed PoMRec. The deployments of PoMRec on MINER and TiMiRec are termed as PoM-MINER and PoM-TiMiRec, respectively. Without losing generality, we reported the Recall@5 and NDCG@5 of the methods in ML-1M and Movie & TV dataset in Table 5.

As can be seen, the deployments PoM-MINER and PoM-TiMiRec outperform their corresponding methods MINER and TiMiRec in both datasets, which demonstrates the effectiveness and applicability of the proposed method. Notably, as aforementioned, the original TiMiRec assigns two embeddings for each item, i.e., one for the multi-interest extractor and the other one for the multi-interest aggregator. Nevertheless, our method aims to adapt the same inputs to different modules. Therefore, in PoM-TiMiRec, we only assigned a single embedding for each item. Intuitively, this adaptation could hurt the performance of PoM-TiMiRec. However, equipped the our proposed method,

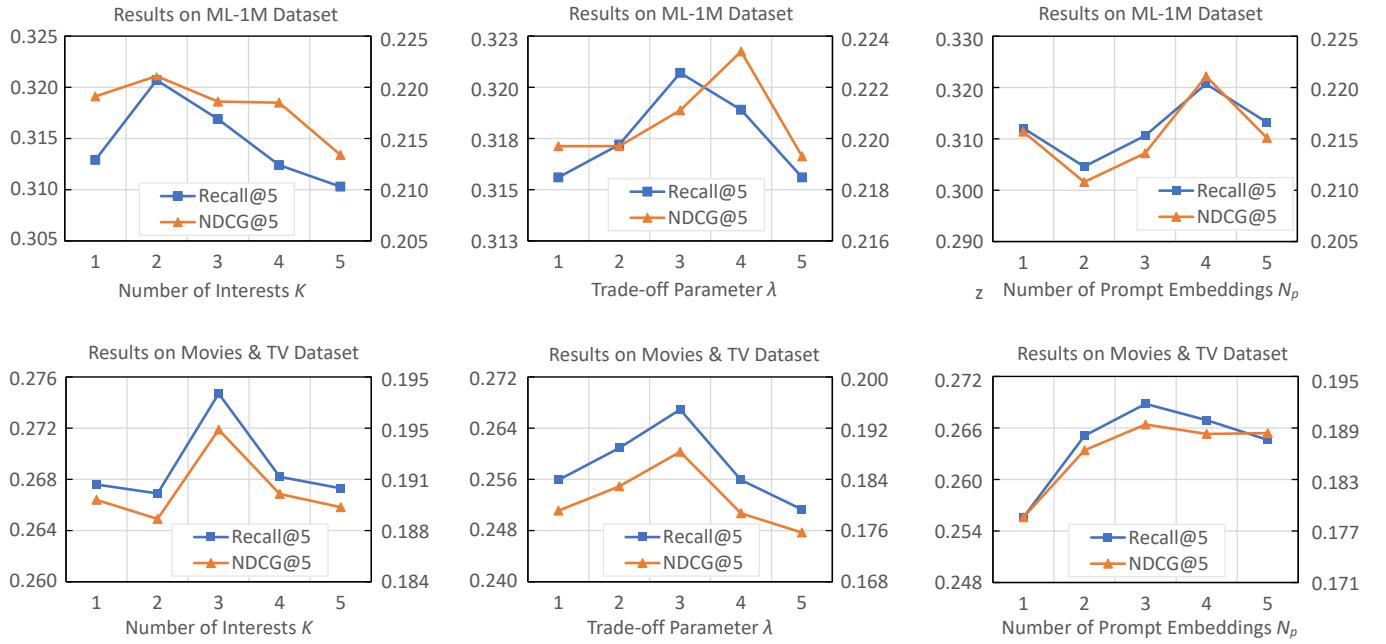


Fig. 3. Performance of the proposed PoMRec with respect to the different hyper-parameters in ML-1M and Movie & TV datasets. The left vertical axis refers to the Recall@5, while the right vertical axis refers to the NDCG@5.

PoM-TiMiRec still works better than TiMiRec, which further proves our effectiveness.

4.5 Hyper-parameter Discussion (RQ4)

In this subsection, we evaluated the following three key hyper-parameters: the number of user interests K defined in Eqn. (2), the trade-off parameter λ between the centrality and dispersion of user interactions during learning the user multi-interests defined in Eqn. (6), and the number of the prompt embeddings N_p defined in Eqn. (1).

4.5.1 The Number of User Interests K

A larger number of user interests K indicates that more interests the user has. In this experiment, we fixed the other hyper-parameters to their default values and tested K from 1 to 5 with a stride of 1. Without losing generality, we reported the Recall@5 and NDCG@5 on both datasets in Figure 3. The results show that, for both datasets, as the number of user interests K increases from 1 to 5, the model performance first increases until it achieves the best performance with the most suitable K , and then decreases. This suggests that the more complex multi-interest embeddings may not lead to better performance. In fact, the overly complex multi-interest embeddings can complicate the training and may degrade the recommendation performance. Furthermore, we observed that the most suitable K for ML-1M dataset is 2, while that for Movies & TV dataset is 3. This may be because that the interests of users in ML-1M are relatively more concentrated, while those in the other dataset are relatively more diverse.

4.5.2 The Trade-off Parameter λ

The trade-off parameter λ balances the importance of the centrality and dispersion of user interactions during learning the learning of user multi-interests. In this experiment,

we fixed the other hyper-parameters to their default values and tested λ from $\{1, 2, 3, 4, 5\}$. The Recall@5 and NDCG@5 results are shown in Figure 3. As can be seen, the model performance first increases until achieving the best performance and then decreases. This demonstrates that involving the dispersion of the user interactions appropriately does benefit the user multi-interest learning. The most suitable λ for ML-1M dataset is 4, while that for Movies & TV dataset is 3. This may be because that the user interactions in the ML-1M dataset are more dispersed than those in Movies & TV dataset, indicating that the dispersion of the user interactions plays a more important role in ML-1M.

4.5.3 The Number of Prompt Embeddings N_p

The hyper-parameter N_p controls the number of the prompt embeddings added to the inputs of user interactions. A larger N_p increases the adaptive ability of the inputs to different learning objectives in the multi-interest extractor and aggregator. In this experiment, we fixed the other hyper-parameters to their default values and tested N_p from $\{1, 2, 3, 4, 5\}$. The Recall@5 and NDCG@5 results are shown in Figure 3. From Figure 3, we can see that the performance first raises along with the number of the prompt embeddings N_p increasing in the two datasets. This demonstrates that it is helpful to insert appropriate prompts to make the inputs adaptive to the different learning objectives. However, the model performance decreases when N_p becomes overly large. This may be because that the multi-interest extractor and aggregator have certain connections despite having different learning objectives for the inputs. For example, the learned user multi-interest embeddings of extractor could help learn their weights in the aggregator. Therefore, excessive prompt embeddings can significantly increase the difference between the extractor and aggregator, while lessening their useful connections.

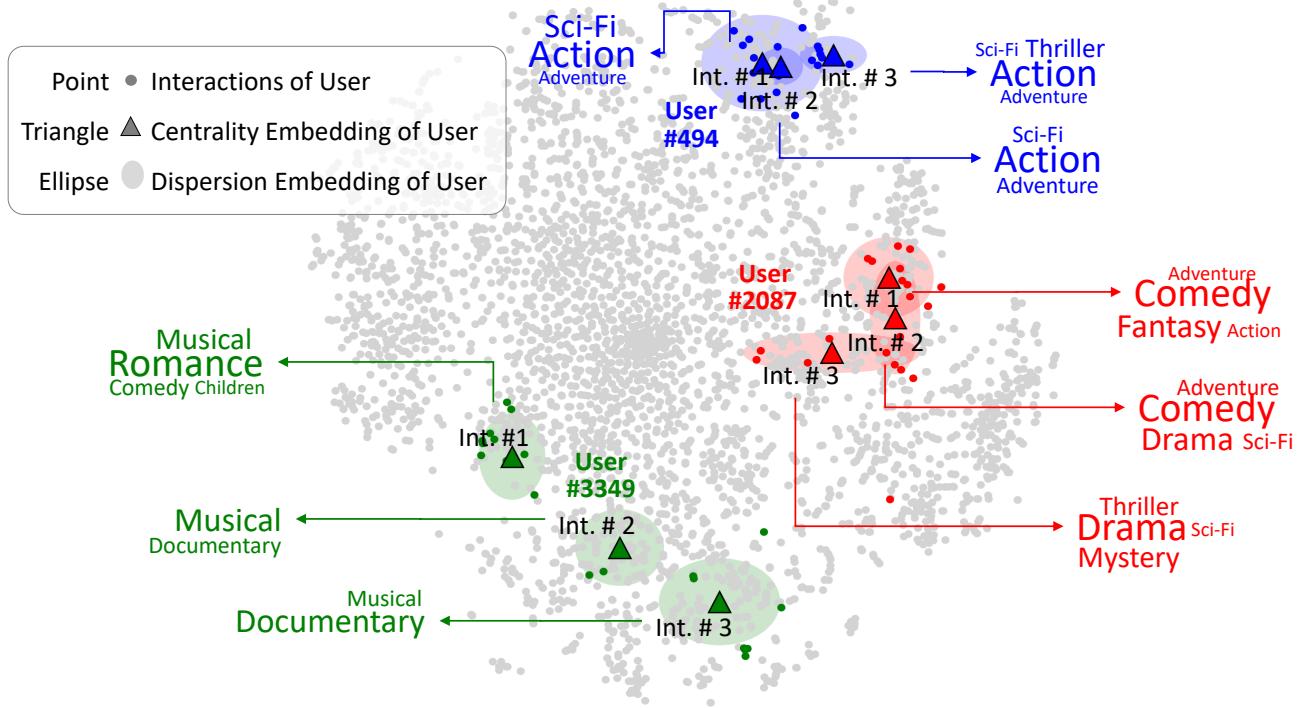


Fig. 4. Visualization of the learned user multi-interests in ML-1M dataset with the tool of t-SNE. The grey points represent the embeddings of all items in the dataset. The interactions of three users are highlighted with different colors. We visualize the learned user multi-interests with triangles (representing the centrality) and ellipses (representing the dispersion). To derive a deep understanding of the user multi-interests, we annotate the word cloud of user each interest generated with the item genres.

4.6 Visualization of The User Multi-interests (RQ5)

To intuitively show the necessity of the centrality and dispersion of the user interactions to capture the user multi-interests, in this subsection, we visualized three examples of the learned user multi-interest embeddings. In particular, we visualized the learned item embeddings of the ML-1M dataset with the tool of t-SNE [39] in Figure 4. Specifically, we highlighted the historical interactions of three random users, i.e., User #494, User #2087, and User #3349 in blue, red, and green, respectively. To depict the learned user multi-interest embeddings, for the k -th interest of the user u , we utilized the t-SNE tool to map the learned mean vector $\mu_{u,t}^{(k)} \in \mathbb{R}^d$ and variance vector $\Sigma_{u,t}^{(k)} \in \mathbb{R}^d$ into a two-dimensional vector, respectively. Then the triangle is drawn with the mapped mean vector, and based on that the ellipse is drawn with the mapped variance vector as the radius. Additionally, to obtain a deeper insight of the learned user multi-interests, we annotated the genres of the items belonging to each user interest by the word clouds, which help us understand the semantics of the three users' multi-interests. The larger the font size, the more frequently the genre occurs in the interest. From Figure 4, we have the following observations:

- The user interests are indeed diverse and multi-faceted, making it helpful to learn multiple interest embeddings for each user to capture their multi-interests. As can be seen, the interactions of green User #3349 can be clearly divided into three groups. Checking the word clouds of item genres in the groups, we found that each group of items has clearly common genres, while different groups have different genres. Specifically, the

most frequent genres of the Interest #1, #2, and #3 of the green User #3349 are Romance, Musical, and Documentary, respectively.

- Although we have learned a fixed number of interests for each user, the proposed PoMRec can deal with cases where a user has fewer interests than the fixed number. For example, as can be seen in Figure 4, the interactions of the blue User #494 are mostly gathered in one area. In such case, the learned mean embeddings, i.e., blue triangles, of the user multi-interests are close to each other, and the variance embeddings, i.e., blue ellipses, are almost overlapping. In addition, checking the word clouds of genres in interests of the blue User #494, we found that these three interests consist of mostly movies with genres of Action. Accordingly, we can infer that this user essentially has only one prominent interest in movies, i.e., the action movies. In this case, it is reasonable that the learned user multi-interests of PoMRec are close to each other.
- The dispersion of the user interactions can complement the centrality, which helps better describe the user multi-interests. For example, as for the red User #2087 in Figure 4, the interactions are divided into three groups. Interestingly, the interactions belonging to Interest #3 are dispersed in the horizontal axis but concentrated along the vertical axis. In such case, the mean embedding of the user interactions, i.e., the third red triangle, cannot represent the two interactions in the left. The learned variance embedding, i.e., the third red ellipse, captures the dispersion of the user interactions in Interest #3 and covers the left interactions of the user.

5 CONCLUSION AND FUTURE WORK

In this paper, we propose a prompt-based multi-interest learning method (PoMRec) for the sequential recommendation. In particular, we insert specific prompts into the user interactions to make them adaptive to the different learning objectives of the multi-interest extractor and aggregator. Moreover, we learn user multi-interest embeddings with not only the centrality of the user interactions but also their dispersion, which could comprehensively capture the user interests. Extensive experiments on three public datasets have demonstrated the effectiveness of the proposed PoMRec. In particular, we have found that although the multi-interest extractor and aggregator have their own learning objectives, they still share certain connections. Therefore, excessive specific prompts can introduce much differences between the two modules, while reducing their connections and hence negatively impacting the model performance. Nevertheless, the current PoMRec only utilizes the user interaction data, while overlooking the valuable information in item multimodal features. In the future, we plan to devise a multimodal multi-interest learning method to enhance the recommendation.

REFERENCES

- [1] L. Wu, C. Quan, C. Li, Q. Wang, B. Zheng, and X. Luo, "A context-aware user-item representation learning for item recommendation," *ACM Trans. Inf. Syst.*, vol. 37, no. 2, pp. 22:1–22:29, 2019.
- [2] F. Xue, X. He, X. Wang, J. Xu, K. Liu, and R. Hong, "Deep item-based collaborative filtering for top-n recommendation," *ACM Trans. Inf. Syst.*, vol. 37, no. 3, pp. 33:1–33:25, 2019.
- [3] X. He, K. Deng, X. Wang, Y. Li, Y. Zhang, and M. Wang, "Lightgcn: Simplifying and powering graph convolution network for recommendation," in *SIGIR conference on research and development in Information Retrieval*. ACM, 2020, pp. 639–648.
- [4] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk, "Session-based recommendations with recurrent neural networks," in *International Conference on Learning Representations*, 2016.
- [5] Q. Liu, S. Wu, D. Wang, Z. Li, and L. Wang, "Context-aware sequential recommendation," in *International Conference on Data Mining*. IEEE, 2016, pp. 1053–1058.
- [6] W. Kang and J. J. McAuley, "Self-attentive sequential recommendation," in *International Conference on Data Mining*. IEEE, 2018, pp. 197–206.
- [7] C. Wang, W. Ma, M. Zhang, C. Chen, Y. Liu, and S. Ma, "Toward dynamic user intention: Temporal evolutionary effects of item relations in sequential recommendation," *ACM Trans. Inf. Syst.*, vol. 39, no. 2, pp. 16:1–16:33, 2021.
- [8] K. Zhao, X. Zhao, Z. Zhang, and M. Li, "Mae4rec: Storage-saving transformer for sequential recommendations," in *International Conference on Information & Knowledge Management*. ACM, 2022, pp. 2681–2690.
- [9] C. Li, Z. Liu, M. Wu, Y. Xu, H. Zhao, P. Huang, G. Kang, Q. Chen, W. Li, and D. L. Lee, "Multi-interest network with dynamic routing for recommendation at tmall," in *International Conference on Information and Knowledge Management*. ACM, 2019, pp. 2615–2623.
- [10] Y. Cen, J. Zhang, X. Zou, C. Zhou, H. Yang, and J. Tang, "Controllable multi-interest framework for recommendation," in *Conference on Knowledge Discovery and Data Mining*. ACM, 2020, pp. 2942–2951.
- [11] C. Wang, Z. Wang, Y. Liu, Y. Ge, W. Ma, M. Zhang, Y. Liu, J. Feng, C. Deng, and S. Ma, "Target interest distillation for multi-interest recommendation," in *International Conference on Information & Knowledge Management*. ACM, 2022, pp. 2007–2016.
- [12] J. Li, J. Zhu, Q. Bi, G. Cai, L. Shang, Z. Dong, X. Jiang, and Q. Liu, "MINER: multi-interest matching network for news recommendation," in *Findings of the Association for Computational Linguistics*. ACL, 2022, pp. 343–352.
- [13] B. Lester, R. Al-Rfou, and N. Constant, "The power of scale for parameter-efficient prompt tuning," in *Conference on Empirical Methods in Natural Language Processing*. ACL, 2021, pp. 3045–3059.
- [14] P. Liu, W. Yuan, J. Fu, Z. Jiang, H. Hayashi, and G. Neubig, "Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing," *ACM Comput. Surv.*, vol. 55, no. 9, pp. 195:1–195:35, 2023.
- [15] F. Sun, J. Liu, J. Wu, C. Pei, X. Lin, W. Ou, and P. Jiang, "Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer," in *International Conference on Information and Knowledge Management*. ACM, 2019, pp. 1441–1450.
- [16] Z. Liu, Z. Fan, Y. Wang, and P. S. Yu, "Augmenting sequential recommendation with pseudo-prior items via reversely pre-training transformer," in *International Conference on Research and Development in Information Retrieval*. ACM, 2021, pp. 1608–1612.
- [17] C. Cheng, H. Yang, M. R. Lyu, and I. King, "Where you like to go next: Successive point-of-interest recommendation," in *International Joint Conference on Artificial Intelligence*. IJCAI, 2013, pp. 2605–2611.
- [18] Y. Zhou, C. Huang, Q. Hu, J. Zhu, and Y. Tang, "Personalized learning full-path recommendation model based on LSTM neural networks," *Inf. Sci.*, vol. 444, pp. 135–152, 2018.
- [19] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Conference on Neural Information Processing Systems*, 2017, pp. 5998–6008.
- [20] Y. Sha and M. D. Wang, "Interpretable predictions of clinical outcomes with an attention-based recurrent neural network," in *International Conference on Bioinformatics, Computational Biology, and Health Informatics*. ACM, 2017, pp. 233–240.
- [21] J. Ma, Z. Zhao, X. Yi, J. Chen, L. Hong, and E. H. Chi, "Modeling task relationships in multi-task learning with multi-gate mixture-of-experts," in *International Conference on Knowledge Discovery & Data Mining*. ACM, 2018, pp. 1930–1939.
- [22] Q. Pi, W. Bian, G. Zhou, X. Zhu, and K. Gai, "Practice on long sequential user behavior modeling for click-through rate prediction," in *International Conference on Knowledge Discovery & Data Mining*. ACM, 2019, pp. 2671–2679.
- [23] X. Li, A. Sun, M. Zhao, J. Yu, K. Zhu, D. Jin, M. Yu, and R. Yu, "Multi-intention oriented contrastive learning for sequential recommendation," in *International Conference on Web Search and Data Mining*. ACM, 2023, pp. 411–419.
- [24] G. Chen, X. Zhang, Y. Zhao, C. Xue, and J. Xiang, "Exploring periodicity and interactivity in multi-interest framework for sequential recommendation," in *International Joint Conference on Artificial Intelligence*. IJCAI, 2021, pp. 1426–1433.
- [25] Y. Gu, X. Han, Z. Liu, and M. Huang, "PPT: pre-trained prompt tuning for few-shot learning," in *Proceedings of the Association for Computational Linguistics*. ACL, 2022, pp. 8410–8423.
- [26] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," in *Neural Information Processing Systems*, 2020.
- [27] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *J. Mach. Learn. Res.*, vol. 21, pp. 140:1–140:67, 2020.
- [28] T. Gao, A. Fisch, and D. Chen, "Making pre-trained language models better few-shot learners," in *Proceedings of the Association for Computational Linguistics*. ACL, 2021, pp. 3816–3830.
- [29] Z. Jiang, F. F. Xu, J. Araki, and G. Neubig, "How can we know what language models know," *Trans. Assoc. Comput. Linguistics*, vol. 8, pp. 423–438, 2020.
- [30] T. Shin, Y. Razeghi, R. L. L. IV, E. Wallace, and S. Singh, "Autoprompt: Eliciting knowledge from language models with automatically generated prompts," in *Conference on Empirical Methods in Natural Language Processing*. ACL, 2020, pp. 4222–4235.
- [31] Z. Liang, H. Hu, C. Xu, J. Miao, Y. He, Y. Chen, X. Geng, F. Liang, and D. Jiang, "Learning neural templates for recommender dialogue system," in *Conference on Empirical Methods in Natural Language Processing*. ACL, 2021, pp. 7821–7833.

- [32] X. Guo, B. Li, and H. Yu, "Improving the sample efficiency of prompt tuning with domain adaptation," in *Association for Computational Linguistics: EMNLP*. ACL, 2022, pp. 3523–3537.
- [33] X. Wang, K. Zhou, J. Wen, and W. X. Zhao, "Towards unified conversational recommender systems via knowledge-enhanced prompt learning," in *Conference on Knowledge Discovery and Data Mining*. ACM, 2022, pp. 1929–1937.
- [34] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "BPR: bayesian personalized ranking from implicit feedback," in *Conference on Uncertainty in Artificial Intelligence*. AUAI Press, 2009, pp. 452–461.
- [35] F. M. Harper and J. A. Konstan, "The movielens datasets: History and context," *ACM Trans. Interact. Intell. Syst.*, vol. 5, no. 4, pp. 19:1–19:19, 2016.
- [36] J. Ni, J. Li, and J. J. McAuley, "Justifying recommendations using distantly-labeled reviews and fine-grained aspects," in *Conference on Empirical Methods in Natural Language Processing and International Joint Conference on Natural Language Processing*. Association for Computational Linguistics, 2019, pp. 188–197.
- [37] D. Li, R. Jin, J. Gao, and Z. Liu, "On sampling top-k recommendation evaluation," in *SIGKDD Conference on Knowledge Discovery and Data Mining*. ACM, 2020, pp. 2114–2124.
- [38] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations*. OpenReview.net, 2015.
- [39] L. van der Maaten, "Accelerating t-sne using tree-based algorithms," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 3221–3245, 2014.



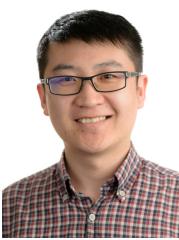
Weili Guan received the master degree from National University of Singapore. After that, she joined Hewlett Packard Enterprise in Singapore as a Software Engineer and worked there for around five years. She is currently a PhD student with the Faculty of Information Technology, Monash University (Clayton Campus), Australia. Her research interests are multimedia computing and information retrieval. She has published many papers at the first-tier conferences and journals, like ACM MM, SIGIR, and IEEE TIP.



Xue Dong received the Ph.D. degree from the School of Software, Shandong University, China in 2023. She is currently a research fellow with the Tsinghua University, China. Her research interests contain multimedia computing, multi-modal recommendation and retrieval. She has published several papers in the top venues, such as ACM SIGIR, MM, TOIS, IEEE TNNLS.



Xuemeng Song received the B.E. degree from the University of Science and Technology of China, in 2012, and the Ph.D. degree from the School of Computing, National University of Singapore, in 2016. She is currently an Associate Professor with Shandong University, China. She has published several papers in the top venues, such as ACM SIGIR, MM, and TOIS. Her research interests include information retrieval and social network analysis. She has served as a reviewer for many top conferences and journals.



Tongliang Liu is the Director of Sydney AI Centre at the University of Sydney. He is also heading the Trustworthy Machine Learning Laboratory. He is broadly interested in the fields of trustworthy machine learning and its interdisciplinary applications, with a particular emphasis on learning with noisy labels, adversarial learning, transfer learning, unsupervised learning, and statistical deep learning theory. He has authored and co-authored more than 100 research articles including ICML, NeurIPS, ICLR, CVPR, ICCV, ECCV, AAAI, IJCAI, KDD, IEEE T-PAMI, TNNLS, and TIP. He is/was a (senior-) meta reviewer for many conferences, such as ICML, NeurIPS, ICLR, UAI, AAAI, IJCAI, and KDD. He is a recipient of Discovery Early Career Researcher Award (DECRA) from Australian Research Council (ARC) and was named in the Early Achievers Leaderboard of Engineering and Computer Science by The Australian in 2020.