Contrastive Pre-training for Sequential Recommendation

Xu Xie[1], Fei Sun[2], Zhaoyang Liu[2], Jinyang Gao[2], Bolin Ding[2], Bin Cui[1]

[1]School of EECS & Key Laboratory of High Confidence Software Technologies (MOE), Peking University

[2]Alibaba Group

{xu.xie,bin.cui}@pku.edu.cn,{ofey.sf,jingmu.lzy,jinyang.gjy,bolin.ding}@alibaba-inc.com

ABSTRACT

Sequential recommendation methods play a crucial role in modern recommender systems because of their ability to capture users' dynamic interests from their historical interactions. Despite their success, we argue that these approaches require huge amounts of parameters to learn a high-quality user representation model. However, they usually suffer from the data sparsity problem, which makes it difficult for them to collect sufficient supervised information to optimize the parameters. To tackle that, inspired by recent advances of pre-training techniques in the natural language processing area, we construct the training signal from unsupervised data and then pre-train the user representation model with this information. We propose a novel model called **C**ontrastive **P**re-training for Sequential **Rec**ommendation (**CP4Rec**), which utilizes the contrastive pre-training framework to extract meaningful user patterns and further encode the user representation effectively. In addition, we propose three data augmentation approaches to construct pre-training tasks and exploit the effects of the composition of different augmentations. Comprehensive experiments on four public datasets demonstrate that CP4Rec achieves state-of-the-art performance over existing baselines especially when limited training data is available.

CCS CONCEPTS

• **Computing methodologies** → **Unsupervised learning**; • **Information systems** → *Recommender systems*.

KEYWORDS

Contrastive Learning; Pre-training; Recommender Systems

1 INTRODUCTION

Recommender systems have been widely employed in online platforms, like Amazon and Alibaba, to satisfy the users' requirements. On these platforms, users' interests hidden in their behaviors are intrinsically dynamic and evolving over time, which makes it difficult for platforms to make appropriate recommendations. To cope with this problem, various methods have been proposed to make *sequential recommendations* by capturing users' dynamic interests from their historical interactions [21, 25, 40–42].

For sequential recommendation task, the most essential problem is how to infer an accurate representation for each user through their historical interactions. With these accurate user representations, we can easily recommend suitable items for each user. Thus, the main line of research work seeks to derive better user representation using more powerful sequential models. Recently, with the advances of deep learning techniques, a lot of works employ deep neural networks to handle this problem and obtain significant performance improvements [21, 25, 42]. These sequential models, such as Recurrent Neural Network (RNN) [8] and Self-attention [43], can learn effective representations of users' behaviors by capturing more complicated sequential patterns. Although these methods have achieved promising results, they require sufficient data to optimize these huge amounts of parameters [50]. When the supervised training data is limited, these methods may fail to infer appropriate user representations.

Similar challenges have also been faced in other domains, such as natural language processing (NLP) and computer vision (CV). In recent years, many works have made a great breakthrough in this problem [4, 10, 36, 38]. They attempt to tackle this challenge by pre-training techniques, which first pre-train the model on a large unlabeled corpus and then finetune the learned model with supervised labels from downstream tasks. In the pre-training stage, training signals are directly constructed from the unlabeled data to learn a universal language or image representation model. For example, in the NLP area, Brown et al. [3] pre-train an autoregressive language model with billions of parameters and achieves strong performance on many NLP tasks, especially in the few-shot setting.

Inspired by the successes of pre-training in computer vision and natural language processing [4, 10, 36, 38], we aim to use pre-training to optimize user representation model for improving sequential recommender systems. To achieve this goal, one straightforward way could be to directly adopt a powerful sequential model like GPT [38] on a larger user behavior corpus. However, this way is not suitable for recommender system for two reasons. First, the objective function of such predictive self-supervised learning is almost the same as the goal of sequential recommendation which is often modeled as sequential prediction task. Second, recommendation systems usually do not have a larger corpus for pre-training. The same objective function on the same data cannot help the user representation modeling in sequential recommendation.

Due to the issues mentioned before, the application of self-supervised learning in recommendation systems is less well studied. The closest line of research seeks to enhance feature representations

via self-supervision signals derived from the intrinsic structure of the raw feature data, e.g., item attributes [50, 55]. However, in many scenarios, additional side information is often not available, and recommender systems can only utilize user-item interaction information.

Different from previous study focusing on enhancing item feature representation in recommender system, we aim to improve user representation model via self-supervised signals constructed from user interaction data with only ID information. Specifically, we propose a novel model called **C**ontrastive **P**re-training for Sequential **Rec**ommendation (CP4Rec). Our model first pre-trains the sequential model with self-supervised signals and then fine-tune the model with the supervised labels. In the pre-training stage, we propose three data augmentation methods (crop/mask/reorder) to project user interaction sequences to different views. With the contrastive learning loss, we encode the user representation by maximizing the agreement between differently augmented views of the same user interaction sequence in the latent space. In this way, CP4Rec can infer accurate user representations and then easily select appealing items for each user individually. We conduct extensive experiments on many real-world public recommendation datasets. Comprehensive experimental results verify that CL4Rec achieves state-of-the-art performance compared with many competitive methods, especially when limited training data is available.

Our primary contributions can be summarized as follows:

- We propose a novel model called Contrastive Pre-training for Sequential Recommendation (CP4Rec) which infers accurate user representations with only users' interaction behaviors. To the best of our knowledge, this is the first work to apply contrastive learning to the sequential recommendation.
- We propose three different data augmentation approaches including cropping, masking, and recording in the pre-training stage. The effects of the composition of multiple data augmentation operations are also exploited in this paper.
- Comprehensive experiments on four public datasets demonstrate the effectiveness of our CP4Rec model. Compared with all competitive baselines, the improvements brought by the contrastive learning framework is almost 4.74% to 9.76% according to the ranking metric on average.

2 RELATED WORK

In this section, we will review previous works related to ours in two aspects, including sequential recommendation and self-supervised learning.

2.1 Sequential Recommendation

Early works on sequential recommendations usually utilize the theory of Markov Chain (MC) to capture sequential patterns from successive item sequences. For example, Rendle et al. [40] combine first-order MC and matrix factorization to capture both the smoothness of subsequent actions and long-term user preference (FPMC). Besides the first-order MCs, there are also methods adopting high-order MC, which consider more preceding items to extract more complicated sequential patterns [17, 18].

With the recent advance of deep learning, there are many works that are equipped with these powerful tools and achieve significant

breakthrough in sequential recommendation tasks [6, 12, 18, 21, 25, 42, 49, 52, 53]. Recurrent neural networks (RNN) [8] and its variants, such as Long Short-Term Memory (LSTM) [22] and Gated Recurrent Unit (GRU) [7], which are designed for sequential data, are appropriate to model user behavior sequences. For example, Hidasi et al. [21] firstly adopt GRU modules to the session-based recommendation task with ranking loss. The following variants modified GRU4Rec by introducing attention mechanism [5, 31, 32], hierarchical structure [37], user-based GRU [12], Graph Neural Networks [15], etc.

Other deep learning models are also adopted to sequential recommendation tasks and achieve excellent performance. For instance, Tang and Wang [42] propose a Convolutional Sequence Embedding Recommendation Model (Caser) to learn sequential patterns as local features of the "image" using both horizontal and vertical convolutional filters. Chen et al. [6] and Huang et al. [24] leverage the memory-augmented neural network (MANN) to store and update useful information explicitly. HGN [33] captures both the long-term and short-term user interest by introducing a hierarchical gating network including a feature gating module, an instance gating module, and an item-item product module.

Recently, the self-attention network [43] has shown promising potential in modeling sequential data. Inspired by this mechanism, Kang and McAuley [25] utilize Transformer layers to adaptively assign weights to previous items in the sequential recommendation task. Sun et al. [41] improve that by adopting a bidirectional Transformer to incorporate user behaviors information from both directions, since the user's historical interactions may not be a rigid order [23]. Wang et al. [45] further equip Transformer with Hypergraph Neural Networks [13] to capture the dynamic meanings of items across time and users.

2.2 Self-supervised Learning

Self-supervised learning has aroused comprehensive awareness recently, since it concentrates on learning useful representations from unlabeled data, which benefits the downstream tasks. Its basic idea is to construct training signals by designing pretext tasks within the raw data and then utilize them to train the model. The pretext tasks usually transform the raw data into another near-equivalent form and then predict the details of the transformation (*prediction task*) or distinguish whether samples are transformed from the same data (*discrimination task*).

The prediction tasks have been widely adopted in Computer Vision and Natural Language Processing. In the computer vision community, various self-supervised learning pretext tasks have been widely exploited [27], such as predicting image rotations [14], predicting relative patch locations [11], solving jigsaw puzzles [36], and solving colorization problems [54]. When it comes to the area of Natural Language Processing, many works are focusing on self-supervised techniques to acquire universal word representations. The language model is a popular self-supervised objective in this area that learns to predict the next word given the previous words [35]. There are also other pretext tasks in the NLP area, such as the Cloze task and next sentence prediction, and so on [10, 38]. Though these prediction tasks have achieved promising results,

they rely on ad-hoc heuristics, which may affect the generality of the representations [4].

In contrast, the discrimination tasks adopt a contrastive learning technique that tries to distinguish the positive sample from massive negative samples in a classification manner. Wu et al. [47] claims that the use of rich negative samples can enhance the performance of contrastive learning, so they propose an approach to sample negative instances from a shared memory bank. Later, He et al. [16] equip the memory bank with a dynamic queue and a moving-averaged encoder to build large and consistent dictionaries. There are also other works that exploit the use of in-batch negative samples rather than a memory bank [4, 51].

When it comes to the field of recommendation systems, some works apply self-supervised learning to improve recommendation performance. For example, Xin et al. [48] propose Self-Supervised Q-learning (SQN) and Self-Supervised Actor-Critic (SAC) to consider long-term user engagement using multiple types of users' behaviors. Yao et al. [50] present a self-supervised learning framework to learn useful representations of items that have a set of categorical features. Later, Zhou et al. [55] design auxiliary self-supervised objectives to learn the correlations among attribute, item, subsequence, and sequence. Different from the above methods, our model focuses on how to learn useful user representations just from a single type of users' behaviors without auxiliary information.

3 CP4REC

In this section, we present our contrastive pre-training framework for sequential recommendation (CP4Rec), which only utilizes information of users' historical behaviors. We first introduce the notations used in this paper and formulate the sequential recommendation problem in Section 3.1. Then, we present our general contrastive pre-training framework in Section 3.2. In Section 3.3, we propose three augmentation methods to pre-train the base model. In section 3.4, we introduce the user representation model used in our approach. Finally, we propose how to fine-tune the pre-trained user representation model. Since our CP4Rec is a general framework, we select the state-of-the-art model—Transformer [43] as our user representation model.

3.1 Notations and Problem Statement

In this paper, we represent column vectors and matrices by bold italic lower case letters (e.g., $\boldsymbol{u}$, $\boldsymbol{v}$) and bold upper case letters (e.g., $\boldsymbol{R}$), respectively. The j-th row of a matrix $\boldsymbol{R}$ is represented by $\boldsymbol{R}_j^\top$. And we use calligraphic letters to represent sets (e.g., $\mathcal{U}$, $\mathcal{V}$, $\mathcal{A}$).

Let $\mathcal{U}$ and $\mathcal{V}$ denote a set of users and items respectively, where $|\mathcal{U}|$ and $|\mathcal{V}|$ denote the numbers of users or items. We represent a user or an item with $u \in \mathcal{U}$ or $v \in \mathcal{V}$. In sequential recommendation tasks, users' behavior sequences are usually in chronological order. Therefore, we represent the interaction sequence for user u with $s_u = [v_1^{(u)}, v_2^{(u)}, \ldots, v_{|s_u|}^{(u)}]$, where $v_t^{(u)}$ denotes the item which user u interacts at the timestep t and $|s_u|$ denotes the length of interaction sequence for user u. Also, let $\mathcal{A}$ denotes a set of augmentations that may be applied in the pre-training tasks.

Based on the above notations, we now define the task for the sequential recommendation. It focuses on predicting the most possible item which the user u will interact with at the timestamp
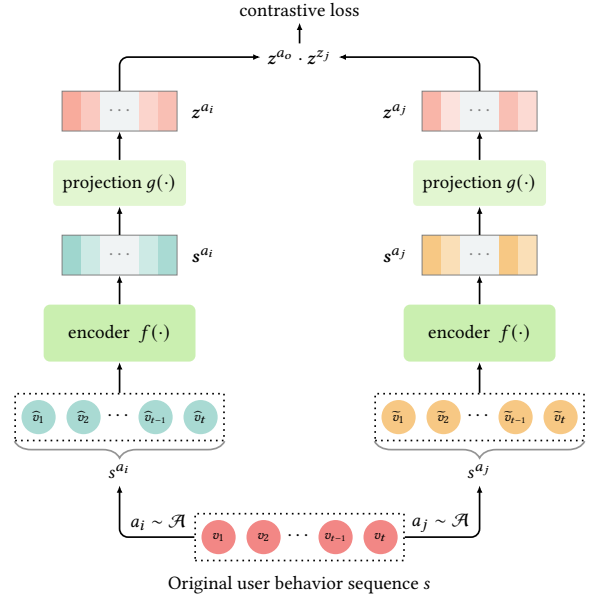

Figure 1: A simple framework for CP4Rec. Two data augmentation methods,a_i and a_j, are sampled from the same augmentation set $\mathcal{A}$. They are applied to each user's sequence and then we can obtain two correlated views of each sequence. A user representation model $f(\cdot)$ and a projection module $g(\cdot)$ transform the augmented sequences to the latent space where the contrastive loss is applied.

$|s_u| + 1$, given only one type of her/his historical interaction sequences without any other auxiliary contextual information. It can be formulated as follows:

$$v_u^* = \arg\max_{v_i \in \mathcal{V}} P\left(v_{|s_u|+1}^{(u)} = v_i \mid s_u\right). \tag{1}$$

3.2 The Contrastive Learning Framework

Inspired by the SimCLR framework [4] for learning visual representation and framework proposed by Yao et al. [50] for learning representations of categorical features, we borrow a similar idea of contrastive learning algorithms to obtain a powerful user representation model. The framework comprises four major components, including a stochastic data augmentation module, a user representation encoder, an auxiliary projection module, and a contrastive loss function. The framework is illustrated in Figure 1.

3.2.1 Data Augmentation Module. A stochastic data augmentation module is employed to transform each data sample randomly into two correlated instances. If the two transformed instances are from the same sample, they are treated as the positive pair. If they are transformed from different samples, they are treated as the negative pair. In the sequential recommendation task, we apply two randomly sampled augmentation methods ($a_i \in \mathcal{A}$ and $a_j \in \mathcal{A}$) to each user's historical behaviors sequence s_u, and obtain two views of the sequence, denoting $s_u^{a_i}$ and $s_u^{a_j}$.

3.2.2 User Representation Encoder. We utilize a neural network as a user representation encoder to extract information from the

augmented sequences. With this encoder, we can obtain meaningful user representations from their augmented sequences, which is $s_u^a = f(s_u^a)$. Since our CP4Rec has no constraints on the choice of user representation model, in this work, we adopt the Transformer [43] model to encode the user representation, which has shown the promising results in recent works [25, 41].

3.2.3 Auxiliary Projection Module.
An auxiliary projection module maps the user representation to another space where the contrastive loss is applied. With this module, we obtain the final representation for each augmented sequence:

$$z_u^a = g(s_u^a) = g(f(s_u^a)). \tag{2}$$

In this work, we use a linear transformation as the projection module for simplicity. As illustrated in SimCLR [4], additional projection after user representation encoder can remove information that may be useful for the downstream task. To maintain more information in the user representations s_u, we throw away this auxiliary projection in the SimCLR framework and only retain the user representation encoder $f(\cdot)$ at the fine-tuning procedure.

3.2.4 Contrastive Loss Function.
Finally, a contrastive loss function is applied to distinguish whether the two representations are derived from the same user historical sequence. To achieve this target, this loss learns to minimize the difference between differently augmented views of the same user historical sequence and maximize the difference between the augmented sequences derived from different users. Considering a mini-batch of N users $u_1, u_2, \ldots, u_N$, we apply two random augmentation operators to each user's sequence and obtain $2N$ augmented sequences $[s_{u_1}^{a_i}, s_{u_1}^{a_j}, s_{u_2}^{a_i}, s_{u_2}^{a_j}, \ldots, s_{u_N}^{a_i}, s_{u_N}^{a_j}]$. Similar to Chen et al. [4] and Yao et al. [50], for each user u, we treat $(s_u^{a_i}, s_u^{a_j})$ as the positive pair, and treat other $2(N-1)$ augmented examples within the same minibatch as negative samples S^-. We utilize cosine similarity to measure the difference between each representation, $\text{sim}(u, v) = \cos(u, v) = u^\top v / \|u\| \|v\|$. Then the loss function for a positive pair $(s_u^{a_i}, s_u^{a_j})$ can be defined similar to the widely used softmax cross entropy loss as:

$$\mathcal{L}(s_u^{a_i}, s_u^{a_j}, S^-) = -\log \frac{\exp(\text{sim}(z_u^{a_i}, z_u^{a_j})/\tau)}{\exp\left(\frac{\text{sim}(z_u^{a_i}, z_u^{a_j})}{\tau}\right) + \sum_{s^- \in S^-} \exp\left(\frac{\text{sim}(z_u^{a_i}, \widetilde{z}_{s^-})}{\tau}\right)}, \tag{3}$$

where S^- is negative sample set for the positive pair $(s_u^{a_i}, s_u^{a_j})$ and τ is a hyper-parameter for softmax temperature.

3.3 Data Augmentation Operators

Based on the above contrastive learning framework, we next discuss the design of the transformations in the contrastive learning framework which can incorporate additional self-supervised signals to enhance the user representation model. We introduce three augmentation approaches which can construct different views of the same sequence but still maintain the main preference hidden in historical behaviors. These operators are briefly illustrated in Figure 2.

3.3.1 Item Crop.
Random crop is a common data augmentation technique to increase the variety of images in the computer vision. It usually creates a random subset of an original image to help the model generalize better. Inspired by the random crop technique in

images, we propose the item crop augmentation method for the contrastive learning task in the sequential recommendation. For each user's historical sequence s_u, we random select a continuous sub-sequence: $s_u^{\text{crop}} = [v_c, v_{c+1}, \ldots, v_{c+L_c-1}]$ with the sequence length $L_c = \lfloor \eta * |s_u| \rfloor$. Therefore, this augmentation method can be formulated as:

$$s_u^{\text{crop}} = a_{\text{crop}}(s_u) = [v_c, v_{c+1}, \ldots, v_{c+L_c-1}]. \tag{4}$$

The effect of our item crop augmentation method can be explained in two aspects. On the one hand, it provides a local view of the user's historical sequence. It enhances the user representation model by learning a generalized user preference without comprehensive information of users. On the other hand, under the contrastive learning framework, if the two cropped sequences have no intersection, it can be regarded as a next sentence prediction task [10]. It pushes the model to predict the change of the user's preference.

3.3.2 Item Mask.
The technique of randomly zero-masking input word, which is also called "word dropout", is widely adopted to avoid overfittings in many natural language processing tasks, such as sentence generation [2], sentiment analysis [9], and question answering [29]. Inspired by this word dropout technique, we propose to apply a random item mask as one of the augmentation methods for contrastive learning. For each user historical sequence s_u, we random mask a proportion γ of items $\mathcal{T}_{s_u} = (t_1, t_2, \ldots, t_{L_m})$ with the length $L_m = \lfloor \gamma * |s_u| \rfloor$. If the item in the sequence is masked, it is replaced by a special item [mask]. Therefore, this augmentation method can be formulated as:

$$s_u^{\text{mask}} = a_{\text{mask}}(s_u) = [\widehat{v}_1, \widehat{v}_2, \ldots, \widehat{v}_{|s_u|}]$$
$$\widehat{v}_t = \begin{cases} v_t, & t \notin \mathcal{T}_{s_u} \\ [\text{mask}] & t \in \mathcal{T}_{s_u}. \end{cases} \tag{5}$$

Since a user's intention is relatively stable over a period of time, user's historical interacted items mostly reflect a similar purpose. For example, if a user intends to purchase a pair of sports shoes, (s)he may click many sports shoes to decide which ones to buy. Therefore, with our random item mask augmentation, the two different views derived from the same user sequence can still reserve the main intention of the user. In this way, this self-supervised signal can prevent the user representation encoder from co-adapting too much.

3.3.3 Item Reorder.
Many approaches employ the strict order assumption that most adjacent items in the users' historical sequences are sequentially dependent. For example, Rendle et al. [40] assume that each item only correlates to its former one. Hidasi et al. [21] also assume items are arranged in a uni-direction order. However, in the real world, sometimes the order of users' interactions are in a flexible manner [42, 46]. As illustrated by Wang et al., the sequence [milk, butter, flour] is almost equivalent to the sequence [*butter*, *milk*, *flour*], since there is no strict order between milk and butter, but the union of them leads to a high purchase probability of flour. Therefore, we can derive a self-supervised augmentation operator to capture the sequential dependencies under the assumption of flexible order. With this operator, we can encourage the user representation model to rely less on the order of interaction sequences,
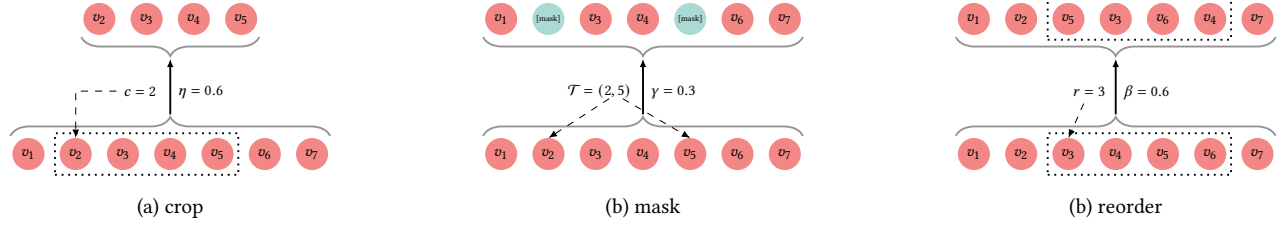
Figure 2: A brief illustration of augmentation operations applied in our CP4Rec model, including item crop, item mask, and item reorder.

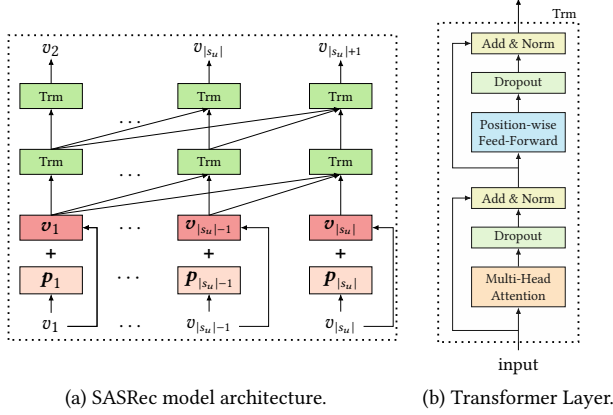(a) SASRec model architecture.   (b) Transformer Layer.

Figure 3: A brief architecture of SASRec model and Transformer Encoder Layer.

thus enhancing the model to be more robust when encountering new interactions.

For this purpose, we adopt the item reorder task as another augmentation method for contrastive learning. Inspired by swap operations in the natural language processing area [1, 30], we alter the order of items in the user sequence with a proportion of β by randomly shuffling their positions. More concretely, for each user historical sequence s_u, we randomly shuffle a continuous subsequence $[v_r, v_{r+1}, \ldots, v_{r+L_r-1}]$, which starts at r with length $L_r = \lfloor \beta * |s_u| \rfloor$, to $[\widehat{v}_r, \widehat{v}_{r+1}, \ldots, \widehat{v}_{r+L_r-1}]$. This augmentation method can be formulated as:

$$s_u^{\text{reorder}} = a_{\text{reorder}}(s_u) = \left[v_1, v_2, \ldots, \widehat{v}_i, \ldots, \widehat{v}_{i+L_r-1}, \ldots, v_{|s_u|}\right]. \quad (6)$$

3.4 User Representation Model

In this paper, we model the user historical sequences by stacking Transformer encoder, which has achieved promising results in sequential recommendation task [25]. The Transformer encoder consists of three sub-layers, an embedding Layer, a multi-head self-attention module and a position-wise feed-forward Network.

3.4.1 The Embedding Layer. The Transformer encoder utilizes an item embedding matrix $E \in \mathbb{R}^{|\mathcal{V}| \times d}$ to project high dimensional one-hot item representations to low dimensional dense vectors. In addition, to represent the position information of sequence, it leverages the learnable position embedding $P \in \mathbb{R}^{T \times d}$ to capture this feature of sequences. Because the number of position vectors

T restricts the maximum length of the user historical sequence, we truncate the input sequence s_u to the last T items if $|s_u| > T$:

$$s_u = \left[v_{|s_u|-T+1}^{(u)}, v_{|s_u|-T+2}^{(u)}, \ldots, v_{|s_u|}^{(u)}\right]. \quad (7)$$

Finally, we can obtain the input representations of items in the user sequence by adding the item embedding and position embedding together as:

$$h_i^0 = v_i + p_t, \quad (8)$$

where $v_i \in E$ is the representation of item v_i in the user sequence s_u. Here, we omit the superscript u for convenience.

3.4.2 Multi-Head Self-Attention Module. After the embedding layer, the Transformer encoder introduces the self-attention module [43] to capture the dependencies between each item pair in the sequence, which is effective in sequence modeling in many tasks. Moreover, to extract the information from different subspaces at each position, here we adopt the multi-head self-attention instead of a single attention function. It first utilizes different linear projections to project the input representations into h subspaces. Then it applies self-attention mechanism to each head, and derives the output representations by concatenating the intermediates and projecting it once more. The computation is as follows:

$$\text{MH}(H^l) = \text{concat}(\text{head}_1; \text{head}_2; \cdots; \text{head}_h)W^O$$
$$\text{head}_i = \text{Attention}(H^l W_i^Q, H^l W_i^K, H^l W_i^V), \quad (9)$$

where $W_i^Q, W_i^K, W_i^V \in \mathbb{R}^{d \times \frac{d}{h}}$ and $W^O \in \mathbb{R}^{d \times d}$. Here, H^l is the input for the $(l+1)$-th layer. The Attention operation in this equation is the scaled dot-product attention, which is implemented as follows:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d/h}}\right)V, \quad (10)$$

where Q, K, and V represent the queries, the keys, and the values, respectively. The factor $\sqrt{d/h}$ in this attention module is the scale factor to avoid large values of the inner product.

In sequential recommendation, we can only utilize the information before the time step t when we predict the next item v_{t+1}. Therefore, we apply the mask operations to the attention mechanism to discard the connections between Q_i and K_j where $j > i$. These operations can avoid information leakage and are beneficial to the model training.

Xu Xie[1], Fei Sun[2], Zhaoyang Liu[2], Jinyang Gao[2], Bolin Ding[2], Bin Cui[1]

3.4.3 Position-wise Feed-Forward Network. Though multi-head self-attention is beneficial to extract useful information from previous items, it is based on simple linear projections. We endow the model with nonlinearity with a position-wise feed-forward network. It is applied at each position of the above sub-layer's output with sharing learnable parameters:

$$\text{PFFN}(\boldsymbol{H}^l) = [\text{FFN}(\boldsymbol{h}_1^l)^\top; \text{FFN}(\boldsymbol{h}_2^l)^\top; \cdots; \text{FFN}(\boldsymbol{h}_{|s_u|}^l)^\top]$$
$$\text{FFN}(\boldsymbol{h}_i^l) = \text{RELU}(\boldsymbol{h}_i^l \boldsymbol{W}_1 + \boldsymbol{b}_1)\boldsymbol{W}_2 + \boldsymbol{b}_2. \tag{11}$$

3.4.4 Stacking More Blocks. Stacking more blocks is usually beneficial to learn more complex patterns for deep learning methods. However, with more parameters and a deeper network, the model becomes harder to converge. To alleviate this problem, We employ several mechanisms, including the residual connection, layer normalization, and dropout module as follows:

$$\text{LayerNorm}(\boldsymbol{x} + \text{Dropout}(\text{sublayer}(\boldsymbol{x}))), \tag{12}$$

where sublayer$(\cdot)$ is the above multi-head self-attention operation and position-wise feed-forward network. These mechanisms are widely adopted to stabilize and accelerate model training.

3.4.5 User Representations. Based on several Transformer blocks, we obtain the user representation at each time step t, which extracts useful information from the items interacted with before t. Since our task is to predict the item at the time step $|s_u| + 1$ for each user u, we set the final representation for user u as her preference vector at time $|s_u| + 1$:

$$\boldsymbol{s}_u = [\text{Trm}^L(s_u)]_{|s_u|}^\top, \tag{13}$$

where L is the number of stacking Transformer layers. And the Trm function is the composition of following operations:

$$\text{Trm}(\boldsymbol{H}^l) = \text{LayerNorm}(\boldsymbol{F}^{l-1} + \text{Dropout}(\text{PFFN}(\boldsymbol{F}^{l-1})))$$
$$\boldsymbol{F}^{l-1} = \text{LayerNorm}(\boldsymbol{H}^{l-1} + \text{Dropout}(\text{MH}(\boldsymbol{H}^{l-1}))). \tag{14}$$

3.5 Fine-tuning Stage

After the pre-training stage, we obtain the pre-trained user representation model which has been optimized with self-supervised signals derived directly from the unlabeled raw data. In the fine-tuning stage, we initialize the parameters of the user representation model with pre-trained parameters and then utilize traditional sequential supervised signals to fine-tune the model. We adopt the binary cross entropy loss for each user u at each time step t as:

$$\mathcal{L}_{\text{tune}} = -(\log \sigma(\boldsymbol{s}_u^\top \boldsymbol{v}_t^+) + \log(1 - \sigma(\boldsymbol{s}_u^\top \boldsymbol{v}_t^-))), \tag{15}$$

where $\boldsymbol{s}_u$, $\boldsymbol{v}_t^+$, and $\boldsymbol{v}_t^-$ indicate the inferred user representation, the item which user u interacts, and the randomly sampled negative item at the time step t, respectively.

4 EXPERIMENTS

In this section, we conduct plenty of experiments to answer the following research questions:

RQ1. How does the proposed CP4Rec framework perform compared to the state-of-the-art baselines in the sequential recommendation task?

Table 1: Dataset statistics (after preprocessing).

Dataset	#users	#items	#actions	avg.length	density
Beauty	22,363	12,101	198,502	8.8	0.07%
Sports	25,598	18,357	296,337	8.3	0.05%
Toys	19,412	11,924	167,597	8.6	0.07%
Yelp	30,431	20,033	316,354	10.4	0.05%

RQ2. How do different augmentation methods impact the performance? What is the influence of different augmentation hyper-parameters on CP4SRec performance?

RQ3. Can the composition of different augmentation methods lead to a better user encoder?

RQ4. What is the impact of the amount of training data on the performance of CP4Rec when encountering with the data sparsity problem?

4.1 Experiments Settings

4.1.1 Datasets. We conduct experiments on four public datasets collected from the real-world platforms. Three of them are obtained from Amazon, one of the largest e-commercial platforms in the world. These datasets have been introduced in [34], which are split by top-level product categories in amazon. In this work, we follow the settings in [55] and adopt three categories, "Beauty", "Sports and Outdoors", and "Toys and Games". Another dataset is collected by Yelp [1], which is a famous business recommendation platform for restaurants, bars, beauty salons, and so on. We follow the settings in [55] and use the transaction records after January 1st, 2019.

For dataset preprocessing, we follow the common practice in [25, 55]. We convert all numeric ratings or presence of a review to "1" and others to "0". Then, for each user, we sort their historical items by the interacted timestamp chronologically to obtain the user interacted sequence. It is worth mentioning that to guarantee each user/item with enough interactions, we follow the preprocessing procedure in [40, 55], which only keeps the "5-core" datasets. We discard users and items with fewer than 5 interaction records iteratively. The processed data statistics are summarized in Table 1.

4.1.2 Evaluation. We adopt the leave-one-out strategy to evaluate the performance of each method, which is widely employed in many related works [19, 25, 55]. For each user, we hold out the last interacted item as the test data and utilize the item just before the last as the validation data. The remaining items are used for training. To speed up the computation of metrics, many previous works use sampled metrics and only rank the relevant items with a smaller set of random items. However, this sample operation may lead to inconsistent with their non-sampled version as illustrated by Krichene and Rendle [28]. Therefore, we evaluate each method on the whole item set without sampling and rank all the items that the user has not interacted with by their similarity scores. We employ Hit Ratio (HR) and Normalized Discounted Cumulative Gain (NDCG) to evaluate the performance of each method which are widely used in related works [25, 40]. HR focuses on the presence of the positive item, while NDCG further takes the rank position

---

[1]https://www.yelp.com/dataset

information into account. In this work, we report HR and NDCG with $k = 5, 10, 20$.

4.1.3 Baselines. To verify the effectiveness of our method, we compare it with the following representative baselines:

- **Pop**. It is a non-personalized approach which recommends the same items for each user. These items are the most popular items which have the largest number of interactions in the whole item set.
- **BPR-MF** [39]. It is one of the representative non-sequential baselines. It utilizes matrix factorization to model users and items with the pairwise Bayesian Personalized Ranking (BPR) loss.
- **NCF** [19].It employs a neural network architecture to model non-sequential user-item interactions instead of the inner product used by matrix factorization.
- **GRU4Rec** [20, 21]. It applies GRU modules to model user sequences for session-based recommendation with ranking loss and is improved with a new class of loss functions and sampling strategy.
- **SASRec** [25]. It is one of the state-of-the-art baselines to solve the sequential recommendation task. It models user sequences through self-attention modules to capture users' dynamic interests.
- **SASRec**$_{BPR}$. To compare the performance with other pre-training strategies, we utilize the item embedding learned by the BPR-MF method to initialize the parameters of SASRec model.

4.1.4 Implementation Details. We implement BPR-MF, NCF, and GRU4Rec with public codes provided by Wang et al. [44] by PyTorch. For other methods, we implement them by Tensorflow. For all models with learnable parameters, we set the embedding dimension size $d = 128$. All other hyper-parameters are set following the suggestions from their original settings mentioned in published papers. We report each baseline performance under its optimal settings.

When it comes to our CP4Rec method, different from the above baselines, it contains two training stages, including the pre-training stage and fine-tuning stage. We initialize all parameters by the truncated normal distribution in the range $[-0.01, 0.01]$ in the pre-training stage. And then we use the learned parameters in this stage to initialize the user representation model and item embedding layer in the fine-tuning stage. We use Adam optimizer [26] to optimize both of the two stages with the learning rate of 0.001, $\beta_1 = 0.9$, $\beta_2 = 0.999$, and linear decay of the learning rate. The batch size is set to 256 in both stages. We pre-train and fine-tune the model with early stopping techniques.

In the pre-training stage, to investigate the effect of each two augmentation method combination, we fixed the augmentation methods used in CP4Rec each time. And we test the crop/mask/reorder proportion of items from $[0.1, 0.3, 0.5, 0.7, 0.9]$. For our user representation model of CL4SRec, we stack 2 self-attention blocks together and set the head number as 2 for each block. For a fair comparison, the maximum sequence length of T is set to 50 when we evaluate each approach, following the settings in [55].

4.2 Overall Performance Comparison (RQ1)

To answer **RQ1**, we compare the performance of our CP4Rec with the above baselines. Table 2 summarizes the best results of all models on four datasets. Note that the improvement columns are the performance of CP4Rec relative to its user representation model SASRec and the pre-training version SASRec$_{BPR}$.

Based on the experiment results, we can observe that:

- The non-personalized method Pop exhibits the worst performance on all datasets since it ignores users' unique preferences hidden in their historical interactions. Considering other baseline methods on all datasets, we observe that sequential methods (e.g., GRU4Rec and SASRec) outperform non-sequential methods (e.g., BPR-MF and NCF) consistently. Compared with those non-sequential methods, these sequential methods utilize sequential information of users' historical interactions, which contributes to improving performance in recommendation systems. Among all the baselines, SASRec achieves state-of-the-art performance on all datasets, which indicates that the powerful self-attention mechanism is suitable for capturing sequence patterns.
- When it comes to the SASRec$_{BPR}$ which utilizes BPR-MF to pre-train SASRec, it does not achieve obvious improvements and even sometimes exhibits worse compared to the SASRec. But it converges more quickly at the fine-tuning step than SASRec, since the pre-training step can warm-up the following procedure. One possible reason is that it utilizes BPR-MF model to fit the same supervised information used in the following step and does not capture auxiliary useful information. Therefore, with the powerful self-attention mechanism in the fine-tuning step, SASRec$_{BPR}$ exhibits similarly to SASRec when they are converged.
- Finally, according to the results, it is obvious that our proposed CL4SRec outperforms all the baselines on all the datasets in terms of all the evaluation metrics. CP4Rec gains 8.16% HR@10, 4.74% HR@20, 9.76% NDCG@10, and 7.53% on NDCG@20 on average against the start-of-the-art baseline SASRec. These experiments verify the effectiveness of our CP4Rec method in the sequential recommendation task. Different from SASRec$_{BPR}$, we adopt the contrastive learning framework to introduce self-supervised signals, which enhances the user representation model to capture more accurate user representations.

4.3 Comparison of Different Augmentation Methods (RQ2)

To answer **RQ2**, in this section, we analyze how different augmentation operators and their proportion rates impact the performance. To examine the effect of each augmentation operator, we only utilize one kind of operator in the pre-training stage with the same proportion parameters each time. Note that item mask and item reorder operations are strong augmentations with high rate of γ and β, while item crop operation is strong augmentation when η is small. We report HR@10 and NDCG@10 as metrics for the experiments due to the space limitation.

Figure 4 demonstrates the performance of different augmentation methods with different proportion rates η, γ, and β varing from

Xu Xie[1], Fei Sun[2], Zhaoyang Liu[2], Jinyang Gao[2], Bolin Ding[2], Bin Cui[1]

Table 2: Performance comparison of different methods on top-N recommendation. Bold scores are the best in method group. Improvement #1 and #2 are calculated over SASRec methods and SASRec$_{BPR}$, respectively.

Datasets	Metric	Pop	BPR-MF	NCF	GRU4Rec	SASRec	SASRec$_{BPR}$	CL4SRec	Improv.#1	Improv.#2
Beauty	HR@5	0.0080	0.0123	0.0116	0.0263	0.0452	0.0474	**0.0513**	17.47%	8.23%
	HR@10	0.0152	0.0308	0.0259	0.0457	0.0715	0.0747	**0.0784**	9.65%	4.95%
	HR@20	0.0217	0.0549	0.0462	0.0743	0.1097	0.1139	**0.1156**	5.38%	1.49%
	NDCG@5	0.0044	0.0066	0.0063	0.0169	0.0300	0.0308	**0.0339**	13.00%	10.06%
	NDCG@10	0.0068	0.0125	0.0108	0.0231	0.0384	0.0396	**0.0425**	10.68%	7.32%
	NDCG@20	0.0084	0.0186	0.0159	0.0303	0.0479	0.0495	**0.0519**	8.35%	4.85%
Sports	HR@5	0.0056	0.0115	0.0095	0.0193	0.0249	0.0267	**0.0292**	17.27%	9.36%
	HR@10	0.0094	0.0213	0.0163	0.0319	0.0408	0.0424	**0.0442**	8.33%	4.25%
	HR@20	0.0192	0.0357	0.0286	0.0512	0.0637	0.0641	**0.0675**	5.97%	5.30%
	NDCG@5	0.0041	0.0067	0.0057	0.0120	0.0165	0.0175	**0.0189**	14.55%	8.00%
	NDCG@10	0.0053	0.0098	0.0079	0.0161	0.0216	0.0226	**0.0238**	10.19%	5.30%
	NDCG@20	0.0078	0.0134	0.0109	0.0209	0.0273	0.0280	**0.0296**	8.42%	5.71%
Toys	HR@5	0.0066	0.0154	0.0132	0.0259	0.0555	0.0575	**0.0627**	11.29%	9.04%
	HR@10	0.0113	0.0287	0.0267	0.0366	0.0841	0.0851	**0.0908**	7.97%	6.70%
	HR@20	0.0163	0.0467	0.0446	0.0525	0.1235	0.1176	**0.1264**	2.35%	7.48%
	NDCG@5	0.0046	0.0078	0.0071	0.0176	0.0382	0.0390	**0.0420**	9.95%	7.69%
	NDCG@10	0.0061	0.0122	0.0114	0.0211	0.0474	0.0479	**0.0516**	8.86%	7.72%
	NDCG@20	0.0073	0.0167	0.0156	0.0251	0.0573	0.0561	**0.0607**	5.93%	8.20%
Yelp	HR@5	0.0057	0.0344	0.0329	0.0217	0.0571	0.0592	**0.0625**	9.46%	5.57%
	HR@10	0.0099	0.0552	0.0496	0.0367	0.0761	0.0779	**0.0812**	6.70%	4.24%
	HR@20	0.0164	0.0858	0.0739	0.0615	0.1025	0.1043	**0.1079**	5.27%	3.45%
	NDCG@5	0.0037	0.0228	0.0216	0.0133	0.0463	0.0486	**0.0510**	10.15%	4.94%
	NDCG@10	0.0051	0.0294	0.0270	0.0182	0.0525	0.0536	**0.0574**	9.33%	7.09%
	NDCG@20	0.0067	0.0371	0.0331	0.0244	0.0594	0.0597	**0.0638**	7.40%	6.87%

0.1 to 0.9. We observe a few trends of different augmentation methods with the variation of proportion rates. First, the performance of CP4Rec equipped with any augmentation method can outperform the SASRec baseline on all datasets for most choices of proportion rates. It indicates the effectiveness of the proposed augmentation methods,CP4Rec, since they all introduce auxiliary self-supervised signals hidden in raw data. And we observe that none of the three augmentation operations can always achieve the best performance compared with other augmentations. For example, Reorder operation achieves the best results on the Beauty dataset, but the Mask operation achieves the best results on the Toys dataset. This demonstrates that different augmentation methods are appropriate for different datasets since they focus on different aspects of the raw data.

Second, we observe how the proportion rates of different augmentation methods affect the recommendation performance. Considering the Item Crop and Item Mask operators, a general pattern is that the performance peaks at a special proportion rate and then deteriorates if we increase or decrease the rate. For example, Item Mask operator peaks at the proportion rate 0.5 on the Beauty dataset. It can be explained that when the proportion rate γ equals to 0, Item Mask operator does not function and when γ equals to 1.0, the whole user sequence only consists of [$mask$] items, thus hurting the performance. Item Crop operator acts similarly to Item Mask operation, except that Item Crop operator does not function

with $\eta = 1.0$ and the user sequence is empty with $\eta = 0$. However, when it comes to Item reorder operator, we observe that a larger reorder proportion rate β leads to a better performance on the Sports/Toys/Yelp dataset. It demonstrates that the sequential patterns in these three datasets are more flexible than the Beauty dataset, since the sequence assumption is less strict with a higher reorder rate.

4.4 Effect of Composition of Different Augmentations (RQ3)

To answer **RQ3**, in this section, we analyze how the composition of different augmentation methods impacts the performance. We apply two different augmentation method a_1 and a_2 to the same user sequence s_u and obtain two views of the sequence, denoting $s_u^{a_1} = a_1(s_u)$ and $s_u^{a_2} = a_2(s_u)$. Therefore, we have three composite augmentation methods of different combinations of basic ones. We examine the influence of these composite augmentation operations with the best proportion rate for each component. We report HR@10 and NDCG@10 as metrics on the Beauty and the Yelp datasets due to the space limitation.

Figure 5 shows the evaluation results. We can observe that the composition of different augmentations does not perform better than anyone of its single component. For example, though Item Mask and Item Reorder operators both achieve the best performance on HR@10, the composition of them does not outperform
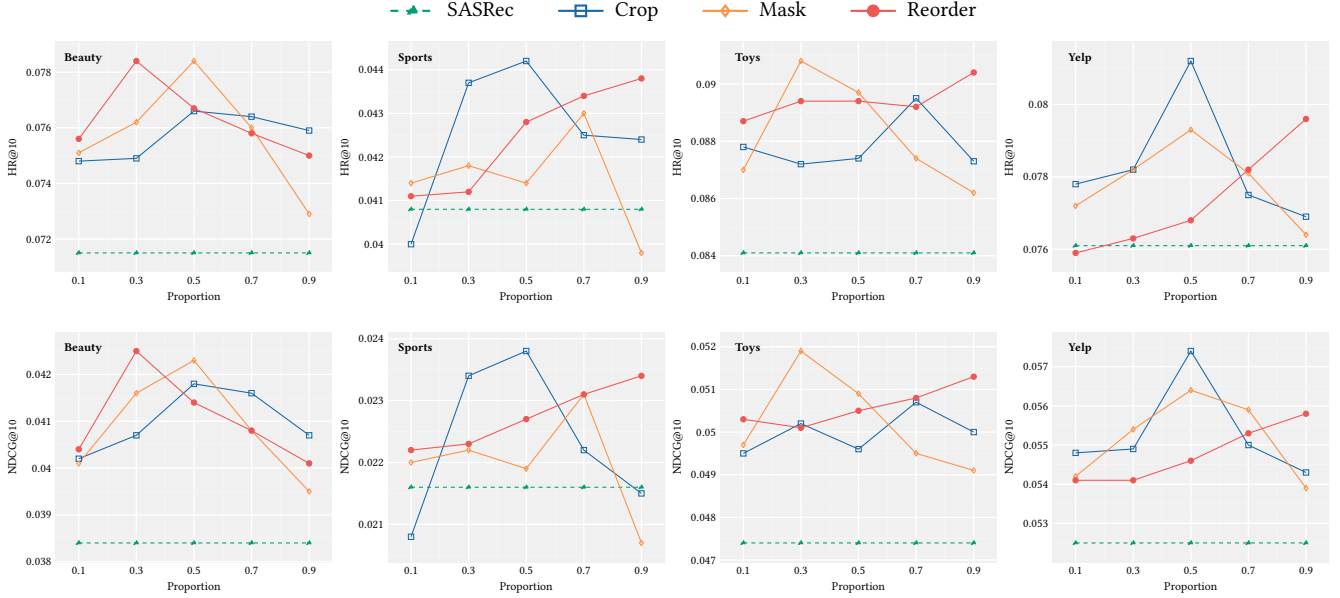
Figure 4: Impact of the different augmentation methods with different proportion η, γ, and β on HR@10 and NDCG@10. The dash line is the performance of the SASRec method.

(a) Hit@10 on Beauty
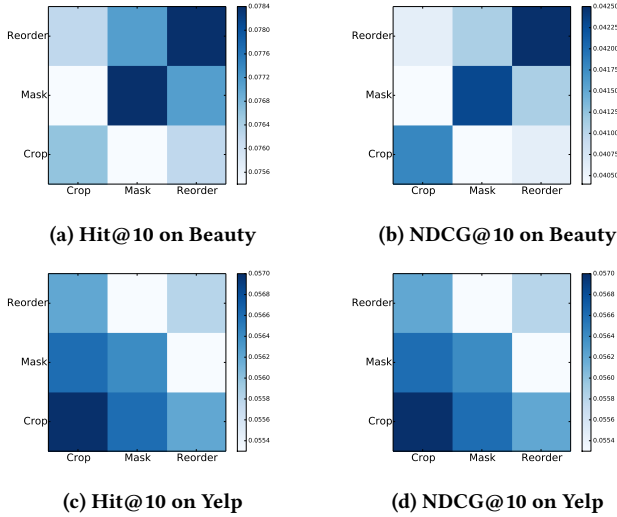
(b) NDCG@10 on Beauty

(c) Hit@10 on Yelp

(d) NDCG@10 on Yelp

Figure 5: Performance (HR@10 and NDCG@10) comparison w.r.t. compositions of different augmentation methods on Beauty and Yelp datasets

Figure 6: Performance comparison on CP4Rec and SASRec w.r.t. the different amounts of training samples.

4.5 Impact of the Amounts of Training Data (RQ4)

To answer **RQ4**, in this section, we investigate whether our proposed CP4Rec can alleviate the data sparsity problem. We simulate the data sparsity scenario by using different proportions of the full dataset, including 100%, 80%, 60%, 40%, and 20%. With this setting, we then evaluate these models and explore how they suffer from the data sparsity problem in a real-world scenario. We utilize the

any of them. But it still achieves better performance than other baselines. This observation implies that homogeneous transformation in CP4Rec framework is better than heterogeneous structure since it may be hard for the user representation model to distinguish whether sequences are transformed from the same data by different augmentation methods.
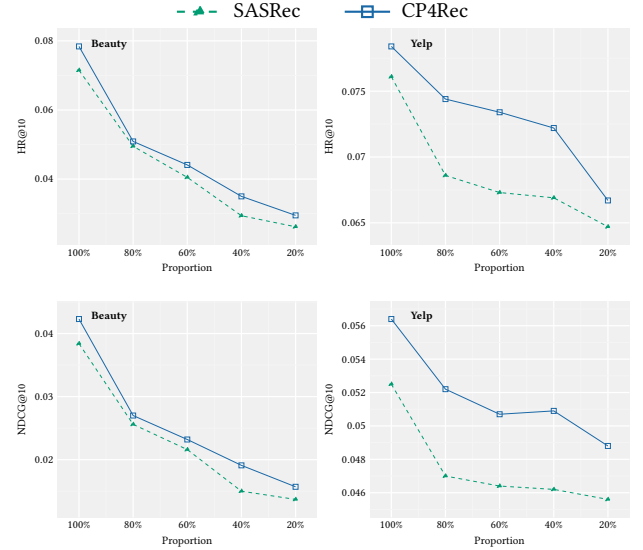
item mask method with $\gamma = 0.5$ for CP4Rec. And we report HR@10 and NDCG@10 as metrics on the Beauty and the Yelp datasets.

Figure 6 shows the evaluation results. We observe a few trends with the variation of amounts of training data. On the one hand, the performance substantially deteriorates when less training data is used. But for different datasets, the impact of data sparsity is different. For example, the performance of SASRec decreases 180.29% and 21.74% on NDCG@10 on Beauty and Yelp dataset with 20% data compared to training with full data, respectively. On the other hand, CP4Rec is consistently better than SASRec in all cases. This observation implies that CP4Rec is able to make better use of the data, which alleviates the influence of data sparsity problem for sequential recommendation to some extent.

5 CONCLUSION AND FUTURE WORK

In this paper, we propose a novel model called Contrastive Pre-training for Sequential Recommendation (CP4Rec), which can learn the effective user representation model only with item interaction information. It is equipped with the contrastive learning framework to extract self-supervised signals just from raw data and utilizes them to pre-train the model. In addition, we propose three data augmentation approaches to construct pre-training tasks and exploit the effects of the composition of different augmentations. The proposed method is verified on four publicly datasets. Extensive experiment results show that our CP4Rec achieves significant improvements and outperforms the start-of-the-art baselines, especially when limited training data is available.

REFERENCES

[1] Mikel Artetxe, Gorka Labaka, Eneko Agirre, and Kyunghyun Cho. 2018. Unsupervised Neural Machine Translation. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings.* OpenReview.net.

[2] Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew M. Dai, Rafal Józefowicz, and Samy Bengio. 2016. Generating Sentences from a Continuous Space. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, CoNLL 2016, Berlin, Germany, August 11-12, 2016.* ACL, 10–21.

[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. (2020). arXiv:2005.14165 [cs.CL]

[4] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. 2020. A Simple Framework for Contrastive Learning of Visual Representations. *CoRR* abs/2002.05709 (2020). https://arxiv.org/abs/2002.05709

[5] Xu Chen, Hongzhi Yin, Hongxu Chen, Rui Yan, Quoc Viet Hung Nguyen, and Xue Li. 2019. AIR: Attentional Intention-Aware Recommender Systems. In *35th IEEE International Conference on Data Engineering, ICDE 2019, Macao, China, April 8-11, 2019.* IEEE, 304–315.

[6] Xu Chen, Hongteng Xu, Yongfeng Zhang, Jiaxi Tang, Yixin Cao, Zheng Qin, and Hongyuan Zha. 2018. Sequential Recommendation with User Memory Networks. 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.* ACM, 108–116.

[7] Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the Properties of Neural Machine Translation: Encoder-Decoder Approaches. In *Proceedings of SSST@EMNLP 2014, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation, Doha, Qatar, 25 October 2014.* Association for Computational Linguistics, 103–111.

[8] Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language*

Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL.* ACL, 1724–1734.

[9] Andrew M. Dai and Quoc V. Le. 2015. Semi-supervised Sequence Learning. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada.* 3079–3087.

[10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers).* Association for Computational Linguistics, 4171–4186.

[11] Carl Doersch, Abhinav Gupta, and Alexei A. Efros. 2015. Unsupervised Visual Representation Learning by Context Prediction. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015.* IEEE Computer Society, 1422–1430.

[12] Tim Donkers, Benedikt Loepp, and Jürgen Ziegler. 2017. Sequential User-based Recurrent Neural Network Recommendations. In *Proceedings of the Eleventh ACM Conference on Recommender Systems, RecSys 2017, Como, Italy, August 27-31, 2017,* Paolo Cremonesi, Francesco Ricci, Shlomo Berkovsky, and Alexander Tuzhilin (Eds.). ACM, 152–160.

[13] Yifan Feng, Haoxuan You, Zizhao Zhang, Rongrong Ji, and Yue Gao. 2019. Hypergraph Neural Networks. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019.* AAAI Press, 3558–3565.

[14] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. 2018. Unsupervised Representation Learning by Predicting Image Rotations. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings.* OpenReview.net.

[15] Xueliang Guo, Chongyang Shi, and Chuanming Liu. 2020. Intention Modeling from Ordered and Unordered Facets for Sequential Recommendation. In *WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020.* ACM / IW3C2, 1127–1137.

[16] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross B. Girshick. 2020. Momentum Contrast for Unsupervised Visual Representation Learning. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020.* IEEE, 9726–9735.

[17] Ruining He, Chen Fang, Zhaowen Wang, and Julian J. McAuley. 2016. Vista: A Visually, Socially, and Temporally-aware Model for Artistic Recommendation. In *Proceedings of the 10th ACM Conference on Recommender Systems, Boston, MA, USA, September 15-19, 2016.* ACM, 309–316.

[18] Ruining He and Julian J. McAuley. 2016. Fusing Similarity Models with Markov Chains for Sparse Sequential Recommendation. In *IEEE 16th International Conference on Data Mining, ICDM 2016, December 12-15, 2016, Barcelona, Spain.* IEEE Computer Society, 191–200.

[19] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of WWW.* 173–182.

[20] Balázs Hidasi and Alexandros Karatzoglou. 2018. Recurrent Neural Networks with Top-k Gains for Session-based Recommendations. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management, CIKM 2018, Torino, Italy, October 22-26, 2018.* ACM, 843–852.

[21] 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.).

[22] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Comput.* 9, 8 (1997), 1735–1780.

[23] Liang Hu, Longbing Cao, Shoujin Wang, Guandong Xu, Jian Cao, and Zhiping Gu. 2017. Diversifying Personalized Recommendation with User-session Context. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017,* Carles Sierra (Ed.). ijcai.org, 1858–1864. https://doi.org/10.24963/ijcai.2017/258

[24] 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.* ACM, 505–514.

[25] Wang-Cheng Kang and Julian J. McAuley. 2018. Self-Attentive Sequential Recommendation. In *IEEE International Conference on Data Mining, ICDM 2018, Singapore, November 17-20, 2018.* IEEE Computer Society, 197–206.

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

[27] Alexander Kolesnikov, Xiaohua Zhai, and Lucas Beyer. 2019. Revisiting Self-Supervised Visual Representation Learning. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019.*

Computer Vision Foundation / IEEE, 1920–1929.

[28] Walid Krichene and Steffen Rendle. 2020. On Sampled Metrics for Item Recommendation. In *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*. ACM, 1748–1757.

[29] Ankit Kumar, Ozan Irsoy, Peter Ondruska, Mohit Iyyer, James Bradbury, Ishaan Gulrajani, Victor Zhong, Romain Paulus, and Richard Socher. 2016. Ask Me Anything: Dynamic Memory Networks for Natural Language Processing. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016 (JMLR Workshop and Conference Proceedings, Vol. 48)*. JMLR.org, 1378–1387.

[30] Guillaume Lample, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2018. Unsupervised Machine Translation Using Monolingual Corpora Only. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

[31] Jing Li, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tao Lian, and Jun Ma. 2017. Neural Attentive Session-based Recommendation. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, CIKM 2017, Singapore, November 06 - 10, 2017*. ACM, 1419–1428.

[32] Fuyu Lv, Taiwei Jin, Changlong Yu, Fei Sun, Quan Lin, Keping Yang, and Wilfred Ng. 2019. SDM: Sequential Deep Matching Model for Online Large-scale Recommender System. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3-7, 2019*. ACM, 2635–2643.

[33] Chen Ma, Peng Kang, and Xue Liu. 2019. Hierarchical Gating Networks for Sequential Recommendation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019*. ACM, 825–833.

[34] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. 2015. Image-based recommendations on styles and substitutes. In *Proceedings of SIGIR*. ACM, 43–52.

[35] Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. 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*. 3111–3119.

[36] Mehdi Noroozi and Paolo Favaro. 2016. Unsupervised Learning of Visual Representations by Solving Jigsaw Puzzles. In *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part VI (Lecture Notes in Computer Science, Vol. 9910)*. Springer, 69–84.

[37] Massimo Quadrana, Alexandros Karatzoglou, Balázs Hidasi, and Paolo Cremonesi. 2017. Personalizing Session-based Recommendations with Hierarchical Recurrent Neural Networks. In *Proceedings of the Eleventh ACM Conference on Recommender Systems, RecSys 2017, Como, Italy, August 27-31, 2017*. ACM, 130–137.

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

[39] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *Proceedings of UAI* (Montreal, Quebec, Canada). 452–461.

[40] 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.

[41] 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*. ACM, 1441–1450.

[42] 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*. ACM, 565–573.

[43] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*. 5998–6008.

[44] Chenyang Wang, Min Zhang, Weizhi Ma, Yiqun Liu, and Shaoping Ma. 2020. Make It a Chorus: Knowledge- and Time-aware Item Modeling for Sequential Recommendation. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*. ACM, 109–118.

[45] Jianling Wang, Kaize Ding, Liangjie Hong, Huan Liu, and James Caverlee. 2020. Next-item Recommendation with Sequential Hypergraphs. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*. ACM, 1101–1110.

[46] Shoujin Wang, Liang Hu, Yan Wang, Longbing Cao, Quan Z. Sheng, and Mehmet A. Orgun. 2019. Sequential Recommender Systems: Challenges, Progress and Prospects. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*. ijcai.org, 6332–6338.

[47] Zhirong Wu, Yuanjun Xiong, Stella X. Yu, and Dahua Lin. 2018. Unsupervised Feature Learning via Non-Parametric Instance Discrimination. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*. IEEE Computer Society, 3733–3742.

[48] Xin Xin, Alexandros Karatzoglou, Ioannis Arapakis, and Joemon M. Jose. 2020. Self-Supervised Reinforcement Learning for Recommender Systems. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*. ACM, 931–940.

[49] An Yan, Shuo Cheng, Wang-Cheng Kang, Mengting Wan, and Julian J. McAuley. 2019. CosRec: 2D Convolutional Neural Networks for Sequential Recommendation. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3-7, 2019*. ACM, 2173–2176.

[50] Tiansheng Yao, Xinyang Yi, Derek Zhiyuan Cheng, Felix X. Yu, Aditya Krishna Menon, Lichan Hong, Ed H. Chi, Steve Tjoa, Jieqi Kang, and Evan Ettinger. 2020. Self-supervised Learning for Deep Models in Recommendations. *CoRR* abs/2007.12865 (2020). https://arxiv.org/abs/2007.12865

[51] Mang Ye, Xu Zhang, Pong C. Yuen, and Shih-Fu Chang. 2019. Unsupervised Embedding Learning via Invariant and Spreading Instance Feature. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*. Computer Vision Foundation / IEEE, 6210–6219.

[52] Feng Yu, Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. 2016. A Dynamic Recurrent Model for Next Basket Recommendation. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval, SIGIR 2016, Pisa, Italy, July 17-21, 2016*. ACM, 729–732.

[53] Fajie Yuan, Xiangnan He, Haochuan Jiang, Guibing Guo, Jian Xiong, Zhezhao Xu, and Yilin Xiong. 2020. Future Data Helps Training: Modeling Future Contexts for Session-based Recommendation. In *WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020*. ACM / IW3C2, 303–313.

[54] Richard Zhang, Phillip Isola, and Alexei A. Efros. 2016. Colorful Image Colorization. In *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part III (Lecture Notes in Computer Science, Vol. 9907)*. Springer, 649–666.

[55] Kun Zhou, Hui Wang, Wayne Xin Zhao, Yutao Zhu, Sirui Wang, Fuzheng Zhang, Zhongyuan Wang, and Ji-Rong Wen. 2020. S^3-Rec: Self-Supervised Learning for Sequential Recommendation with Mutual Information Maximization. *CoRR* abs/2008.07873 (2020).