Knowledge Prompt-tuning for Sequential Recommendation

Jianyang Zhai
Sun Yat-sen University
Guangzhou, China
PengCheng Laboratory
Shenzhen, China
zhaijy01@pcl.ac.cn

Xiawu Zheng*
PengCheng Laboratory
Shenzhen, China
zhengxw01@pcl.ac.cn

Chang-Dong Wang*
Sun Yat-sen University
Guangdong Provincial Key
Laboratory of Intellectual Property
and Big Data
Guangzhou, China
changdongwang@hotmail.com

Hui Li
School of Informatics, Xiamen
University
Xiamen, China
hui@xmu.edu.cn

Yonghong Tian
Peking University
Beijing, China
PengCheng Laboratory
Shenzhen, China
yhtian@pku.edu.cn

ABSTRACT

Pre-trained language models (PLMs) have demonstrated strong performance in sequential recommendation (SR), which are utilized to extract general knowledge. However, existing methods still lack domain knowledge and struggle to capture users' fine-grained preferences. Meanwhile, many traditional SR methods improve this issue by integrating side information while suffering from information loss. To summarize, we believe that a good recommendation system should utilize both general and domain knowledge simultaneously. Therefore, we introduce an external knowledge base and propose Knowledge Prompt-tuning for Sequential Recommendation (**KP4SR**). Specifically, we construct a set of relationship templates and transform a structured knowledge graph (KG) into knowledge prompts to solve the problem of the semantic gap. However, knowledge prompts disrupt the original data structure and introduce a significant amount of noise. We further construct a knowledge tree and propose a knowledge tree mask, which restores the data structure in a mask matrix form, thus mitigating the noise problem. We evaluate KP4SR on three real-world datasets, and experimental results show that our approach outperforms state-of-the-art methods on multiple evaluation metrics. Specifically, compared with PLM-based methods, our method improves NDCG@5 and HR@5 by 40.65% and 36.42% on the books dataset, 11.17% and 11.47% on the music dataset, and 22.17% and 19.14% on the movies dataset, respectively. Our code is publicly available at the link: https://github.com/zhaijianyang/KP4SR.

*Corresponding authors.

CCS CONCEPTS

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

KEYWORDS

Sequential Recommendation, Pre-trained Language Model, Prompt Learning, Knowledge Graph

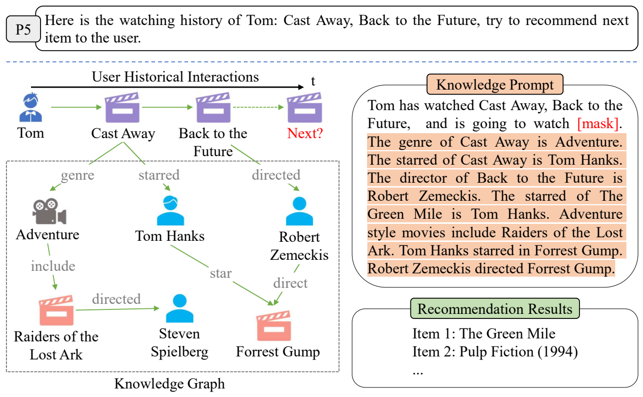
ACM Reference Format:
Jianyang Zhai, Xiawu Zheng, Chang-Dong Wang, Hui Li, and Yonghong Tian. 2023. Knowledge Prompt-tuning for Sequential Recommendation. In *Proceedings of the 31st ACM International Conference on Multimedia (MM '23), October 29–November 3, 2023, Ottawa, ON, Canada.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3581783.3612252

1 INTRODUCTION

Recommendation systems aim to suggest desirable items among an extensive item collection to target users, thereby alleviating the problem of information overload on the internet. Traditional recommendation algorithms, such as collaborative filtering [15], only mine the static correlation between users and items, ignoring the dynamic changes in user preferences implied in the historical interaction sequence. Therefore, the sequential recommendation that mines the dynamic changes in user preferences has become an important research direction in recommendation systems.

Some early studies modeled the historical behavior sequences of users using Markov chains [29, 31], but they often struggle to handle complex sequential patterns. With the development of deep learning, deep neural networks have achieved great success in sequential recommendation [13, 33]. Among them, models based on self-attention mechanisms, which adaptively assign weights to user interaction sequences, have become competitive mainstream solutions [5, 19, 32]. However, most of these methods only model the IDs of users and items, considering only the user's sequential preferences, and cannot capture the user's fine-grained preferences.

Many studies have introduced side information into SR [16, 44, 47] to address these issues. Some methods extract item attribute information [44, 47] and fuse them at different stages. For example, FDSA [44] uses different self-attention blocks to encode items and side information and only fuses their representations in the final



Here is the watching history of Tom: Cast Away, Back to the Future, try to recommend next item to the user.

Tom has watched Cast Away, Back to the Future, and is going to watch [mask]. The genre of Cast Away is Adventure. The starred of Cast Away is Tom Hanks. The director of Back to the Future is Robert Zemeckis. The starred of The Green Mile is Tom Hanks. Adventure style movies include Raiders of the Lost Ark. Tom Hanks starred in Forrest Gump. Robert Zemeckis directed Forrest Gump.

Item 1: The Green Mile
Item 2: Pulp Fiction (1994)
...

Figure 1: Comparison between P5 [9] (up) and our KP4SR (down), which introduces domain knowledge. We introduce an external knowledge graph and transform it into knowledge prompts to bridge the semantic gap between textual and structured data, which enriches the semantic features of items and improves the accuracy and explainability of recommendation results.

stage. S^3Rec [47] integrates side information through pre-training. However, inefficient feature fusion methods may result in the loss of useful information. Therefore, these methods focus on finding efficient solutions for fusing item embeddings and side information embeddings. There are also some methods that introduce external KG to assist the recommendation process [16, 17] and learn entity and relation embeddings through knowledge graph embedding (KGE) [2, 23] techniques. However, the primary optimization objective of KGE is the completion or prediction of edges in the KG rather than recommendation tasks. Overall, although side information can benefit recommendations, how to effectively incorporate side information into the recommendation process is still a challenging and unresolved problem.

Recently, PLMs have achieved great success in natural language processing (NLP) [3, 43]. By learning general knowledge from large corpora through self-supervised tasks, many researchers have utilized PLMs to solve recommendation tasks [9, 24, 35]. For example, PEPLER [21] used GPT-2 [27] to generate more natural recommendation explanations by treating users and items as personalized prompts. P5 [9] transformed the recommendation task into an NLP task and unified multiple recommendation tasks, such as SR, in one framework using the T5 [28] model. Although these methods have achieved good results, PLM-based RS methods still face difficulties in capturing complex user preferences because PLMs lack domain knowledge regarding users and items. Therefore, introducing domain knowledge is necessary when using PLMs to solve recommendation tasks. A straightforward and simple approach is to describe domain knowledge using natural language text and then use the powerful reasoning ability of PLMs to improve recommendation performance, as shown in Figure 1. However, there are two challenges with this approach: 1) How to convert structured knowledge graphs into text sequences. 2) Converting into text sequences may destroy the original data structure and how to deal with the noise caused by irrelevant entities and relationships.

Inspired by prompt learning in NLP [3, 41], we propose Knowledge Prompt-tuning Sequential Recommendation (KP4SR) to overcome the above challenges. Specifically, we improve P5 [9] by designing a masked personalized prompt (MPP) template set to convert the SR task into a pre-training task, which not only accelerates the convergence speed but also improves the model performance. Then, we design a set of relation templates to convert triplets into triplet prompts, which are combined to form knowledge prompts (KP). Finally, we propose prompt denoising (PD), constructing a knowledge tree and a knowledge tree mask to eliminate the mutual influence of irrelevant triplets. Extensive experimental results show that our method outperforms state-of-the-art methods on multiple metrics in three real-world datasets.

- We propose KP4SR, which, to the best of our knowledge, is the first work that transforms knowledge graphs into knowledge prompts to improve SR performance.
- We construct KP, which addresses the problems of semantic difference between structured knowledge data contained in the KG and the sequential text data used by PLMs and allows for easy utilization of high-order information from the KG.
- We propose PD, which mitigates knowledge noise by restoring the KG data structure in the form of a mask matrix.
- We conduct extensive experiments on three datasets, and the results demonstrate the effectiveness of our method. In addition, ablation experiments show that transforming the SR task into an NLP task still follows the general pattern of NLP, which indicates the great research prospects and research value of PLMs in improving the performance of recommendation systems.

2 RELATED WORK

2.1 Sequential Recommendation

Early studies utilized Markov Chains [29, 31] to model users' historical interaction sequences, but they often struggle with complex sequential patterns. Later, deep neural networks have shown strong performance in SR. GRU4Rec [13] leverages recurrent neural networks to encode user interaction sequences into hidden states to improve recommendation performance. Caser [33] uses horizontal and vertical convolutional filters to learn multi-level patterns and user preferences. Attention-based approaches allocate different weights to determine the relevance between users' historical interactions and target items, capturing users' dynamic preferences, such as SASRec [19], BERT4Rec [32], and Transformers4Rec [5]. CLSR [46] and SLi-Rec [42] improve recommendation performance by modeling users' long-term and short-term preferences. CL4SRec [39] introduces contrastive learning, which learns better sequence representations through self-supervised signals at the user behavior sequence level. However, they only model the IDs of users and items and cannot capture users' fine-grained preferences.

2.2 Side Information for SR

Many studies have used side information to improve SR, which mainly includes item attribute information and external knowledge base. For example, FDSA [44] encodes items and side information using different self-attention blocks, and integrates their

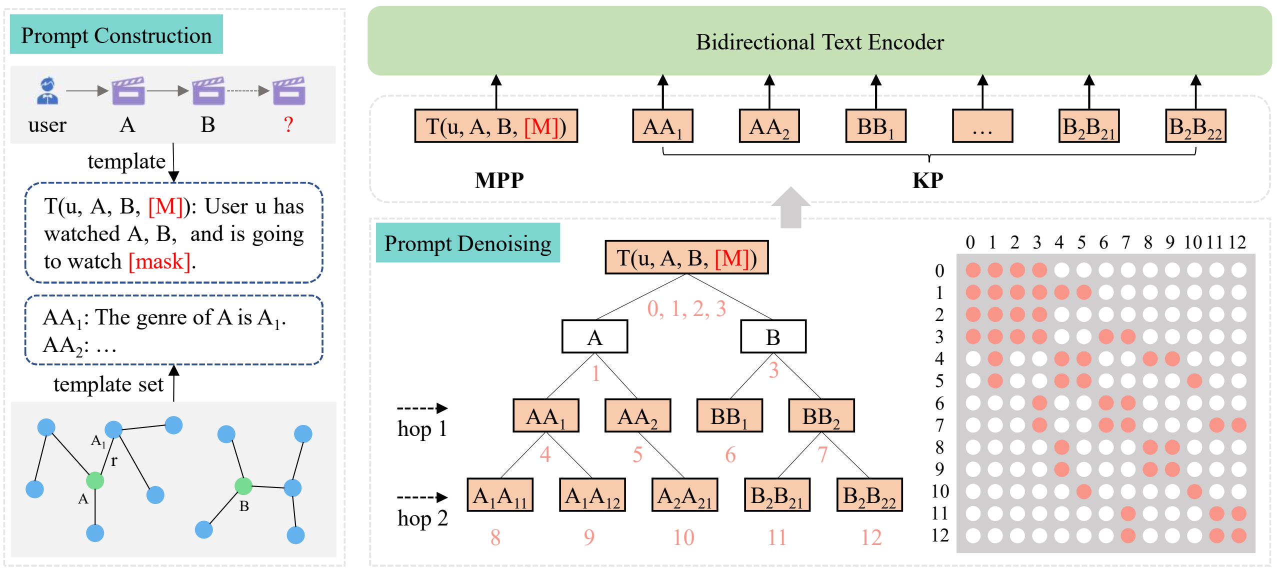


Figure 2: The overall framework of KP4SR. Firstly, we use a masked personalized prompt (MPP) template to transform the user-item interaction sequence into MPP (top-left). Then, we use relationship templates to transform the knowledge graph into a knowledge prompt (KP) (bottom-left). Finally, we construct a knowledge tree and propose a knowledge tree mask to alleviate the problem of knowledge noise (right).

representations in the final stage. S^{3}Rec [47] designs four auxiliary self-supervised objectives, utilizing the principle of maximum mutual information (MIM) to learn the correlation between attributes, items, sub-sequences, and sequences. DIF-SR [40] believes that early integration limits the expressive power of the attention matrix and the flexibility of gradients. It transfers the side information from the input to the attention layer and then decouples the attention calculation between various side information and item representations. KSR [16] integrates the RNN-based network and Key-Value Memory Network (KV-MN) together, and uses KG to capture users' fine-grained preferences. DHIMN [38] applies a message-passing layer based on Dynamic Heterogeneous Information Networks (DHIN) to capture advanced semantic knowledge in the KG, but ignores the heterogeneous information of item relationships in the KG. Compared to them, our method converts KG into text sequences and utilizes the powerful ability of PLMs to improve SR.

2.3 PLMs for Recommendation

PLMs have achieved tremendous success in natural language processing (NLP) [3, 6, 43], and many researchers have started to use PLMs to solve recommendation tasks [24]. PEPLER [21] employs GPT-2 [27] to generate personalized explanations for recommendations by using users and items as personalized prompts. METER [8] further incorporates visual information to improve the quality of recommendation explanations. In addition, SpeedyFeed [37] and NRMS [36] utilize PLMs to enhance news recommendations. P5 [9] transforms recommendation tasks into NLP tasks and unifies multiple recommendation tasks in one framework using T5 [28]. Some works propose knowledge-enhanced dialogue recommendation systems and use PLMs to generate smoother conversations [34, 35]. However, they used Graph Convolutional Networks (GCN) to model KG information, which faces the semantic gap that exists in natural language.

2.4 Prompt Learning

Prompt learning involves designing prompts for specific tasks to reframe downstream tasks as pre-training tasks, thus addressing the gap between pre-training tasks and downstream targets [10, 24]. Early research relied on manually crafted discrete prompts to guide pre-training language models [3, 28]. Recently, many works have focused on automatically generating discrete prompts for specific tasks [7, 18]. However, these methods still rely on generative models or complex rules to control prompt quality. In contrast, some works propose using learnable continuous prompts that can be directly optimized [20, 22]. Some researchers have incorporated prompt learning into recommendation systems, designing specific personalized prompts for different tasks, such as PEPLER [21] and M6-Rec [4]. Our KP4SR is an improvement on P5[9], and their personalized prompts do not convert the recommendation task into a pre-training task, which is data inefficient.

3 PROBLEM DEFINITION

We first introduce the symbols used in this paper. A typical recommendation scenario usually consists of a user set $\mathcal{U}$ and an item set $\mathcal{V}$. By sorting the interactions between users and items by timestamps, we can obtain the interaction sequence S_u of user u, which can be represented as $S_u = \{v_1^u, v_2^u, ..., v_{|u|}^u\}$, where $|u|$ denotes the length of the sequence, $u \in \mathcal{U}$ and $v \in \mathcal{V}$. Our goal is to predict the next item $v_{|u|+1}^u$ that the user is likely to interact with. To describe our method more clearly, we omit the subscripts and superscripts and simply define the user's interaction sequence as $\{A, B, C, D, E, \ldots\}$.

KG is a structured knowledge base that contains a set of triples, which can be defined as $\mathcal{KG} = \{(h, r, t) \mid h, t \in \mathcal{E}, r \in \mathcal{R}\}$, where $\mathcal{E}$ is the set of entities and $\mathcal{R}$ is the set of relations. A triple (h, r, t) represents a connection between the head entity h and the tail entity t through the relation r. We assume that each item in the recommendation system has a corresponding entity connection, i.e., $\mathcal{V} \in \mathcal{E}$. Therefore, for each item v, we can obtain an n-hop knowledge subgraph G_v^n centered on v.

Our goal is to incorporate domain knowledge from KG into PLMs to mine users' complex preferences. For instance, given a basic input sample: *"Tom has watched Cast Away, Back to the Future, and is going to watch [mask].",* we need to input the relevant KG information, such as *(Cast Away, film.genre, Adventure).* Therefore, the first challenge we face is how to input structured KG into PLMs while bridging the semantic gap between sequence text and structured knowledge. The second challenge is how to alleviate knowledge noise, as not all knowledge is helpful, and irrelevant and noisy knowledge can affect model performance [25]. In Section 4, we will elaborate on our method.

4 METHODOLOGY

4.1 Overview

We propose KP4SR, which transforms a structured KG into knowledge prompts to improve SR. It mainly consists of two modules: prompt construction module (Section 4.2) and prompt denoising module (Section 4.3). The overall framework is shown in Figure 2.

The prompt construction module consists of the masked personalized prompt (MPP) and knowledge prompt (KP). Firstly, we construct a set of MPP templates by converting the user-item interaction sequence into MPP. This allows us to transform the recommendation task into a natural language cloze task. Then, we construct a set of relationship templates to convert triples into triple prompts and further combine these triple prompts to form KP. This enables us to transform structured KG into text sequences. For the prompt denoising module, we first construct a knowledge tree based on the MPP and triple prompts, and then design the knowledge tree mask to alleviate the noise problem caused by irrelevant triple prompts. We will provide more details on the method in the following sections.

4.2 Prompts Construction

Prompt learning has achieved great success in NLP, and many researchers have applied it to recommendation systems [21, 35]. In this section, we construct a collection of MPP templates and relationship templates, which transform recommendation data and KG into textual prompts. This not only eliminates semantic differences between them but also utilizes generic knowledge in PLMs.

4.2.1 Masked personalized prompts. In recommendation systems, personalized prompts refer to personalized fields for different users and items [9, 21]. Inspired by P5 [9], we construct a collection of MPP templates for SR, which helps the model discover various aspects of users and items. Specifically, MPP can transform the recommendation task into a pre-training task, namely a cloze task,

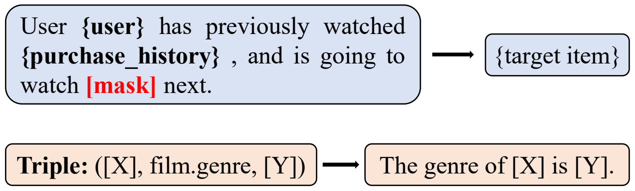

User **{user}** has previously watched **{purchase_history}** , and is going to watch **[mask]** next.
{target item}
Triple: ([X], film.genre, [Y])
The genre of [X] is [Y].

Figure 3: Examples of prompt templates. We can fill the data of user-item interaction sequences into templates to obtain MPP (up). For the relation template, we can obtain a triple prompt by replacing [X] and [Y] with the head entity and tail entity, respectively (down).

as shown in Figure 3. For a user u and his/her interaction sequence $\{A, B, C, D, E, F\}$, we can fill in the corresponding fields in the template to obtain: *User u has previously watched {A, B, C, D, E}, and is going to watch [mask] next.* Here, [mask] is the next item to be predicted, i.e., the target item F.

By constructing input-output pairs using MPP, we can leverage PLMs to extract rich semantics into user and item tokens, which will help capture users' dynamic preferences. MPP can transform the recommendation task into a pre-training task, improving task performance when downstream task data is sparse. In addition, it can also increase data utilization efficiency and accelerate model convergence.

4.2.2 Knowledge prompts. Using KG as side information in recommendation systems can significantly improve their performance. However, as structured data, KG cannot be directly input into PLMs, and there is a semantic gap between structured KG and sequence text. Therefore, we propose transforming the KG into a text sequence to address these issues.

For a triple (h, r, t), where h represents the head entity, r represents the relation, and t represents the tail entity, we manually design a relation template for each relation $r \in \mathcal{R}$ to express the semantics of the corresponding triple. For example, in Figure 3, we design a template for the relation *film.genre*: *The genre of [X] is [Y].* Then for the triple *(Cast Away, film.genre, Adventure)*, we replace *[X]* and *[Y]* with the head and tail entities, respectively, to obtain a basic triple prompt: *"The genre of Cast Away is Adventure.".*

For an entity h, we define the set of triples $S_1 = \{(h, r_1, t_1), (h, r_2, t_2), \ldots\}$ with h being the head entity as the 1-hop triple set of entity h, where $\{t_1, t_2, \ldots\}$ is the set of tail entities of entity h. Additionally, tail entities can also be treated as head entities, and they can have multiple relationships and tail entities as well. In S_1, there are multiple tail entities $\{t_1, t_2, \ldots\}$. For tail entity t_1, its set of 1-hop triplets can be represented as $S_{t1} = \{(t_1, r_{11}, t_{11}), (t_1, r_{12}, t_{12}), \ldots\}$. For tail entity t_2, its set of 1-hop triplets can be represented as $S_{t2} = \{(t_2, r_{21}, t_{21}), (t_2, r_{22}, t_{22}), \ldots\}$, and so on. Then, The set of 2-hop triplets for entity h can be represented as $S_2 = \{S_{t1}, S_{t2}, \ldots\}$. We convert each triple into a triple prompt, which can be used to generate multi-hop triple prompts for entity h. We then query with the multi-hop triple prompts for each item and combine them into a text sequence to construct KP.

By constructing relation templates, we can transform structured KG into text sequences to extract multi-level information simply.

4.2.3 Fused Prompt. After obtaining MPP and KP, we directly concatenate them as the input of PLM. Specifically, MPP can be represented as: $X_d = \{x_1, x_2, ..., [mask], ..., x_m\}$, where x_i is the i-th token of the text sequence, m represents the length of the text in tokens, and $[mask]$ represents the next item to be predicted. KP can be represented as: $X_k = \{x_1, x_2, ..., x_n\}$. The final input is:

$$X_{prompt} = [SPE]X_d[SPE]X_k[SPE]. \tag{1}$$

Here, $[SPE]$ denotes a special token.

By inputting the MPP and KP into PLMs, we can integrate the recommendation task into a full language environment and use the powerful ability of PLMs to extract users' fine-grained preferences.

4.3 Prompt Denoising

Converting KG to knowledge prompts can disrupt the original data structure and introduce a large amount of irrelevant and noisy knowledge. For example, the two triple prompts *"The genre of* h_1 *is* t_1*."* and *"The director of* h_2 *is* t_2*."* have different head and tail entities and are not logically or semantically related. The mutual influence between them will generate knowledge noise. Especially when the prompts contain multi-hop triple prompts, this noise will even be amplified. To alleviate the noise problem, we use MPP and triple prompts to construct a knowledge tree, and then propose the knowledge tree mask for denoising.

4.3.1 Konwledge tree construction. To have a clear understanding of the structure and semantic relationships between triple prompts, we construct a knowledge tree as shown in Figure 2.

The root node of the knowledge tree is the MPP, which contains multiple items that the user has interacted with. Therefore, the knowledge tree has multiple knowledge subtrees. Each knowledge subtree is composed of an entity and its multi-hop triple prompts. Suppose that the MPP contains two items corresponding to entities A and B, then the knowledge tree consists of two knowledge subtrees, namely $subTree(A)$ and $subTree(B)$. Suppose XX_i is a triple prompt consisting of (X, r, X_i), then the root node of knowledge subtree $Tree(A)$ is entity A, and the child nodes of A are its 1-hop triple prompts, AA_1 and AA_2. The 2-hop triple prompts of entity A are the 1-hop triple prompts of entities A_1 and A_2, namely, A_1A_{11}, A_1A_{12}, and A_2A_{21}. For example, in Figure1, the movie "Cast Away" can be represented by two triplets: (Cast Away, genre, Adventure) and (Cast Away, starred, Tom Hanks). We use A to represent "Cast Away", A_1 to represent "Adventure", and A_2 to represent "Tom Hanks." By introducing the relationship templates "The genre of [X] is [Y]." and "[X] starring [Y].", we obtain two triplet prompts for A: "AA_1: The genre of Cast Away is Adventure." and "AA_2: Cast Away starring Tom Hanks.". AA_1 and AA_2 are 1-hop triplet prompts for A.

Traversing the knowledge tree in a hierarchical manner yields a sequence of prompts composed of personalized and knowledge prompts.

4.3.2 Konwledge tree mask. The knowledge tree presents the logical and semantic relationships between MPP and triple prompts. Triple prompts without logical and semantic relationships will generate much noise, and we should limit their mutual influence. For this purpose, we propose a knowledge tree mask mechanism.

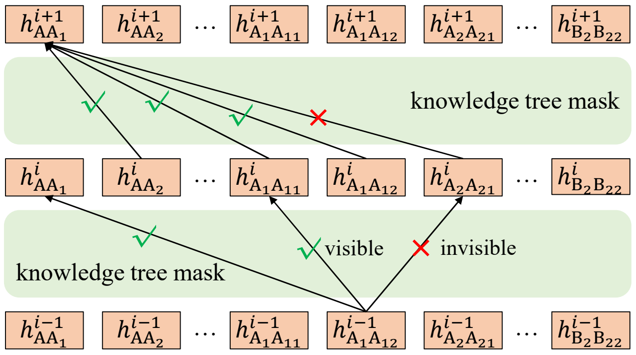


Figure 4: Illustration of prompt denoising. For any two triple prompts, if they contain the same entities, they are visible to each other; otherwise, they are invisible to each other.

Specifically, for the input sequence composed of MPP and KP, we use the mask matrix to limit the mutual influence of input tokens, as shown in Figure 2. The transformer-based language model usually uses an attention mask matrix to deal with the input problem of non-fixed-length sequences, which is formulated as follows:

$$Q^{i+1}, K^{i+1}, V^{i+1} = h^i W_q, h^i W_k, h^i W_v, \tag{2}$$

$$h^{i+1} = \text{softmax}\left(\frac{Q^{i+1}K^{i+1^\top} + M}{\sqrt{d_k}}\right)V^{i+1}. \tag{3}$$

Here, W^q, W^k, W^v are trainable parameters of the model, h^i is the hidden state output of the i-th layer, d_k is the scaling factor, and M is the attention mask matrix. By designing the attention mask matrix M, we can control the attention between input tokens. Therefore, we use the knowledge tree to design the mask matrix M, specifically:

$$M_{ij} = \begin{cases} 0 & x_i, x_j \in N_o \\ 0 & x_i \in N_1, x_j \in N_2 \\ 0 & x_i \in N_p, x_j \in N_c \text{ or } x_i \in N_c, x_j \in N_p \\ -\infty & \text{otherwise} \end{cases}. \tag{4}$$

Here, x_i and x_j are any two tokens in the prompt sequence, 0 means that the i-th token can see the j-th token, $-\infty$ means that the i-th token cannot see the j-th token, N_m represents a node in the knowledge tree, N_1 and N_2 are nodes with the same parent node, and N_p and N_c are a pair of parent and child nodes with a parent-child relationship in the knowledge tree.

The mask matrix M indicates that, for any node, it can see itself, parent nodes, child nodes, and sibling nodes. For example, the triple prompt AA_1 in Figure 4 can see the entity A, triple prompts AA_2, A_1A_{11}, and A_1A_{12}. It can be seen that any node and its visible nodes have the same entity, and invisible nodes contain different entities. Therefore, the knowledge tree mask matrix can keep the original knowledge structure of the knowledge prompt, solve the problem of irrelevant and noisy knowledge, and improve the model's performance.

4.4 Training and Recommendation

We employ the T5 model architecture [28],which is an encoder-decoder-based pre-trained language model using mask prediction

Table 1: Statistics of the datasets.

Stats.	books	music	movies
# Users	70,463	70,774	69,878
#Items	24,921	335,619	10,130
# Interactions	845,321	8,768,434	9,991,477
Density	4.8×10^{-4}	3.7×10^{-4}	1.4×10^{-2}
	Knowledge Graph		
# Relations	16	6	44
# Entities	103,695	1,434,216	123,306
# Triples	402,543	2,724,800	954,004

as the pre-training task. We construct personalized prompts with masks, transforming the recommendation task into a mask prediction task similar to the pre-training task of PLMs. The loss function is given by:

$$\mathcal{L}_\theta = -\sum_{j=1}^{|\mathbf{y}|} \log P_\theta \left(\mathbf{y}_j \mid \mathbf{y}_{<j}, \mathbf{X}_{\text{prompt}} \right). \tag{5}$$

Here, θ represents the model parameters, and y represents the predicted output.

During the recommendation stage, we use the same method as P5 [9], which applies beam search to generate a list of potential next items.

5 EXPERIMENTS

We conduct extensive experiments on three publicly available datasets to answer the following research questions:

- **RQ1:** Does KP4SR outperform the current state-of-the-art SR methods?
- **RQ2:** Does the conversion of the recommendation task to an NLP task follow the general patterns of the NLP field?
- **RQ3:** What is the impact of each component and hyperparameters in KP4SR?
- **RQ4:** Does KP4SR have generalization capability for unknown templates?

5.1 Experiments Settings

5.1.1 Datasets. We conduct experiments on three public datasets: Amazon books [12], LFM-1b [11], and Movielens-10M [30]. These datasets record the interaction information between users and books, music, and movies.

The KG we used is from KB4Rec [45], which links the above three widely used datasets with the widespread knowledge base Freebase[1] to provide side information for recommendation systems. We apply the same preprocessing steps as P5 [9] and filter out all users and items that appeared less than five times. For the LFM-1b dataset, we filter out tracks that were played less than 10 times. In addition, we search Freebase and convert all entity IDs to text. However, for items, we still use item IDs as inputs and outputs according to the approach in [9]. The statistics of the preprocessed datasets are shown in Table 1.

5.1.2 Evaluation. Following the prior work [9], we evaluate our KP4SR using Hit Rate@k (HR@k) and Normalized Discounted Cumulative Gain@k (NDCG@k), and report HR@5, 10 and NDCG@5, 10. Higher values indicate better performance for all metrics. We use the leave-one-out strategy to evaluate the performance of each method, which has been widely used in many related works [9]. Specifically, for each user-item interaction sequence, the last two items are kept as validation and test data, and the rest of the items are used to train the SR model. To make a fair comparison, we evaluate the model performance in a fully ranking manner. The ranking results are obtained on the entire item set rather than sampled results.

5.1.3 Baselines. We choose to use the following state-of-the-art SR methods as baselines in our experiments:

- **Caser** [33]: A CNN-based model that uses horizontal and vertical convolution filters to learn multiple patterns and user preferences.
- **GRU4Rec***: GRU4Rec [13] is a session-based RS that uses RNNs to capture sequential patterns. GRU4RecF [14] incorporates item attribute information. GRU4RecKG is an extension of GRU4Rec that connects items and their corresponding KG embeddings as inputs.
- **BERT4Rec** [32]: A bidirectional self-attention network that models user behavior sequences using cloze tasks.
- **SASRec***: SASRec [19] is an attention-based model that uses self-attention networks for SR. SASRecF is an extension of SASRec that connects items and their features as inputs.
- **KSR** [16]: An RNN and memory-based model that captures attribute-level user preferences using KG.
- **FDSA** [44]: A self-attention-based model that integrates heterogeneous features into a feature sequence for SR.
- **S^{3}Rec** [47]: A self-supervised learning-based model that has four carefully designed optimization objectives for learning correlations in raw data.
- **DIF-SR** [40]: An attention-based model that performs sequence recommendation by decoupling side information fusion.
- **P5** [9]: A PLM-based recommendation system that unifies multiple recommendation tasks into a single framework through personalized prompt sets.

5.1.4 Implementation Details. Our KP4RS utilizes a pre-trained T5 model as its backbone. For KP4RS, the encoder and decoder have six layers, a model dimension of 512, and 8 heads of attention. During the training phase, we use mixed precision to accelerate the training speed. We use 4 Ascend 910 NPUs with a batch size of 64. We use the AdamW optimizer [26] with a peak learning rate of 1e-3 and set the maximum length of the input tokens to 512. Due to the length limitation of the input (see Appendix for details), we only study the knowledge prompts within three hops.

For all baselines, if negative samples are needed to calculate the next item prediction loss during the training phase, we follow the usual practice of randomly selecting a negative sample for each interaction. For a fair comparison, the maximum interaction sequence length for all experiments is set to 5 unless otherwise specified.

Table 2: Overall performance. Bold scores represent the highest results of all methods. Underlined scores stand for the second highest results of all methods.

Methods	books				music				movies			
	NDCG@5	NDCG@10	HR@5	HR@10	NDCG@5	NDCG@10	HR@5	HR@10	NDCG@5	NDCG@10	HR@5	HR@10
Caser	0.0220	0.0294	0.0356	0.0587	0.0165	0.0232	0.0271	0.0477	0.0309	0.0462	0.052	0.0999
GRU4Rec	0.0235	0.0317	0.0380	0.0635	0.0222	0.0317	0.0374	0.0668	0.0378	0.0554	0.0643	0.1191
BERT4Rec	0.0204	0.0282	0.0323	0.0567	0.0242	0.0356	0.0426	0.0781	0.0328	0.0488	0.056	0.1062
SASRec	0.0254	0.0362	0.0466	0.0803	0.0327	0.047	0.0634	0.1078	0.0312	0.0459	0.0538	0.0994
GRU4RecF	0.0240	0.0321	0.0381	0.0633	0.0266	0.0377	0.0441	0.0788	0.0372	0.0551	0.0644	0.1204
GRU4RecKG	0.0233	0.0314	0.0373	0.0625	0.0222	0.0313	0.0380	0.0664	0.0374	0.0562	0.0650	0.1237
KSR	0.0240	0.0317	0.0383	0.0623	0.0330	0.0411	0.0504	0.0757	0.0394	0.0574	0.0679	0.1242
FDSA	0.0221	0.0309	0.0355	0.0631	0.0185	0.0261	0.0304	0.0539	0.0354	0.0523	0.0604	0.1132
SASRecF	0.0238	0.0319	0.0379	0.0631	0.0310	0.0418	0.0503	0.0839	0.0294	0.0441	0.0503	0.0964
S^{3}Rec	0.0249	0.0356	0.0452	0.0783	0.0301	0.0443	0.0524	0.0968	0.0306	0.0461	0.0536	0.1019
DIF-SR	0.0298	0.0416	0.0584	0.0948	0.0573	0.0678	**0.1110**	**0.1433**	0.0492	0.0689	0.0875	**0.1489**
P5	0.0433	0.0501	0.0604	0.0813	0.0815	0.0879	0.0994	0.1193	0.0618	0.0738	0.0888	0.1261
KP4SR	**0.0609**	**0.0691**	**0.0824**	**0.1077**	**0.0906**	**0.0975**	0.1108	0.1319	**0.0755**	**0.0891**	**0.1058**	0.1481

5.2 Overall Performance Comparison (RQ1)

The results of different methods on three datasets are shown in Table 2. According to the experimental results, we can see that, compared to the five SR methods (Caser, GRU4Rec, BERT4Rec, SASRec, and P5) that do not integrate side information, P5 outperforms other methods by a large margin. We believe there are two main reasons for this: firstly, P5 is a PLM-based method with better initialization parameters and a larger model that can accommodate more information. Secondly, P5 converts the recommendation task into an NLP task, which can leverage the general knowledge in PLMs. This indicates that PLMs have great potential for application in solving recommendation tasks. SASRec ranks second to P5 and performs well on books and music datasets, but it underperforms on the movie dataset. This suggests that different datasets have varying sequence patterns, impacting the performance of SR methods.

Some SR methods that integrate side information do not achieve better performance. For example, GRU4RecF and GRURecKG fuse item attribute information and KG information at an early stage, and they do not achieve better results. This is because early feature fusion may result in the loss of useful information. Similarly, SASRecF directly connects items and item features as input, and they also suffer from the information loss issue. In contrast, DIF-SR achieves much better results than other baselines. This is because DIF-SR decouples the fusion of side information, moves the side information from input to attention layers, and further decouples the attention calculation of various side information and item representations.

Finally, it is clear that our KP4SR achieves the best results in most evaluation metrics on all three datasets, especially outperforming other methods by a large margin on the NDCG metric. KP4SR and P5 use the same PLM architecture, but our method performs much better than P5 on all datasets. Specifically, for NDCG@5 and HR@5, our method achieved a 40.65% and 36.42% improvement on the books dataset, an 11.17% and 11.47% improvement on the music dataset, and a 22.17% and 19.14% improvement on the movies dataset. This is because KP4SR not only converts the recommendation task into a pre-training task but also efficiently utilizes domain knowledge.

Table 3: Pre-training fine-tuning vs Prompt-tuning

	method	NDCG@5	NDCG@10	HR@5	HR@10	epoch
books	P5	0.0433	0.0501	0.0604	0.0813	300
	MPP	0.0505	0.0570	0.0658	0.0861	90
	Improvement	16.63%	13.77%	8.94%	5.90%	-70%
music	P5	0.0815	0.0879	0.0994	0.1193	300
	MPP	0.0839	0.0904	0.1018	0.1220	140
	Improvement	2.94%	2.84%	2.41%	2.26%	-53.33%
movies	P5	0.0618	0.0738	0.0888	0.1261	260
	MPP	0.0645	0.0765	0.0906	0.1282	150
	Improvement	4.37%	3.66%	2.03%	1.67%	-42.31%

5.3 Pre-training fine-tuning vs Prompt-tuning (RQ2)

P5 uses a collection of personalized prompt templates to transform recommendation tasks into question-answering tasks. This differs from pre-training tasks and essentially belongs to the pre-training fine-tuning paradigm. We design the MPP, which can transform a recommendation task into a pre-training task, specifically a cloze task, as shown in Table 3. It can be observed that MPP achieves better performance on all three datasets and converges faster, reducing the number of training epochs by 70%, 53.33%, and 42.31%, respectively. Therefore, compared to first pre-training and then fine-tuning, transforming recommendation tasks into NLP tasks and then using prompt-tuning for recommendation is more competitive.

5.4 The impact of KP and PD (RQ3)

Next, we conduct experiments to investigate the impact of KP and PD on three datasets. The results are reported in Table 4. From the table, we can observe:

Firstly, without PD, the model performance gradually improves with the increase of knowledge prompt hops on books and music datasets. On the movies dataset, the best performance is achieved with 1-hop knowledge prompts, and too much knowledge can negatively affect performance. This suggests that PLMs can leverage knowledge prompts to help capture user preferences and improve recommendation performance, but the impact may vary across different datasets.

Table 4: Results when using different settings of KP and PD. *n* represents the number of KP hops.

setting		books		music		movies	
KP	PD	NDCG@5	HR@5	NDCG@5	HR@5	NDCG@5	HR@5
0	–	0.0505	0.0658	0.0839	0.1018	0.0645	0.0906
1	✗	0.0503	0.0697	0.0881	0.1077	0.0671	0.0944
2	✗	0.0522	0.0722	0.0885	0.1080	0.0650	0.0923
3	✗	0.0543	0.0753	0.0891	0.1093	0.0623	0.0892
1	✓	0.0542	0.0734	0.0894	0.1091	**0.0729**	**0.1012**
2	✓	**0.0575**	**0.0787**	0.0886	0.1086	0.0677	0.0959
3	✓	0.0523	0.0737	**0.0899**	**0.1102**	0.0702	0.0980

Table 5: Performance with different degrees of knowledge tree. *For the books dataset, the degrees are 2, 4, 6, and 8.

degrees	books		music		movies	
	NDCG@5	HR@5	NDCG@5	HR@5	NDCG@5	HR@5
2*	0.0575	0.0787	0.0890	0.1097	0.0729	0.1012
3*	0.0588	0.0807	0.0889	0.1097	0.0738	0.1032
4*	0.0608	0.0813	**0.0906**	**0.1108**	**0.0755**	**0.1058**
5*	**0.0609**	**0.0824**	0.0899	0.1102	0.0712	0.1005

Secondly, when the KP hops are the same, PD generally improves performance, indicating that the proposed PD can effectively mitigate knowledge noise. However, PD is not always effective as it may ignore potential relationships between triple prompts.

5.5 Impact of the degree of knowledge tree (RQ3)

We investigate the influence of the degree of the knowledge tree (i.e., the number of tail entities connected to the head entity) on recommendation performance. The results are shown in Table 5. On the books dataset, the performance improves as the degree increases, while on the music and movies datasets, the performance decreases after a certain degree. This phenomenon is consistent with the analysis in Section 5.4, which suggests that a certain amount of knowledge can improve performance, but excessive noise knowledge may lower performance.

5.6 Maximum sequence length

Due to the input length limitation, the maximum length of the user-item interaction sequence in the previous studies was set to 5. In this section, we investigate the effect of maximum interaction sequence length on recommendation performance, as shown in Figure 5. On the music dataset, the performance decreases as the maximum sequence length increases. On the books and movies datasets, the maximum sequence length has little impact on performance. This indicates that users' behavior is more dependent on their recent interactions with items, and a larger maximum sequence length may not necessarily lead to better performance, as it may introduce additional noise.

5.7 Generalization of KP4SR (RQ4)

We manually design 11 MPP templates for each dataset, with the first ten used for training, the first one used as the default testing template, and the eleventh used to test the model's generalization

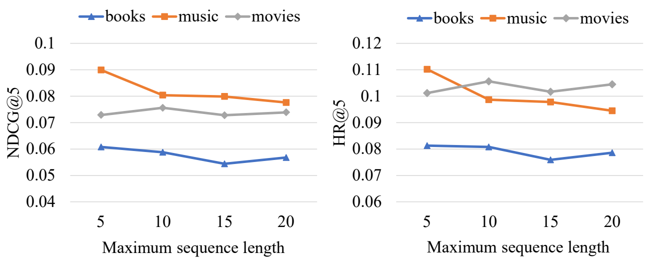


Figure 5: Performance with different maximum sequence length.

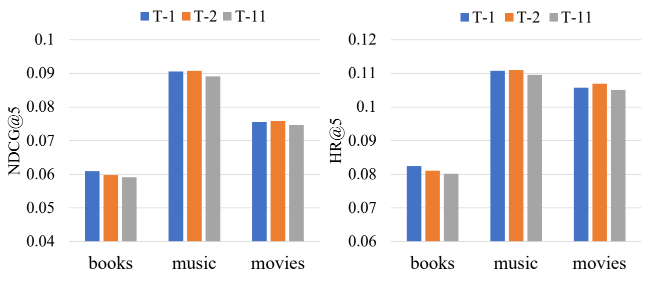


Figure 6: Performance with different prompt templates.

to unknown templates. We evaluate the performance of three templates, and the results are shown in Figure 6. Comparing templates 1 and 2, we can see that there are performance differences across different templates, indicating that better-designed templates can lead to better performance. At the same time, the model's performance only slightly decreases on unknown templates, indicating that KP4SR has a good generalization ability to unknown templates.

6 CONCLUSION AND FUTURE WORK

In this paper, we propose KP4SR, which is the first work that transforms KG into knowledge prompts to improve SR. Firstly, we design a set of MPP templates to transform the SR task into an NLP task, significantly improving the recommendation performance and convergence speed. Secondly, we construct a set of relation templates to transform the KG into KP. It not only addresses the problems of semantic differences but also enables the easy utilization of high-order information in the KG. Next, we propose PD to alleviate noise issues. The PD restores the data structure in the form of a mask matrix, which eliminates the impact of irrelevant triples. Finally, extensive experiments demonstrate the effectiveness of our method. In future work, we will continue to explore how to use large language models to achieve better recommendation performance.

ACKNOWLEDGMENTS

This work was supported by Key-Area Research and Development Program of Guangdong Provinc (2021B0101400002), NSFC (62276277), Guangdong Basic and Applied Basic Research Foundation (2022B1515120059), Guangdong Provincial Key Laboratory of Intellectual Property and Big Data (2018B030322016), The Major Key Project of PCL(PCL2021A13).

REFERENCES

[1] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*. 1247–1250.

[2] Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating Embeddings for Modeling Multi-relational Data. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger (Eds.). 2787–2795. https://proceedings.neurips.cc/paper/2013/hash/1cecc7a77928ca8133fa24680a88d2f9-Abstract.html

[3] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (Eds.). https://proceedings.neurips.cc/paper/2020/hash/1457c0d6bfcb4967418bfb8ac142f64a-Abstract.html

[4] Zeyu Cui, Jianxin Ma, Chang Zhou, Jingren Zhou, and Hongxia Yang. 2022. M6-Rec: Generative Pretrained Language Models are Open-Ended Recommender Systems. *ArXiv preprint* abs/2205.08084 (2022). https://arxiv.org/abs/2205.08084

[5] Gabriel de Souza Pereira Moreira, Sara Rabhi, Jeong Min Lee, Ronay Ak, and Even Oldridge. 2021. Transformers4rec: Bridging the gap between nlp and sequential/session-based recommendation. In *Proceedings of the 15th ACM Conference on Recommender Systems*. 143–153.

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, Minneapolis, Minnesota, 4171–4186. https://doi.org/10.18653/v1/N19-1423

[7] Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making Pre-trained Language Models Better Few-shot Learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Association for Computational Linguistics, Online, 3816–3830. https://doi.org/10.18653/v1/2021.acl-long.295

[8] Shijie Geng, Zuohui Fu, Yingqiang Ge, Lei Li, Gerard de Melo, and Yongfeng Zhang. 2022. Improving Personalized Explanation Generation through Visualization. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Dublin, Ireland, 244–255. https://doi.org/10.18653/v1/2022.acl-long.20

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

[10] Yuxian Gu, Xu Han, Zhiyuan Liu, and Minlie Huang. 2022. PPT: Pre-trained Prompt Tuning for Few-shot Learning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Dublin, Ireland, 8410–8423. https://doi.org/10.18653/v1/2022.acl-long.576

[11] F. Maxwell Harper and Joseph A. Konstan. 2016. The MovieLens Datasets: History and Context. *ACM Trans. Interact. Intell. Syst.* 5, 4 (2016), 19:1–19:19. https://doi.org/10.1145/2827872

[12] Ruining He and Julian J. 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, WWW 2016, Montreal, Canada, April 11 - 15, 2016*, Jacqueline Bourdeau, Jim Hendler, Roger Nkambou, Ian Horrocks, and Ben Y. Zhao (Eds.). ACM, 507–517. https://doi.org/10.1145/2872427.2883037

[13] 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

[14] Balázs Hidasi, Massimo Quadrana, Alexandros Karatzoglou, and Domonkos Tikk. 2016. Parallel Recurrent Neural Network Architectures for Feature-rich Session-based Recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems, Boston, MA, USA, September 15-19, 2016*, Shilad Sen, Werner Geyer, Jill Freyne, and Pablo Castells (Eds.). ACM, 241–248. https://doi.org/10.1145/2959100.2959167

[15] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative filtering for implicit feedback datasets. In *2008 Eighth IEEE international conference on data mining*. Ieee, 263–272.

[16] Jin Huang, Wayne Xin Zhao, Hongjian Dou, Ji-Rong Wen, and Edward Y. Chang. 2018. Improving Sequential Recommendation with Knowledge-Enhanced Memory Networks. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08-12, 2018*, Kevyn Collins-Thompson, Qiaozhu Mei, Brian D. Davison, Yiqun Liu, and Emine Yilmaz (Eds.). ACM, 505–514. https://doi.org/10.1145/3209978.3210017

[17] Xiaowen Huang, Quan Fang, Shengsheng Qian, Jitao Sang, Yan Li, and Changsheng Xu. 2019. Explainable Interaction-driven User Modeling over Knowledge Graph for Sequential Recommendation. In *Proceedings of the 27th ACM International Conference on Multimedia, MM 2019, Nice, France, October 21-25, 2019*, Laurent Amsaleg, Benoit Huet, Martha A. Larson, Guillaume Gravier, Hayley Hung, Chong-Wah Ngo, and Wei Tsang Ooi (Eds.). ACM, 548–556. https://doi.org/10.1145/3343031.3350893

[18] Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2020. How Can We Know What Language Models Know? *Transactions of the Association for Computational Linguistics* 8 (2020), 423–438. https://doi.org/10.1162/tacl_a_00324

[19] Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive sequential recommendation. In *2018 IEEE international conference on data mining (ICDM)*. IEEE, 197–206.

[20] Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The Power of Scale for Parameter-Efficient Prompt Tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Online and Punta Cana, Dominican Republic, 3045–3059. https://doi.org/10.18653/v1/2021.emnlp-main.243

[21] Lei Li, Yongfeng Zhang, and Li Chen. 2022. Personalized prompt learning for explainable recommendation. *ArXiv preprint* abs/2202.07371 (2022). https://arxiv.org/abs/2202.07371

[22] Xiang Lisa Li and Percy Liang. 2021. Prefix-Tuning: Optimizing Continuous Prompts for Generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Association for Computational Linguistics, Online, 4582–4597. https://doi.org/10.18653/v1/2021.acl-long.353

[23] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning Entity and Relation Embeddings for Knowledge Graph Completion. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA*, Blai Bonet and Sven Koenig (Eds.). AAAI Press, 2181–2187. http://www.aaai.org/ocs/index.php/AAAI/AAAI15/paper/view/9571

[24] Peng Liu, Lemei Zhang, and Jon Atle Gulla. 2023. Pre-train, Prompt and Recommendation: A Comprehensive Survey of Language Modelling Paradigm Adaptations in Recommender Systems. *ArXiv preprint* abs/2302.03735 (2023). https://arxiv.org/abs/2302.03735

[25] Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Qi Ju, Haotang Deng, and Ping Wang. 2020. K-BERT: Enabling Language Representation with Knowledge Graph. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*. AAAI Press, 2901–2908. https://aaai.org/ojs/index.php/AAAI/article/view/5681

[26] Ilya Loshchilov and Frank Hutter. 2019. Decoupled Weight Decay Regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net. https://openreview.net/forum?id=Bkg6RiCqY7

[27] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog* 1, 8 (2019), 9.

[28] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *J. Mach. Learn. Res.* 21 (2020), 140:1–140:67. http://jmlr.org/papers/v21/20-074.html

[29] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing personalized Markov chains for next-basket recommendation. In *Proceedings of the 19th International Conference on World Wide Web, WWW 2010, Raleigh, North Carolina, USA, April 26-30, 2010*, Michael Rappa, Paul Jones, Juliana Freire, and Soumen Chakrabarti (Eds.). ACM, 811–820. https://doi.org/10.1145/1772690.1772773

[30] Markus Schedl. 2016. The lfm-1b dataset for music retrieval and recommendation. In *Proceedings of the 2016 ACM on international conference on multimedia retrieval*. 103–110.

[31] Guy Shani, David Heckerman, Ronen I Brafman, and Craig Boutilier. 2005. An MDP-based recommender system. *Journal of Machine Learning Research* 6, 9 (2005).

[32] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019. BERT4Rec: Sequential Recommendation with Bidirectional Encoder Representations from Transformer. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3-7, 2019*, Wenwu Zhu, Dacheng Tao, Xueqi Cheng, Peng Cui, Elke A. Rundensteiner, David Carmel, Qi He, and Jeffrey Xu Yu (Eds.). ACM, 1441–1450. https://doi.org/10.1145/3357384.3357895

[33] Jiaxi Tang and Ke Wang. 2018. Personalized Top-N Sequential Recommendation via Convolutional Sequence Embedding. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, WSDM 2018, Marina Del Rey, CA, USA, February 5-9, 2018*, Yi Chang, Chengxiang Zhai, Yan Liu, and Yoelle Maarek (Eds.). ACM, 565–573. https://doi.org/10.1145/3159652.3159656

[34] Lingzhi Wang, Huang Hu, Lei Sha, Can Xu, Daxin Jiang, and Kam-Fai Wong. 2022. RecInDial: A Unified Framework for Conversational Recommendation with Pretrained Language Models. In *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing*. 489–500.

[35] Xiaolei Wang, Kun Zhou, Ji-Rong Wen, and Wayne Xin Zhao. 2022. Towards Unified Conversational Recommender Systems via Knowledge-Enhanced Prompt Learning. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 1929–1937.

[36] Chuhan Wu, Fangzhao Wu, Tao Qi, and Yongfeng Huang. 2021. Empowering news recommendation with pre-trained language models. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1652–1656.

[37] Shitao Xiao, Zheng Liu, Yingxia Shao, Tao Di, Bhuvan Middha, Fangzhao Wu, and Xing Xie. 2022. Training large-scale news recommenders with pretrained language models in the loop. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 4215–4225.

[38] Tao Xie, Yangjun Xu, Liang Chen, Yang Liu, and Zibin Zheng. 2021. Sequential recommendation on dynamic heterogeneous information network. In *2021 IEEE 37th International Conference on Data Engineering (ICDE)*. IEEE, 2105–2110.

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

[40] Yueqi Xie, Peilin Zhou, and Sunghun Kim. 2022. Decoupled side information fusion for sequential recommendation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1611–1621.

[41] Hongbin Ye, Ningyu Zhang, Shumin Deng, Xiang Chen, Hui Chen, Feiyu Xiong, Xi Chen, and Huajun Chen. 2022. Ontology-enhanced Prompt-tuning for Few-shot Learning. In *Proceedings of the ACM Web Conference 2022*. 778–787.

[42] Zeping Yu, Jianxun Lian, Ahmad Mahmoody, Gongshen Liu, and Xing Xie. 2019. Adaptive User Modeling with Long and Short-Term Preferences for Personalized Recommendation. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, Sarit Kraus (Ed.). ijcai.org, 4213–4219. https://doi.org/10.24963/ijcai.2019/585

[43] Wei Zeng, Xiaozhe Ren, Teng Su, Hui Wang, Yi Liao, Zhiwei Wang, Xin Jiang, ZhenZhang Yang, Kaisheng Wang, Xiaoda Zhang, et al. 2021. PanGu-α: Large-scale Autoregressive Pretrained Chinese Language Models with Auto-parallel Computation. *ArXiv preprint* abs/2104.12369 (2021). https://arxiv.org/abs/2104.12369

[44] Tingting Zhang, Pengpeng Zhao, Yanchi Liu, Victor S. Sheng, Jiajie Xu, Deqing Wang, Guanfeng Liu, and Xiaofang Zhou. 2019. Feature-level Deeper Self-Attention Network for Sequential Recommendation. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, Sarit Kraus (Ed.). ijcai.org, 4320–4326. https://doi.org/10.24963/ijcai.2019/600

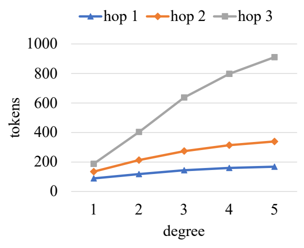
[45] Wayne Xin Zhao, Gaole He, Kunlin Yang, Hong-Jian Dou, Jin Huang, Siqi Ouyang, and Ji-Rong Wen. 2019. KB4Rec: A Data Set for Linking Knowledge Bases with Recommender Systems. *Data Intelligence* 1, 2 (2019), 121–136. https://doi.org/10.1162/dint_a_00008

[46] 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*. 2256–2267.

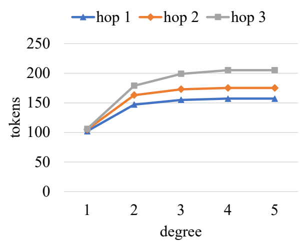
[47] Kun Zhou, Hui Wang, Wayne Xin Zhao, Yutao Zhu, Sirui Wang, Fuzheng Zhang, Zhongyuan Wang, and Ji-Rong Wen. 2020. S3-Rec: Self-Supervised Learning for Sequential Recommendation with Mutual Information Maximization. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*, Mathieu d'Aquin, Stefan Dietze, Claudia Hauff, Edward Curry, and Philippe Cudré-Mauroux (Eds.). ACM, 1893–1902. https://doi.org/10.1145/3340531.3411954

A LENGTH OF INPUT TOKENS

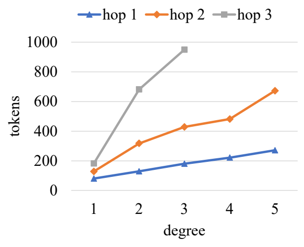
We statistic the input tokens length of three datasets under different conditions, as shown in Figure 7. Each data point represents the median length of all samples rather than the mean. For KP4SR, the default maximum input length is 512. Therefore, in Section 5.3, we set the degree to 2 for the books and movies datasets, and set the degree to 5 for the music dataset. In Section 5.6, we set the hop to 2 and the degree to 6 for the books dataset, and set the hop to 1 and the degree to 2 for the music and movies datasets. Figure 7(d) shows the length of input samples under different maximum sequence lengths.



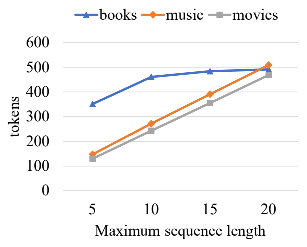
(a) books



(b) music



(c) movies



(d) items

Figure 7: Length of input tokens