Learning Transferable User Representations with Sequential Behaviors via Contrastive Pre-training

Mingyue Cheng[†], Fajie Yuan[§], Qi Liu[†*], Xin Xin[‡], Enhong Chen[†]
[†]Anhui Province Key Laboratory of Big Data Analysis and Application, School of Data Science,
University of Science and Technology of China, mycheng@mail.ustc.edu.cn, {qiliuql, cheneh}@ustc.edu.cn
[§]Westlake University, yuanfajie@westlake.edu.cn
[‡]Shandong University, xinxin@sdu.edu.cn

***Abstract*—Learning effective user representations from sequential user-item interactions is a fundamental problem for recommender systems (RS). Recently, several unsupervised methods focusing on user representations pre-training have been explored. In general, these methods apply similar learning paradigms by first corrupting the behavior sequence, and then restoring the original input with some item-level prediction loss functions. Despite its effectiveness, we argue that there exist important gaps between such item-level optimization objective and user-level representations, and as a result, the learned user representations may only lead to sub-optimal generalization performance. In this paper, we propose a novel self-supervised pre-training framework, called CLUE, which stands for employing Contrastive Learning for modeling sequence-level User rEpresentation. The core idea of CLUE is to regard each user behavior sequence as a whole and then construct the self-supervision signals by transforming the original user behaviors by data augmentations (DA). Specifically, we employ two Siamese (weight-sharing) networks to learn the user-oriented representations, where the optimization goal is to maximize the similarity of learned representations of the same user by these two encoders. More importantly, we perform careful investigation of the impacts of view generating strategies for user behavior inputs from a more comprehensive perspective, including processing sequential behaviors by explicit DA strategies and employing dropout as implicit DA. To verify the effectiveness of CLUE, we perform extensive experiments on several user-related tasks with different scales and characteristics. Our experimental results show that the user representations learned by CLUE surpass existing item-level baselines under several evaluation protocols.**

***Index Terms*—Recommender systems; User representation; Contrastive learning; Sequential behaviors**

I. Introduction

With the rapid growth of the Internet, online information grows explosively. When making decisions in daily life, people are often exposed to huge amounts of information or data, the problem is often referred to as Information Overload. Fortunately, recommender systems (RS), as an important information filtering technique, have been extensively studied in the past two decades, which help users find personalized items that fit their needs [1], [2]. The basic idea of RS is built following the simple assumption that if users rate items similarly in the past, they are likely to rate other items similarly in the future.

To capture such similarity between users, one common paradigm of RS is to project users as embedded feature vectors, a.k.a., representations. Actually, learning accurate representations [3] has become a fundamental problem in personalized RS. Earlier works like *matrix factorization* (MF) [4] parameterize the IDs of a user and item by formulating item recommendation task as a matrix completion problem. Recently, significant progress has been made in user representations learning with deep neural network techniques [5]. For example, [6] proposed to leverage Auto-Encoder framework to learn latent representations by first corrupting user-item records and then reconstructing the full input. Meanwhile, some lines of research employ DNNs to learn representations of users and items based on the two-tower structure [3], [7]. Particularly, a series of deep sequential recommender models [8]–[10] have been applied to encode user interaction sequences, which yield improved results than the common DNN-based methods. Nevertheless, these standard models usually fail to learn optimal representations when encountering the data sparsity problems (e.g., the cold-start issues) with limited context information.

Inspired by the breakthrough performance of pre-training in NLP [11], [12] and CV [13]–[15], several recent works [16] have explored such techniques for recommendation tasks by first pre-training generic user representations and then transferring them to downstream tasks. The main advantages of these pre-training approaches can be roughly summarized into two aspects. On the one hand, the pre-trained model can capture general-purpose user preference via exploiting historical user-item interaction data. For example, PeterRec [16] randomly masks a certain percentage of items and then predicts them at these masked positions. It is the first work clearly (by experiments) demonstrating that the learned user representations could be transferred to improve multiple user-related tasks, including but not limited to profile predictions and cold-user based recommendations [17]. On the other hand, the pre-trained network can obtain knowledge from some auxiliary tasks when learning user representations, which can be used to enhance other tasks. One related work is S^3-Rec [18] which first corrupts the original inputs and then restores them by adopting mutual information as the optimization principle.

While these pre-training methods are proven to benefit a diversity of user-related downstream tasks, we find that the common learning paradigms of these methods are usually based on the *item-level* optimization objective: first corrupting

Fajie Yuan is joint first author, and Qi Liu is corresponding author.

the full user behavior sequence by masking and then restoring these masked tokens. Correspondingly, such methods actually do not *explicitly* perform holistic user sequence learning, i.e., they largely emphasize the item representation but lack of a careful design on complete user representation. Hence, we argue that such learning schemes might not be appropriate for encoding effective user representations since there exists a big gap between the item-level optimization objective and sequence-level user representation learning. In addition, previous works show that [4], [19], [20] it is more reasonable to model user representations by distinguishing the preference difference between users rather than guessing the absolute preference level on candidate items.

Based on the above analysis, we delve research into learning general-purpose user representations, which can then be well transferred to various downstream tasks. To do so, we propose a novel self-supervised pre-training framework, namely **CLUE**, based on applying **C**ontrastive **L**earning to train generic **U**ser r**E**presentations from ordered historical behaviors. The core idea of CLUE is to regard all historical interaction records as a whole and construct the self-supervision signals by transforming views of the original user behaviors and perform sequence-level comparison optimization. Specifically, we employ two Siamese (weighting-sharing) networks to learn the user-oriented representations, in which the optimization objective is to maximize the similarity of learned representations by the two Siamese (weighting-sharing) encoders. Meanwhile, we carefully explore the view transforming strategies to obtain multiple views of the same user behavior sequences, including the commonly used data augmentation (DA) strategy (including cropping, masking, permutation) with explicitly processing sequence behavior and a strikingly simple yet effective dropout strategy.

The contributions are summarized as follows:

- We present CLUE, a general framework for learning user representations with sequence-level contrastive learning. This is greatly different from previous methods (e.g., PeterRec [16] and Conure [21]) that model user interaction sequences via the item-level prediction loss. To our best knowledge, CLUE is also the first work that uses contrastive pre-training to study the transfer learning of user representations for downstream tasks.
- We thoroughly investigate the view generating strategies for CLUE. By comparing with the most popular DA methods, we surprisingly find that the simple dropout operation [22] that has been widely adopted by many deep neural networks (DNN) could be always sufficient and performs better for creating good views in the recommendation field. This insightful finding largely simplifies the procedures of CLUE since many explicit data augmentation [15] strategies can be omitted.
- To verify the effectiveness of CLUE, we perform extensive experiments on a diversity of user-related tasks with different scales and characteristics, including the cold-start recommendations, user profiling, and sequential recommendations. We also perform valuable visual explanations for deeply understanding the learned user embedding. Our experimental results demonstrate that the user representations learned by CLUE are more effective and transferable than existing top-performing baselines.

II. Related work

In this section, we will review related works closely to this work from two aspects, including user representation learning and self-supervised learning.

A. User Representation Learning

Learning optimal user representations is a fundamental research problem in recommendation area. With effective user representations, systems can not only capture user interest for delivering a personalized ranking over item set for each user but also be helpful for user modeling, which is a conceptual understanding of the user (e.g., user profile prediction [23], [24]). Earlier methods like matrix factorization (MF) project the userID into latent vectors by conducting matrix competition task [20], [25]. Later on, some deep recommendation approaches [16], [26] are developed to improve recommendation performances by the powerful representation abilities of deep learning. Representative methods in [27] propose to leverage Auto-Encoder to learn latent representations of corrupted user-item preferences and recommend new items to the users given the existing preference set as input at prediction stage. Another prevalent line of deep user representation is to employ the two-tower architecture to learning high-quality user representations [3]. Recently, a large body of research [10], [28] show that sequential recommendation models achieve significant performance improvements in modeling user preference by exploiting historical sequence actions. Among them, many researchers have paid special attention to three lines of work: RNN-based [8], CNN-based [10], [28], and attention-based sequential models [29].

B. Self-supervised Learning

Recently, self-supervised learning [30]–[32] have become prevalent since the parameters optimized by the self-supervised loss can be easily utilized to benefit other tasks. Most mainstream approaches fall into one of two classes: generation or discrimination. For generative-based methods, in NLP area, the language model is a popular self-supervised objective that learns to predict the next word given the previous words [33]. Also, the cloze and next sentence prediction task are widely adopted in [12]. Though these fine-grained prediction tasks have achieved promising results, some researchers hold that its computation is expensive and might not be necessary for representation learning [15]. Recently, discrimination-based methods, such as contrastive learning, have been well studied in visual representation learning. The core idea of such methods is to pull the positive example pairs closer and push the negative instance pairs apart. Some recent works e.g., [15] demonstrate that such a learning paradigm could nearly achieve competitive results compared with supervised learning. Simple and effective instantiations

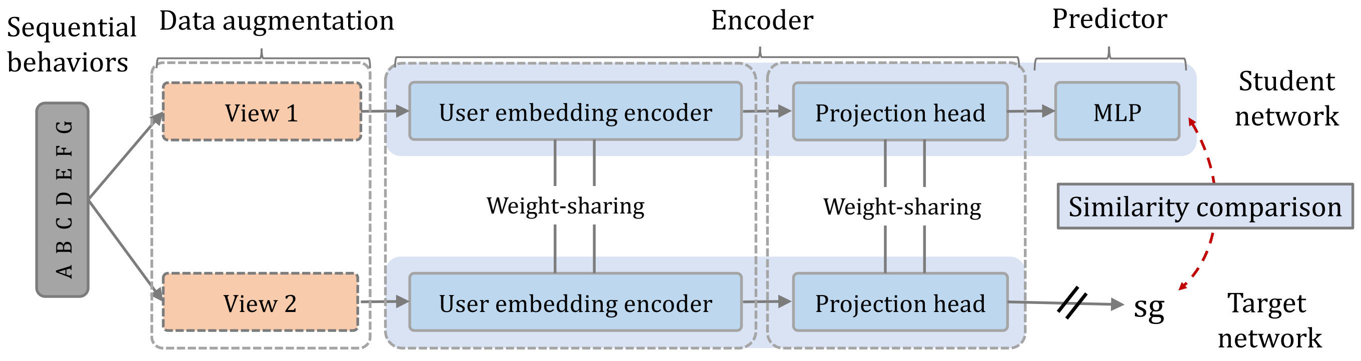

Fig. 1. Illustration of the contrastive pre-training network in CLUE, where SG denotes stop-gradient operator.

of contrastive learning have been developed using Siamese network [34], [35]. Most of these contrastive learning methods benefit from a large number of negative examples [36], [37]. Also, recent work in [38], [39] proposed to directly predict the output of one view from another view without using negative instance comparison.

When it comes to recommendation area, existing pre-training methods consistently adopt item-level optimization objectives. To the best of our knowledge, it remains under-explored how to present unsupervised sequence-level user representations and whether such representations are transferable across different tasks.

III. Problem and Framework Overview

In this section, we first formalize the user representation pre-training problem and introduce the overview of CLUE framework.

A. Problem Statement

In this paper, we represent column vectors and matrices by bold italic lower case letters and bold upper case letters, respectively. Suppose we have two tasks: a pre-training and a downstream task. Let $\mathcal{U}$ and $\mathcal{I}$ respectively denote the set of user and item set, where $|\mathcal{U}|$ and $|\mathcal{I}|$ indicate the number of users and items. We represent the user behavior sequence for user u with $x^u = \{x_1^u, x_2^u, ..., x_n^u\}(x_i^u \in \mathcal{X})$, where $\mathcal{X}$ denotes the item set in pre-training stage. Here x_t^u denotes the t-th interacted item of u and n denotes the length of interaction sequence for user u. If the sequence length is more than n, we consider the most recent n actions. Each instance in the downstream task consists of a userID u and its label. If u has g different labels, then there will be g instances for u in the downstream task.

B. Framework Overview

The training procedure of CLUE is composed of two standard transfer learning stages. The first stage is to learn a high-capacity user representation model directly from massive user sequential records. Then, there is a supervised fine-tuning stage, where the pre-trained representation is adapted to the downstream task with the ground truth of new tasks. Figure 1 illustrates the proposed self-supervised pre-training framework. In popular contrastive learning schemes, semantically close neighbors usually are pulled close together and non-neighbors are pushed apart. However, it is hard to distinguish whether two users are semantically close neighbors or not. Hence, we discard the widely used negative sampling method [15], [40] but only retain positive examples as training signals by following recently proposed methods [38]. In the pre-training network, for user u, two randomly augmented views of the same sequence behaviors including $\bar{x}^u$ and $\tilde{x}^u$ are regarded as input. Then, we employ two Siamese (weighting) networks, referred to as *student* and *target* network, for transforming the user sequential behaviors into latent representations, by following the commonly used schemes [39]. During training, the Siamese network projects the input sequence into latent representations and then tries to discriminate the matching results from one another.

Specifically, in CLUE framework, we need to deal with two main challenges: (1) how to alleviate the model collapse issue caused by the discarding of negative instance (Section IV-A); (2) how to construct positive pairs as self-supervised training signals via effective data augmentation (DA) strategies (Section IV-B).

IV. CLUE

In this section, we firstly present the general pre-trained network of CLUE, which aims at characterizing the user similarity with effective user representations learned from sequence-level comparison. Then, we introduce how to obtain view transformations from the same user's historical behaviors arranged in chronological order. After that, we introduce the network transferring process, i.e., transferring the pre-trained user representation to target user-oriented tasks.

A. Contrastive Pre-training Network

In the following, we would explain the pre-training network on the basis of the self-attentive architecture in SASRec [41] since such self-attentive architecture has shown state-of-the-art results in modeling user dynamic preference [18], [29], [42] from sequential behavior.

1) User Representation Encoder: In this part, we would introduce the encoding network, which is composed of the representation backbone of user sequential behavior and projection head. As discussed above, we instantiate the pre-training network by employing self-attentive based Transformer block as backbone network to model user sequence actions. The self-attentive architecture consists of embedding layers and self-attention blocks.

a) Embedding Layer: In the embedding mapping stage, we create an item embedding matrix $\mathbf{M} \in \mathbb{R}^{|\mathcal{I}| \times \mathrm{d}}$, where d denotes the latent dimension. Given a n-length item sequence, we apply a look-up operation from $\mathbf{M}$ to form the input embedding matrix $\mathbf{E} \in \mathbb{R}^{\mathrm{n} \times \mathrm{d}}$. Besides, we incorporate a learnable position embedding $\mathbf{P} \in \mathbb{R}^{\mathrm{n} \times \mathrm{d}}$ to enhance the input representation of the item sequence. By this means, the sequence representation $\mathbf{E}_{\mathrm{I}} \in \mathbb{R}^{\mathrm{n} \times \mathrm{d}}$ can be obtained by summing two embedding matrices: $\mathbf{E}_{\mathrm{I}} = \mathbf{E} + \mathbf{P}$.

b) Self-attention Block: Based on the embedding layer, we develop the item encoder by stacking multiple self-attention blocks. A self-attention block generally consists of two sub-layers, i.e., a multi-head self-attention layer and a point-wise feed-forward network. The multi-head self-attention mechanism has been adopted for effectively extracting the information selectively from different representation subspaces. Specifically, the multi-head self-attention is defined as:

$$\begin{aligned} &\mathrm{MultiHeadAttn}(\mathbf{F}^{\mathrm{l}}) = [\mathrm{head}_1, \mathrm{head}_2, ..., \mathrm{head}_{\mathrm{h}}]\mathbf{W}^{\mathrm{O}}, \\ &\mathrm{head}_{\mathrm{i}} = \mathrm{Attention}(\mathbf{F}^{\mathrm{l}}\mathbf{W}_{\mathrm{i}}^{\mathrm{Q}}, \mathbf{F}^{\mathrm{l}}\mathbf{W}_{\mathrm{i}}^{\mathrm{K}}, \mathbf{F}^{\mathrm{l}}\mathbf{W}_{\mathrm{i}}^{\mathrm{V}}), \end{aligned} \tag{1}$$

where the F^l is the input for l-th layer. When $l = 0$, we set $\mathrm{F}^0 = \mathrm{E}_{\mathrm{I}}$, and the projection matrices $\mathbf{W}_{\mathrm{i}}^{\mathrm{Q}}, \mathbf{W}_{\mathrm{i}}^{\mathrm{K}}, \mathbf{W}_{\mathrm{V}}^{\mathrm{Q}} \in \mathbb{R}^{\mathrm{d} \times \mathrm{d}}$ and $\mathbf{W}^{O} \in \mathbb{R}^{d \times d}$ are the corresponding learnable parameters for each attention head. The attention function is implemented by scaled dot-product operation:

$$\mathrm{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \mathrm{softmax}(\frac{\mathbf{Q}\mathbf{K}^{\top}}{\sqrt{\mathrm{d}/\mathrm{h}}})\mathbf{V}, \tag{2}$$

where $\mathbf{Q} = \mathbf{F}^{\mathrm{l}}\mathbf{W}_{\mathrm{i}}^{\mathrm{Q}}, \mathbf{K} = \mathbf{F}^{\mathrm{l}}\mathbf{W}_{\mathrm{i}}^{\mathrm{K}}$, and $\mathbf{V} = \mathbf{F}^{l}\mathbf{W}_{i}^{V}$ are the linear transformations of the input embedding matrix, $\sqrt{d/h}$ is the scale factor to avoid large values of the inner product.

Though multi-head self-attention is beneficial to extract useful information from previous items' embeddings with adaptive weights, it is still based on a simple linear transformation. To endow the model with non-linearity, a two-layer feed-forward network is applied. Formally, the computation is defined as:

$$\begin{aligned} &\mathbf{F}^{\mathrm{l}} = [\mathrm{FFN}(\mathbf{F}_1^{\mathrm{l}})^{\top}; ...; \mathrm{FFN}(\mathbf{F}_{\mathrm{n}}^{\mathrm{l}})^{\top}], \\ &\mathrm{FFN}(\mathrm{x}) = (\mathrm{ReLU}(\mathrm{x}\mathbf{W}_1 + \mathbf{b}_1))\mathbf{W}_2 + \mathbf{b}_2, \end{aligned} \tag{3}$$

where $\mathbf{W}_1, \mathbf{b}_1, \mathbf{W}_2, \mathbf{b}_2$ are trainable parameters. It should be noted that we omit the description of used dropout mechanism in [41] for clarity.

c) Projection Network: Inspired by the idea of label embedding [43], we convert the discrimination problem of user behavior sequence into latent vector matching tasks. Hence, we additionally add a projection head to further project the augmented sequence into a common high-dimensional space, in which the similarity of two augmented views is readily computed as the distance between them. In detail, we employ 3 fully connected (FC) layers, equipped with batch normalization (BN) operation for each layer, as projection head by following [15]. In particular, we only retain the user representation module at the fine-tuning stage.

2) Prediction Network: To prevent the model collapse issues [**?**], we further add a predictor head for the student network. Similarly, we employ a 2 FC layer equipped with BN operation as the prediction head. Note that this predictor is only applied to the student branch, making the architecture asymmetric between the student and target pipeline.

3) Contrastive Optimization Objective: The representation distance between two views of corrupted sequence input can be measured by cosine similarity. Equivalently, we take the negative through cosine similarity as the optimization objective and minimize the semantic matching error between the normalized predictions and target projections,

$$\mathcal{D}(p_1, z_2) = -\frac{p_1}{||p_1||_2} \cdot \frac{z_2}{||z_2||_2}, \tag{4}$$

where $\| \cdot \|_2$ denotes l_2-norm, $p_1 \triangleq h_{\vartheta}(f_{\theta}(\bar{x}^u))$ and $z_2 \triangleq f_{\theta}(\tilde{x}^u)$. This is equivalent to the mean squared error of l_2-normalized vectors, up to a scale of 2.

Furthermore, inspired by previous work [38], we define a symmetrized loss as:

$$\mathcal{L}_{pre} = \frac{1}{2}\mathcal{D}(p_1, z_2) + \frac{1}{2}\mathcal{D}(p_2, z_1). \tag{5}$$

Note that one key component for our pre-trained network is a stop-gradient (sg) operation, which is used to avoid the model collapse issue. Such operation has been widely adopted in several existing prevalent contrastive learning approaches [13], [38]. Hence, we modify it as

$$\mathcal{D}(p_1, \mathrm{sg}(\mathrm{z}_2)), \tag{6}$$

where it means that z_2 is treated as constant in this term. Similarly, the equation in Eq (5) can be implemented as:

$$\mathcal{L}_{pre} = \frac{1}{2}\mathcal{D}(p_1, \mathrm{sg}(\mathrm{z}_2)) + \frac{1}{2}\mathcal{D}(\mathrm{p}_2, \mathrm{sg}(\mathrm{z}_1)), \tag{7}$$

where the encoder on $\tilde{x}^u$ receives no gradient from z_2 in the first term, while it receives gradients from p_2 in the second term. Here, we employ the SGD optimizer and use it in a mini-batch manner. In addition, we briefly summarize the proposed pre-training network in Algorithm 1.

B. Data Augmentation Strategies

In this subsection, we discuss another core part in CLUE, i.e., how to construct different views from the same sequential user behaviors via data augmentation (DA) strategies. The appropriate DA plays a vital role in CLUE framework since

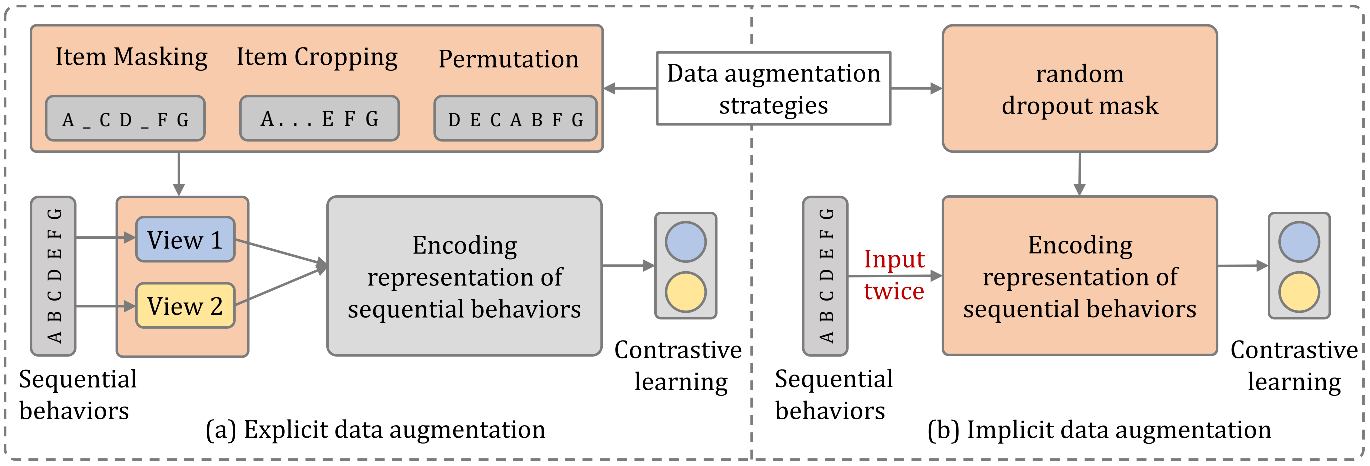

Fig. 2. Illustration of the data augmentation strategies for generating views for contrastive learning.

Algorithm 1 Contrastive Pre-training Network

```
1: Initialize parameters $\theta, \vartheta$.
2: **While** $\theta, \vartheta$ has not converged **do**
3:     $\bar{x}$, $\tilde{x}$ = Aug(x), Aug(x).
4:     $z_1, z_2 = f_\theta(\bar{x}_1), f_\theta(\tilde{x}_2)$.
5:     $p_1, p_2 = h_\vartheta(f(x_1)), h_\vartheta(f(x_2))$.
6:     Update parameters $\theta, \vartheta$ by computing the gradients
7:     of the contrastive loss function in Eq (7).
8: **end while**
```

the essence of contrastive learning is committed to learning the invariance of different views of the same data [15].

Although data augmentation techniques have been widely explored in other areas, it still remains under-explored on how to generate self-supervised training signals from massive user behavior sequences. Hence, we would discuss how to construct positive instance pairs for contrastive pre-training framework from two perspectives as shown in Figure 2.

1) Explicit Data Augmentation: By revisiting previous works [10], [16], we could obtain satisfied recommendation lists without strictly following the chronological order. Actually, such intuition can be easily accepted in real-world scenarios. Take a toy example, after a user purchases phone, she may click a phone case and screen protector in the session, but there is no obvious sequential dependency among these items - in other words, it is likely that the user takes action in any order. Based on the above analysis, we introduce three augmentation approaches. (1) item masking: following, random items are sampled and replaced with [MASK] elements; (2) item cropping: randomly select a continuous sub-sequence and drop it off; (3) permutation: random select a piece of subsequence segments are shuffled it with a random order.

2) Implicit Data Augmentation: The principal assumptions of self-supervised contrastive learning are to seek representations of the world that are invariant to a family of viewing conditions. It is natural since it is commonsense that how you look at an object does not change its identity. Hence, we propose to generate noise views by leveraging the random dropout operation [22], which implicitly takes effect in the body of neural network. The idea is extremely simple: the key ingredient to getting this to work with identical positive pairs is through the use of independently sampled dropout masks. In standard training self-attentive-based sequential architecture, there is a dropout mask placed on fully connected layers as well as attention probabilities. We simply feed the same sequential behaviors to the encoder network *twice* by applying different dropout masks. Please note that it is just the standard dropout mask in the Transformer block and we do not add any additional dropout. In this way, the positive pair takes exactly the same sequence behavior as input, in which the only difference about the embedding is caused by the dropout. Extensive experiments about these implicit augmentation methods would be further discussed in the experimental part.

C. User Representation Evaluation

In this subsection, we aim to adapt the learned representation to specific downstream tasks.

1) User Representation Transferring: After the pre-trained stage, we obtain the pre-trained user representation model which has been optimized with training examples derived directly from unlabeled raw data. For obtaining each user representation, we propose to insert a special token -User Representation ([UR] token) - at the end of the user behavior sequence during the pre-training and treat its embedding as the representation of the current user. Then, followed by a simple classifier (e.g., softmax classifier), the user representation can be used to construct a predictor for downstream tasks. Instead of introducing a new parameter transferring technique, we primarily interested in showing readers how would the learned user representations perform applied in downstream tasks. In this work, we evaluate the target task by fine-tuning the whole pre-trained user encoder. We are aware of other advanced user representation manner which might improve the performance of downstream tasks, such as averaging the item representation at the last layer. We leave it in the future since it is not the focus of this work.

*2) **Adapting to User-related Tasks**:* In this work, we aim to evaluate the effectiveness of learned pre-trained user representations based on three types of downstream tasks: 1) cold-start recommendation; 2) user modeling refers to the process of obtaining the user profile, which is a conceptual understanding of the user for personalized RS service; 3) sequential recommendation refers to modeling user interests by capturing sequential patterns on historical interaction records. All of these three downstream tasks can be regarded as standard multi-class classification optimization problem [8], [26], [44] by learning the connection between context input and supervision signals. Without loss of generality, we consistently employ softmax cross-entropy loss as the optimization objective for three downstream tasks. Further, we describe more task-specific downstream task details in the corresponding subsection of Section V.

D. Summary and Remarks

To summary, we propose a self-supervised pre-training framework for obtaining user useful user representation via sequence-level contrast optimization. Actually, the proposed pre-training framework is very flexible. First, the proposed pre-training network is a model-agnostic framework. That is, the user representation backbone is not limited by the used self-attentive-based architecture. Other powerful user encoding representation networks, such as RNN-based, CNN-based models, can also be used to instantiate the pre-training network. Second, the pre-training framework performs asymmetric architecture since that such structure can help alleviate the model collapse issue cased by removing negative instance sampling process. Third, the implicit augmentation strategies are very simple but very effective in obtaining different views of the same sequential user behaviors. This insightful setting largely simplifies the procedures of CLUE since many explicit DA strategies can be omitted.

V. Experiments

In this section, we conduct extensive experiments to validate the effectiveness of the CLUE framework.

A. Experimental Settings

To evaluate the effectiveness of our proposed CLUE, we respectively apply the learned generic user representations for three user-related tasks: cold-start recommendation, user profiling, and sequential recommendation tasks. In the several downstream tasks, we encourage the knowledge learned from the self-supervision training signals can be helpful for boosting downstream tasks. To verify whether the self-supervised learned user representation helpful for downstream tasks, we compare CLUE with two main cases: well-pre-trained and non-pre-trained settings. We denote CLUE with randomly initialized weights as CLUEZero.

1) Cold-start Recommendation: We first empirically study the effectiveness of the learned user representation by transferring knowledge cross different platform for cold-start recommendation. To be more specific, we aim to verify whether the learned user representations can be universal and effective to benefit cold-start recommendation across different platforms, where the same users are involved. For well-pre-trained settings, we choose PeterRec [16] as competitive baselines, in which mask item prediction loss is considered as the pre-training optimization objective to learn generic user representations for downstream tasks. Furthermore, for a fair comparison, we also treat the user behavior in source domain as features and feed it into DeepFM [45], NFM [46]. Here, we conduct experiments on several large scale datasets made public by [16] including two cold-start recommendation and three user profile datasets.

- ColdRec-1: This contains both source and target datasets. The source dataset is the news recommendation data collected from QQ Browser[1] recommender systems. For each user, the behavior sequence is constructed by using her 50 recent interaction records in chronological order. For users that have less than 5- interactions, we simply pad with zero in the beginning of sequence. The target dataset is collected from Kandian[2] where interaction can be a piece of news, a video, or an advertisement. All users in Kandian are cold with at most three interactions (i.e., $g \leq 3$) and half of them have only one interaction. It should be noted that all users in the target domain have the corresponding records in the source dataset.
- ColdRec-2: It has similar characteristics with ColdRec-1. Likewise, all users in the target dataset have corresponding records in the source dataset. The source dataset contains recent 100 interactions of each user including news and videos. The users in the target dataset have at most 5 interactions (i.e., $g \leq 5$).

2) User Profiling: For user profiling task, we choose the same baselines as the setting of cold-start recommendation.

- AgeEst: It has only a target dataset since all users are from the source dataset of ColdRec-2, each instance in AgeEst is a user and her bracket label ($g = 1$) - one class represents 10 years. The source dataset contains recent 100 interactions of each user including news and videos.
- GenEst: Similar to AgeEst, each instance in GenEst is a user and her gender (male or female) label ($g = 1$) obtained by the registration information.
- LifeEst: Similar to AgeEst, each instance in LifeEst is a user and her life status label ($g = 1$), e.g., single, married, pregnancy or parenting.

3) Sequential Recommendation: In addition, we also verify the effectiveness of pre-trained methods in terms of enhancing sequential recommendation problem. Here, we choose S^{3}Rec [18] as the competitive well-pre-trained baselines, which is proposed for enhance sequential recommendation by

[1]https://browser.qq.com/

[2]https://sdi.3g.qq.com/v/2019111020060911550

TABLE I
STATICS OF THE DATASETS FOR DOWNSTREAM USER-ORIENTED TASKS.

Dataset	Cold-start recommendation		User profiling			Sequential recommendation	
	ColdRec-1	ColdRec-2	AgeEst	GenEst	LifeEst	Movielens30	Movielens50
# Num. users	1,649,095	1,472,248	1,551,357	1,548,844	1,075,010	876,162	552,438
# Num. source items	191,014	645,980	-	-	-	-	-
# Max length in source domain	50	100	-	-	-	-	-
# Num. target items (classes)	20,342	17,879	646,279	645,729	487,728	23,515	23,515
# Max length in target domain	3	5	100	100	100	30	50

TABLE II
PERFORMANCE COMPARISON IN TERMS OF *MRR*@10 AND *Recall*@10 ON COLD-START RECOMMENDATION AND *Acc* ON USER PROFILING .

Model	ColdRec-1		ColdRec-2		AgeEst	GenEst	LifeEst
	MRR@10	Recall@10	MRR@10	Recall@10	Acc	Acc	Acc
NFM	0.0070	0.0075	0.0420	0.0363	0.6156	0.8940	0.5135
DeepFM	0.0090	0.0100	0.0434	0.0373	0.6312	0.9025	0.5153
CLUEZero	0.0119	0.0129	0.0464	0.0393	0.6248	0.9015	0.5123
PeterRec	0.0123	0.0134	0.0470	0.0397	**0.6369**	**0.9056**	0.5191
CLUE	**0.0139**	**0.0150**	**0.0481**	**0.0403**	0.6268	0.9019	**0.5354**

regarding the mutual information maximization principle as pre-training optimization objective. In addition, we also choose previous representative sequential recommenders, including GRU4Rec [8] and Caser [28], as compared baselines. Here, we conduct extensive experiments on the benchmark dataset MovieLens[3].

- MovieLens30 & MovieLens50: To make the reliability of experimental results, we perform the basic pre-processing by filtering out interactions with less than 5 users and users with less than 10 items. Then, we define the maximum length of the interaction sequence as 30 (or 50). Sequences shorter than 30 (or 50) will be padded with zero at the beginning of the sequence.

More statistics of our datasets are presented at Table I.

4) Evaluation Metrics: For cold-start recommendation and user profile prediction, we randomly split each dataset into training (80%), validation(5%), and test (15%) by following [16], [45]. As for sequential recommendation, following previous works [41], [47], [48], we apply the *leave-one-out* strategy for evaluating the performance of sequential recommendation. Concretely, for each user interaction sequence, the last item is used as the test data, the item before the last one is used as the validation data, and the remaining data is used for training. For the item recommendation task in sequential modeling and cold-start issues, we use two popular ranking metrics - Mean Reciprocal Rank (*MRR*@10) and *Recall*@10. Note that we calculate metrics according to the ranking of all candidate items rather than the sampled version and report the average score of overall test users. We refer interested readers to [49] for more details. We also use the classification accuracy (denoted by *Acc*) for user profile prediction, where $Acc =$ number of correct predictions/total number of predictions. For all metrics, the higher the value, the better the performance.

[3]http://files.grouplens.org/datasets/movielens/

5) Implementation Details: All of the methods are trained on a Linux server with two 2.20 GHz Intel Xeon E5-2650 CPUs and four Tesla V100 GPUs. For the self-attentive architecture[4], Caser[5], GRU4Rec [6], we use codes provided by the corresponding authors. For NFM and DeepFM, we implement it by Tensorflow. For common parameters in all models, we set the batch size b as 64 due to computation cost issue and set the embedding hidden size with 64. All other hyper-parameters and initialization strategies are either followed by the suggestion from the authors' methods or tuned on validation sets. We report the results of each baseline under its optimal hyper-parameter settings. In particular, the number of self-attention blocks and attention heads are set as 2 and 4 for all experiments. In the Transformer architecture, the default dropout probability is 0.5. In both pre-training and fine-tuning stage, the learning rate is set to 0.001, and all methods are optimized by standard Adam optimizer. The explicit augmentation proportion and dropout probability in implicit augmentation is search from $[0.1, 0.2, ..., 0.9]$ for several data augmentation techniques. It should be noted that dropout strategy is the default setting for the experimental results of CLUE.

B. Overall Performance of Downstream Tasks

The results of downstream tasks, evaluated by fine-tuning the entire pre-trained model, are respectively shown in Table II and Figure 3. Based on the results, we can find:

For several baseline methods in the cold-start recommendation and user profiling, we notice that self-attentive recommendation architecture respectively performs superior performance in the cold-start recommendation and shows competitive performance in user profile prediction tasks compared

[4]https://github.com/kang205/SASRec
[5]https://github.com/graytowne/caserpytorch
[6]https://github.com/hidasib/GRU4Rec

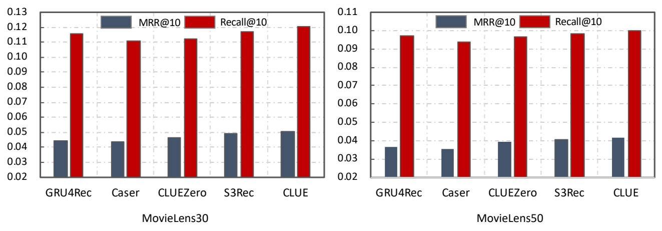

Fig. 3. Sequential recommendation performance comparison.

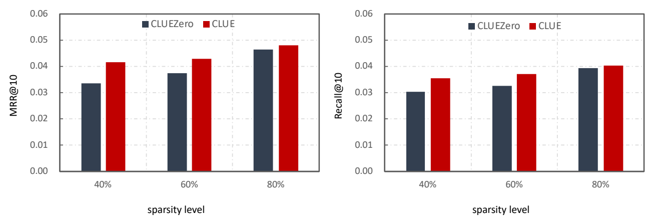

Fig. 4. Cold-start recommendation performance comparison w.r.t the amount of training data on ColdRec-2.

with non-sequential based methods, i.e., NFM and DeepFM. Such results reflect it is necessary to consider the sequential behavior pattern for cold-start recommendation and user profiling tasks. Also, we notice that the self-attentive surpass previous sequential recommender models, i.e., GRU4Rec and Caser, in sequential recommendation tasks. Such experimental phenomenon indicates that self-attentive-based architecture is largely suitable for capturing sequential behavior patterns for item recommendation tasks.

For cold-start recommendation and user profiling tasks, we notice that all pre-training-based methods could consistently benefit the three downstream tasks by transferring knowledge from constructed self-supervised training signals. Since all three methods, including CLUEZero, PeterRec, and CLUE, adopt the same architecture strictly following the same hyper-parameter setting, such results suggest that the learned user representation (from different platforms) can be transferred to help downstream tasks while the same users are involved. Also, we could find that our proposed CLUE could surpass previous competitive baseline methods PeterRec in most situations, which adopts item-level optimization objectives for obtaining pre-trained parameters. Such results demonstrate that sequence-level pre-training be more suitable for characterizing user similarity or achieve competitive results.

For sequential recommendation tasks, compared with baselines, it is clear to see that CLUE performs better than them on two datasets. Such results reflect the necessity of explicitly optimizing user representation for sequential recommendation tasks. The main reason behind such phenomenon reflects the transferred knowledge trained with our sequence-level optimization goal can be helpful for item-level sequential recommendation objective.

C. Performance Comparison w.r.t the Training Data Sparsity

In actually, in CV or NLP-like domain, pre-training methods have been empirically verified in largely benefiting the target task while data is extreme sparsity. Hence, we are interested in analyzing similar phenomenon in RS area by simulating the data sparsity scenarios with leveraging different proportions of the full dataset, i.e., 40%, 60%, 80%. For clarity and saving space purpose, we only report the best results of CLUE with dropout augmentation strategy on ColdRec-2 and show the specific performance comparison in Figure 4. As we can see, the recommendation performance substantially drops when less training data is used. Nevertheless, CLUE consistently outperforms baselines especially in data with a larger sparsity level (40%). This observation implies that the CLUE shows some potential to alleviate the data sparsity issue via fully mining the knowledge from limited user behavior information.

TABLE III
PERFORMANCE COMPARISON W.R.T VARIOUS DATA AUGMENTATION STRATEGIES INCLUDING THREE EXPLICIT AND ONE IMPLICIT STRATEGY.

DA strategies	MovieLens30		MovieLens50	
	MRR@10	Recall@10	MRR@10	Recall@10
Item masking	0.0494	0.1178	0.0420	0.1006
Item cropping	0.0483	0.1163	0.0395	0.0953
Permutation	0.0489	0.1169	0.0406	0.0980
Dropout	0.0509	0.1203	0.0419	0.0999

D. Impact of Data Augmentation Strategies

In this subsection, we turn to analyze the performance of CLUE in the setting of different augmentation strategies as illustrated in Section IV-B.To save page space, we only report the results of sequential recommendation on MovieLens30 and MovieLens50. We show the sequential recommendation performance in Table III.

First, we could find that a few explicit augmentation compositions only perform slightly better than compared baselines. The main reason behind such results might be such augmentation composition strategies dramatically destroy the original user sequence behavior and provide limited useful knowledge for downstream tasks. Surprisingly, we notice that the simple implicit data augmentation strategies, i.e., leveraging dropout mask to generate augmented views for self-supervised contrastive learning, achieve competitive or even superior performance than common explicit strategies. Actually, such insightful findings are very useful and can largely simplify the procedures of our proposed self-supervised contrastive learning framework since these complicated explicit DA strategies can be replaced.

Based on this, we also study the performance comparison w.r.t different dropout ratio range from $\{0.1, 0.3, 0.5, 0.7, 0.9\}$ for exploring the sensitivity to dropout probability. As is shown in Figure 5, we notice that the implicit augmentation strategy is largely sensitive to the dropout probability. We guess the main reason is the random dropout augmentation strategy has the strong capacity to generate noise transformation for the

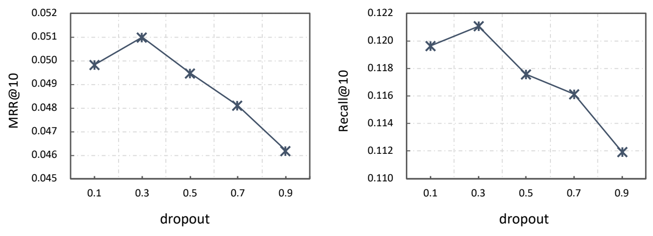

Fig. 5. Impact of the implicit DA (i.e., dropout based) strategies on the MovieLens30 dataset with different dropout probabilities.

same sequential behavior input via non-linear transformation.

E. Ablation Studies of Model Collapse Issue

In this subsection, we conduct ablation experiments to verify the effectiveness of stop-gradient operation and asymmetric framework in alleviating model collapse issues due to discarding the use of negative instances. In view of the limited space, we mainly report the experimental results of sequential recommendation task on MovieLens30. Figure 6 (left) presents a self-supervised contrastive training loss on "with" and "without stop-gradient operation". It should be noted that all hyper-parameters are kept unchanged and stop-gradient is the only difference. Likewise, in Figure 6 (right), we also report the training loss in terms of whether preserve predictor head. From the results, removing either the stop-gradient operator or the predictor head, we can find that the optimizer quickly reached the minimum possible loss of -1. Such results mean that almost all users are mapped into a small area and are somehow collapsed. Such ablation results indicate that it is vital to simultaneously preserve two key components (i.e., asymmetric framework and stop-gradient operation) for preventing severe model collapse [38], [39].

F. Understanding the Pre-trained User Representations

Recently, *uniformity* and *alignment* have been recognized as two key properties for measuring the quality of representations [22], [50]. Specifically, the former property indicates how well the embedding are uniformly distributed while the latter property measure how close the semantic similar instance while are mapped in latent space. To deeply understand the quality of user representation trained via different methods, we project the user representation into 2-dimensional space using PCA for visualization. For clarity and saving space purpose, we only show the case of user embedding trained by PeterRec and CLUE (leveraging dropout augmentation strategy) on GenEst dataset. From the results shown in Figure 7, we notice that the user representation optimized with PeterRec are mapped in a narrow space compared with the user embeddings trained with CLUE. To some extent, we can find that such visualization provides some evidence for help illustrating why the user representation trained with CLUE could achieve promising results for downstream tasks. Actually, such insightful findings can motivate us further deeply think about the key properties w.r.t user digital representations.

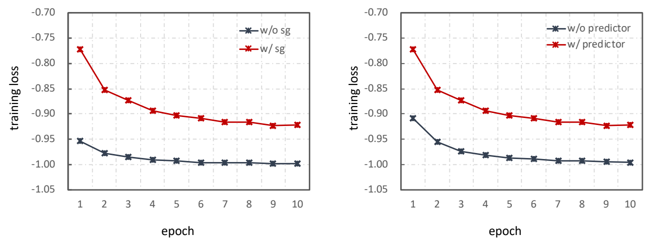

Fig. 6. Ablation analysis: (a) removing the predictor head; (b) removing the stop-gradient (sg) operator.

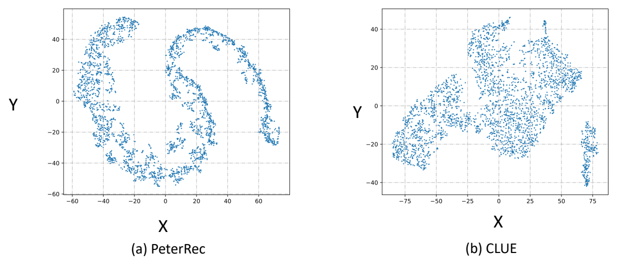

Fig. 7. User representation 2D visualization on GenEst dataset: (a) visualization of user embeddings trained with PeterRec, and (b) visualization of user embeddings from CLUE.

VI. Conclusions and Future Work

In this work, we delved the research into learning transferable user representations, which can be utilized to benefit multiple user-related tasks. We argue that modeling the user behaviors with sequence-level optimization objective can bring more benefits for learning user-oriented representations. Hence, we proposed a novel self-supervised pre-training framework called CLUE, in which we innovatively adopted contrastive learning as pre-training optimization objective for sequence-level user representations optimization. Furthermore, we also proposed a new perspective on data augmentation via adopting continual transformation with user sequence as input. Extensive empirical studies were conducted on several user-related tasks, showing the superiority of CLUE in characterizing user similarity from sequential behaviors compared with competitive approaches. Furthermore, we also reported some insightful observations by extensive ablation studies, which might be directions for future research in the recommender systems area.

We believe our attempts can inspire more work, and summarize the potential future directions from three main directions: (1) extend our CLUE by considering user behavior records and content information, simultaneously; (2) fuse item-level prediction loss into CLUE methods for furthering enhancing user representations; (3) explore automatic data augmentation techniques or adaptively corrupt the sequence input according to the specific downstream user modeling tasks.

Acknowledgements. This research was partially supported by grants from the National Natural Science Foundation of China (Grants No. 61922073, U20A20229, 71802068 and WK5290000002).

References

[1] Q. Liu, Y. Ge, Z. Li, E. Chen, and H. Xiong, "Personalized travel package recommendation," in *IEEE International Conference on Data Mining (ICDM)*. IEEE, 2011, pp. 407–416.

[2] R. Yu, Q. Liu, Y. Ye, M. Cheng, E. Chen, and J. Ma, "Collaborative list-and-pairwise filtering from implicit feedback," *IEEE Transactions on Knowledge and Data Engineering*, 2020.

[3] P.-S. Huang, X. He, J. Gao, L. Deng, A. Acero, and L. Heck, "Learning deep structured semantic models for web search using clickthrough data," in *ACM CIKM*, 2013, pp. 2333–2338.

[4] Q. Liu, E. Chen, H. Xiong, C. H. Ding, and J. Chen, "Enhancing collaborative filtering by user interest expansion via personalized ranking," *IEEE TTSMC, Part B (Cybernetics)*, vol. 42, no. 1, pp. 218–233, 2011.

[5] R. Salakhutdinov, A. Mnih, and G. Hinton, "Restricted boltzmann machines for collaborative filtering," in *ICML*, 2007, pp. 791–798.

[6] S. Sedhain, A. Menon, S. Sanner, and L. Xie, "Autorec: Autoencoders meet collaborative filtering," *WWW*, 2015.

[7] Y. Shen, X. He, J. Gao, L. Deng, and G. Mesnil, "A latent semantic model with convolutional-pooling structure for information retrieval," in *ACM CIKM*, 2014, pp. 101–110.

[8] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk, "Session-based recommendations with recurrent neural networks," *arXiv preprint arXiv:1511.06939*, 2015.

[9] Z. Li, H. Zhao, Q. Liu, Z. Huang, T. Mei, and E. Chen, "Learning from history and present: Next-item recommendation via discriminatively exploiting user behaviors," in *ACM SIGKDD*, 2018, pp. 1734–1743.

[10] F. Yuan, A. Karatzoglou, I. Arapakis, J. M. Jose, and X. He, "A simple convolutional generative network for next item recommendation," in *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, 2019, pp. 582–590.

[11] A. Radford, "Improving language understanding by generative pre-training," 2018.

[12] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[13] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 9729–9738.

[14] X. Chen, S. Xie, and K. He, "An empirical study of training self-supervised vision transformers," *arXiv preprint arXiv:2104.02057*, 2021.

[15] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," *arXiv preprint arXiv:2002.05709*, 2020.

[16] F. Yuan, X. He, A. Karatzoglou, and L. Zhang, "Parameter-efficient transfer from sequential behaviors for user modeling and recommendation," in *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2020, pp. 1469–1478.

[17] O. Barkan, A. Caciularu, I. Rejwan, O. Katz, J. Weill, I. Malkiel, and N. Koenigstein, "Cold item recommendations via hierarchical item2vec."

[18] K. e. a. Zhou, "S3-rec: Self-supervised learning for sequential recommendation with mutual information maximization."

[19] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "Bpr: Bayesian personalized ranking from implicit feedback," in *UAI*. AUAI Press, 2009, pp. 452–461.

[20] M. Cheng, R. Yu, Q. Liu, V. W. Zheng, H. Zhao, H. Zhang, and E. Chen, "Alpha-beta sampling for pairwise ranking in one-class collaborative filtering," in *2019 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2019, pp. 1000–1005.

[21] F. Yuan, G. Zhang, A. Karatzoglou, J. Jose, B. Kong, and Y. Li, "One person, one model, one world: Learning continual user representation without forgetting," in *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2021, pp. 696–705.

[22] T. Gao, X. Yao, and D. Chen, "Simcse: Simple contrastive learning of sentence embeddings," *arXiv preprint arXiv:2104.08821*, 2021.

[23] P. Wang, Y. Fu, H. Xiong, and X. Li, "Adversarial substructured representation learning for mobile user profiling," in *ACM SIGKDD*, 2019, pp. 130–138.

[24] P. Wang, K. Liu, L. Jiang, X. Li, and Y. Fu, "Incremental mobile user profiling: Reinforcement learning with spatial knowledge graph for modeling event streams," in *ACM SIGKDD*, 2020, pp. 853–861.

[25] Y. Koren, "Factorization meets the neighborhood: a multifaceted collaborative filtering model," in *ACM SIGKDD*, 2008, pp. 426–434.

[26] M. Cheng, F. Yuan, Q. Liu, S. Ge, Z. Li, R. Yu, D. Lian, S. Yuan, and E. Chen, "Learning recommender systems with implicit feedback via soft target enhancement," in *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2021, pp. 575–584.

[27] Y. Wu, C. DuBois, A. Zheng, and M. Ester, "Collaborative denoising auto-encoders for top-n recommender systems," in *WSDM '16*, 2016.

[28] J. e. a. Tang, "Personalized top-n sequential recommendation via convolutional sequence embedding," in *ACM WSDM*, 2018, pp. 565–573.

[29] Y. Sun, F. Yuan, M. Yang, G. Wei, Z. Zhao, and D. Liu, "A generic network compression framework for sequential recommender systems," in *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2020, pp. 1299–1308.

[30] J. Ma, C. Zhou, H. Yang, P. Cui, X. Wang, and W. Zhu, "Disentangled self-supervision in sequential recommenders," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020, pp. 483–491.

[31] Y. Yu, Y. Li, J. Shen, H. Feng, J. Sun, and C. Zhang, "Steam: Self-supervised taxonomy expansion with mini-paths," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020, pp. 1026–1035.

[32] X. Xie, F. Sun, Z. Liu, S. Wu, J. Gao, B. Ding, and B. Cui, "Contrastive learning for sequential recommendation," *arXiv preprint arXiv:2010.14395*, 2020.

[33] M. E. Peters and M. N. et al, "Deep contextualized word representations," in *NAACL-HLT*, 2018.

[34] M. Ye, X. Zhang, P. C. Yuen, and S.-F. Chang, "Unsupervised embedding learning via invariant and spreading instance feature," in *IEEE CVPR*, 2019, pp. 6210–6219.

[35] P. Bachman, R. D. Hjelm, and W. Buchwalter, "Learning representations by maximizing mutual information across views," in *Advances in Neural Information Processing Systems*, 2019, pp. 15 535–15 545.

[36] Z. e. a. Wu, "Unsupervised feature learning via non-parametric instance discrimination," in *IEEE CVPR*, 2018, pp. 3733–3742.

[37] Y. Tian, D. Krishnan, and P. Isola, "Contrastive multiview coding," *arXiv preprint arXiv:1906.05849*, 2019.

[38] J.-B. Grill and F. e. a. Strub, "Bootstrap your own latent: A new approach to self-supervised learning," *arXiv preprint arXiv:2006.07733*, 2020.

[39] X. Chen and K. He, "Exploring simple siamese representation learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 15 750–15 758.

[40] F. Yuan, G. Guo, J. M. Jose, L. Chen, H. Yu, and W. Zhang, "Lambdafm: learning optimal ranking with factorization machines using lambda surrogates," in *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, 2016, pp. 227–236.

[41] W.-C. Kang and J. McAuley, "Self-attentive sequential recommendation," in *IEEE ICDM*. IEEE, 2018, pp. 197–206.

[42] P. Zhao, K. Xiao, Y. Zhang, K. Bian, and W. Yan, "Amer: Automatic behavior modeling and interaction exploration in recommender system," *arXiv preprint arXiv:2006.05933*, 2020.

[43] S. Bengio, J. Weston, and D. Grangier, "Label embedding trees for large multi-class tasks," 2010.

[44] P. Covington, J. Adams, and E. Sargin, "Deep neural networks for youtube recommendations," in *Proceedings of the 10th ACM conference on recommender systems*, 2016, pp. 191–198.

[45] H. Guo and R. e. a. Tang, "Deepfm: a factorization-machine based neural network for ctr prediction," *arXiv preprint arXiv:1703.04247*, 2017.

[46] X. He and T.-S. Chua, "Neural factorization machines for sparse predictive analytics," in *ACM SIGIR*, 2017, pp. 355–364.

[47] F. S. et al, "Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer," *ACM CIKM*, 2019.

[48] R. Ren, Z. Liu, Y. Li, W. X. Zhao, H. Wang, B. Ding, and J.-R. Wen, "Sequential recommendation with self-attentive multi-adversarial network," *arXiv preprint arXiv:2005.10602*, 2020.

[49] W. Krichene and S. Rendle, "On sampled metrics for item recommendation," in *Proceedings of the 26th ACM SIGKDD*, 2020, pp. 1748–1757.

[50] T. Wang and P. Isola, "Understanding contrastive representation learning through alignment and uniformity on the hypersphere," in *ICML*. PMLR, 2020, pp. 9929–9939.