Contrastive Curriculum Learning for Sequential User Behavior Modeling via Data Augmentation

Shuqing Bian[1,3], Wayne Xin Zhao[2,3*], Kun Zhou[1,3],
Jing Cai[4], Yancheng He[4], Cunxiang Yin[4], Ji-Rong Wen[1,2,3]
[1]School of Information, Renmin University of China, Beijing, China
[2]Gaoling School of Artificial Intelligence, Renmin University of China, Beijing, China
[3]Beijing Key Laboratory of Big Data Management and Analysis Methods, Beijing, China
[4]Platform and Content Group, Tencent, Shenzhen, China
{shuqingbian, batmanfly}@gmail.com, {samscai, collinhe, jasonyin}@tencent.com, jrwen@ruc.edu.cn

ABSTRACT

Within online platforms, it is critical to capture the semantics of sequential user behaviors for accurately modeling user interests. However, dynamic characteristics and sparse behaviors make it difficult to learn effective user representations for sequential user behavior modeling.

Inspired by the recent progress in contrastive learning, we propose a novel Contrastive Curriculum Learning framework for producing effective representations for modeling sequential user behaviors. We make important technical contributions in two aspects, namely *data quality* and *sample ordering*. Firstly, we design a model-based data generator by generating high-quality samples confirming to users' attribute information. Given a target user, it can leverage the fused attribute semantics for generating more close-to-real sequences. Secondly, we propose a curriculum learning strategy to conduct contrastive learning via an easy-to-difficult learning process. The core component is a learnable difficulty evaluator, which can score augmented sequences, and schedule them in curriculums. Extensive results on both public and industry datasets demonstrate the effectiveness of our approach on downstream tasks.

CCS CONCEPTS

• **Information systems** → **Personalization**.

KEYWORDS

User Behavior Modeling, Contrastive Curriculum Learning,

ACM Reference Format:
Shuqing Bian[1,3], Wayne Xin Zhao[2,3*], Kun Zhou[1,3], and Jing Cai[4], Yancheng He[4], Cunxiang Yin[4], Ji-Rong Wen[1,2,3]. 2021. Contrastive Curriculum Learning for Sequential User Behavior Modeling via Data Augmentation. In *Proceedings of the 30th ACM International Conference on Information and Knowledge Management (CIKM '21), November 1–5, 2021, Virtual Event, QLD, Australia.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3459637.3481905

*Corresponding author.

CIKM '21, November 1–5, 2021, Virtual Event, QLD, Australia

ACM ISBN 978-1-4503-8446-9/21/11.
https://doi.org/10.1145/3459637.3481905

1 INTRODUCTION

Recent years have witnessed the great success of many online platforms, such as Amazon and Tencent. On online platforms, users' behaviors are highly dynamic and evolving over time. Thus it is critical to capture the dynamics of sequential user behaviors for modeling user interests. To cope with this problem, various methods have been proposed to learn user representation for making accurate prediction such as sequential recommendation and attribute prediction based on historical interactions [14, 17].

As a major technical approach, sequential modeling methods [13, 31, 44] aim to capture sequential patterns or characteristics underlying users' historical behaviors. Such a motivation has been extensively explored with the recent progress of deep learning. A number of studies have adopted recurrent neural networks (RNNs) [7], convolutional neural networks (CNNs) [32], and self-attention mechanisms [17] to learn good representations of user preference by effectively modeling historical behavior sequences.

Although existing methods show performance improvement, they rely on the loss of the downstream task to learn the model architecture. It is found that such an optimization way is easy to suffer from issues such as data sparsity [11, 30]. Similar issues are also raised in other domains, such as natural language processing and computer vision. In recent years, many efforts have been devoted to tackling this challenge [3, 6]. A mainstream technique is the *pre-training* paradigm, which first pre-trains the model on a large unlabeled corpus and then fine-tunes the model with supervised labels from downstream tasks. In order to better leverage intrinsic data correlations for pre-training, contrastive learning is proposed to improve the sequence modeling [34, 49]. It conducts data augmentation to obtain different data views, and learns the correlation between different views to enhance the representation.

However, there are two major issues when applying contrastive learning in the sequential modeling task. First, the data quality of augmented sequences directly influences the learning performance. Existing methods mainly adopt heuristic strategies for data augmentation, and it is difficult to guarantee the data quality. Second, in representation space, the augmented instances are not evenly generated in the entire semantic space. They are likely to be with *different difficulty levels* (*e.g.,* how easy it is accurately distinguished in representations) for the learner. Therefore, a random sample

ordering (a concept from curriculum learning [1]) with augmented data is likely to yield the performance degeneration in practice [16].

To this end, the key challenges of contrastive learning for sequential modeling can be summarized in two aspects, namely *data quality* and *sample ordering* of the augmented sequences. For data quality, we consider designing a model-based data generator by learning to generate more controllable samples. Since our task is related to user modeling, the generator is able to augment sequences confirming to users' attribute information. For sample ordering, we believe the learning process should be scheduled in an easy-to-difficult process, so that the model capacity can be gradually improved. Therefore, we incorporate a novel curriculum learning strategy by evaluating sample difficulty and scheduling their order.

In this paper, we propose a novel Contrastive Curriculum Learning (CCL) framework for producing effective representations for modeling sequential user behaviors. The basic idea of our approach is to generate high-quality augmented data for contrastive learning and conduct the contrastive curriculum learning via an *easy-to-difficult* training process. Firstly, we develop a context-aware data generator based on the Transformer architecture for data augmentation. Secondly, we design a difficulty evaluator to score the augmented sequences, and schedule them in an easy-to-difficult order. Finally, we propose a curriculum learning strategy to conduct contrastive learning via the elaborately designed curriculum. Putting the three key steps together, our approach is able to gradually improve the learned user representations.

To the best of our knowledge, it is the first time that contrastive curriculum learning has been applied for sequential user modeling. To validate the effectiveness of our approach, we conduct extensive experiments with downstream tasks, namely user activity prediction, item recommendation and user attribute prediction. Experimental results on both public and industry datasets show that our approach is more effective than a number of competitive methods.

2 RELATED WORK

In this section, we summarize the related work in three fields.

Sequential User Behavior Modeling. Sequential user behavior modeling, which captures user preferences from behavior data, is critically important since it contributes significant improvement for real-world applications. As a major direction, many methods [7, 12, 29] in recommender systems have been proposed by constructing user models on user behavior data. Early work mostly focus on Markov chain [26] models. They employ Markov decision processes in the recommender system to provide recommendations using sequential information. With the success of deep learning, researchers adopt deep neural network [14, 31] to model sequential dynamics. Recently, self-attention attains promising results in various sequential behavior modeling tasks. Kang and McAuley [17] firstly adopt the self-attention mechanism for sequential user behavior modeling. Besides that, Many sequence modeling methods have also been applied to user privacy protection [52], user activity prediction [53] and user retention analysis [8].

Curriculum Learning. Curriculum learning is a learning paradigm that starts from simple patterns and gradually increases to more complex patterns. This idea is inspired by the human learning process. Bengio *et al.* [1] examine curriculum learning and demonstrate empirically that such curriculum approaches indeed help decrease training times and sometimes even improve generalization. Curriculum learning has also been applied to many tasks. Jiang *et al.* [15] manage curriculum learning as an optimization problem. Sachan and Xing [27] apply self-paced learning for neural question answering. Tay *et al.* [33] propose curriculum pointer-generator networks for reading comprehension over long narratives. Platanios *et al.* [25] apply curriculum learning for neural machine translation, aiming to reduce the need for specialized training heuristics and boost the performance of existing systems. In our work, our approach is closely related to [25, 33], where a curriculum is formed via two steps: first evaluating the difficulty, then sampling the examples into batches accordingly.

Contrastive Learning. Contrastive learning, which aims to learn high-quality representation via a self-supervised manner, recently achieves remarkable successes in various fields [2, 50]. The common motivation behind these work is the InfoMax principle [37], which we here instantiate as maximizing the mutual information (MI) between two views [43]. It learns discriminative representations by contrasting positive and negative samples. In natural language processing, the most classic model Word2vec [24] uses co-occurring words and negative sampling to learn word embeddings. To efficiently learn sentence representations, Logeswaran and Lee [21] treat the context sentences as positive samples and the others as negative samples to optimize a contrastive loss. In computer vision, a large collection of work [9, 22] learns self-supervised image representation by minimizing the distance between two views of the same image. Its effectiveness has been verified in multiple tasks, including shape recognition and object classification.

Different from these studies, our work proposes a novel contrastive curriculum learning approach for user representation. We design a context-aware data augmentation strategy suitable for user sequential behavior, by which the augmented instances are more "close-to-real". Also, we propose a curriculum learning strategy to conduct contrastive learning via an easy-to-difficult learning process to further enhance the user representation.

3 PRELIMINARIES

In this section, we first formulate the problem statement and then introduce the base model for sequential user modeling.

3.1 Problem Statement

In this part, we present the used notations throughout the paper, and define our task.

Let $\mathcal{U}$ denote a user set and $\mathcal{I}$ denote an item set. We consider the sequential interaction scenario between users and items. On an online platform, a user usually interacts with a number of items at different timestamps. Therefore, in chronological order, we denote the interaction sequence for user u as $s_u = i_1 \rightarrow \cdots \rightarrow i_t \rightarrow \cdots \rightarrow i_n$, where i_t is the item that u has interacted with at step t and n is the length of interaction records for user u. Besides, each user is associated with several attributes (*e.g.,* occupation, city and interest), denoted by $\mathcal{A}_u = \{a_1, \cdots, a_k\}$.

To further enhance the user representation, our core approach is to produce a set of augmented sequence data, which would be

utilized in the pre-training phase. Specifically, for each user u, we apply the augmentation method (see Section 4.1) to the original user interaction sequence s_u and obtain augmented sequence data, denoted by $\mathcal{Z}_u = \{z_1, \cdots, z_j, \cdots, z_m\}$, where z_j denotes the j-th augmented sequence for user u and m is the number of augmented sequences. Each augmented sequence z_j represents similar or related semantics for s_u, which aim to enhance the semantic representation for user u.

Based on the above notations, given both sequential data s_u and augmented sequential data $\mathcal{Z}_u$, our aim is to learn an effective d-dimensional representation for user u, denoted by $\boldsymbol{v}_u \in \mathbb{R}^d$. We expect the learned user representation can capture user interests or sequential behavior characteristics, which can potentially improve various downstream applications.

3.2 Base Model for Sequential User Behavior

For modeling sequential user behaviors, we develop the base model by following the Transformer architecture [35], including the embedding layer and the primary Transformer module, which has been shown to be effective in modeling sequential data [17].

3.2.1 Attribute-Specific Embedding Layer. In original Transformer network, the original embedding layer contains item embedding and position embedding at each position. In our work, we would like to inject useful attribute information for enhancing the sequential modeling. Given the attribute set $\mathcal{A}_u$ of user u, we map each attribute from $\mathcal{A}_u$ into a d-dimensional embedding. Since a user is usually associated with multiple attributes, we perform the average-pooling operation to obtain the attribute-based representation $\boldsymbol{a}_u \in \mathbb{R}^d$ for user u. We further concatenate $\boldsymbol{a}_u$ in n times (sequence length), and derive an attribute embedding matrix $\mathbf{A} \in \mathbb{R}^{n \times d}$ for input. In order to enhance the effect of user attribute information, we add the attribute embedding at each position:

$$\mathbf{E}_I = \mathbf{I} + \mathbf{P} + \mathbf{A}, \tag{1}$$

where $\mathbf{I} \in \mathbb{R}^{n \times d}$ is the item embedding matrix obtained by the look-up operation, $\mathbf{P} \in \mathbb{R}^{n \times d}$ is a position embedding matrix calculated by sinusoidal function [35], and $\mathbf{A} = [\boldsymbol{a}_u]_n$ is the attribute embedding matrix repeating the attribute embedding $\boldsymbol{a}_u$ by n times.

3.2.2 Transformer Module. Based on the above embedding layer, we develop the Transformer module and produce user representation $\boldsymbol{u}$. Generally, the Transformer module is composed by multi-head self-attention (MHA) block and point-wise feed-forward network. The MHA block has been adopted for effectively extracting the information selectively from different representation subspaces. Specifically, the multi-head self-attention is defined as:

$$\text{MHA}(\mathbf{F}^l) = [head_1, head_2, ..., head_h]\mathbf{W}^O, \tag{2}$$

$$head_i = \text{Attention}(\mathbf{F}^l\mathbf{W}_i^Q, \mathbf{F}^l\mathbf{W}_i^K, \mathbf{F}^l\mathbf{W}_i^V), \tag{3}$$

where the $\mathbf{F}^l$ is the input for the l-th layer. When $l = 0$, the input $\mathbf{F}^0$ is set to $\mathbf{E}_I$ (Eq. 1), and the projection matrix $\mathbf{W}_i^Q \in \mathbb{R}^{d \times d/h}$, $\mathbf{W}_i^K \in \mathbb{R}^{d \times d/h}$, $\mathbf{W}_V^Q \in \mathbb{R}^{d \times d/h}$ 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. Furthermore, we endow the non-linearity of the self-attention block by applying a point-wise feed-forward network:

$$\mathbf{F}^l = [\text{FFN}(\mathbf{F}_1^l)^\top; \cdots; \text{FFN}(\mathbf{F}_n^l)^\top], \tag{4}$$

$$\text{FFN}(\boldsymbol{x}) = (\text{ReLU}(\boldsymbol{x}\mathbf{W}_1 + \mathbf{b}_1))\mathbf{W}_2 + \mathbf{b}_2, \tag{5}$$

where $\mathbf{W}_1, \mathbf{b}_1, \mathbf{W}_2$ and $\mathbf{b}_2$ are trainable parameters. In the final layer of the Transformer layer, we utilize the output of the self-attention block at the last position as the final user representation $\boldsymbol{v}_u$.

4 APPROACH

Inspired by recent progress of contrastive learning [34, 42] and curriculum learning [1], we propose a novel Contrastive Curriculum Learning (CCL) framework for producing effective representations for modeling sequential user behaviors.

The basic idea of our approach is to generate high-quality augmented data for contrastive learning and conduct the contrastive curriculum learning via an *easy-to-difficult* training process. As major contributions, the augmented sequences can well conform to user's attribute information, and our approach is able to schedule the augmented sequences and learn with the augmented sequences in *increasing difficulty levels*.

The overview of the proposed CCL framework is presented in Figure 1. First, we train a context-aware data generator by injecting attribute information of a target user, and utilize it to augment new sequences via the mask-and-fill operation. Second, we propose to adopt curriculum learning to schedule the augmented sequences by measuring their difficulty levels. Finally, we design a contrastive learning objective to enhance user representations in an easy-to-difficult order of scheduled courses (*i.e.,* augmented sequences).

4.1 Context-aware Data Augmentation

As a major implementation way, contrastive learning usually relies on data augmentation to derive different views of the original data [3]. Previous approaches utilize heuristic strategies to augment data such as random item mask [40] or random crop [42]. However, these heuristic strategies are likely to generate out-of-distribution instances and may dramatically change the underlying semantics of the original data [4, 38]. To solve these problems, we propose a context-aware data augmentation strategy to produce high-quality augmented sequences adhereing to the attributes of users.

4.1.1 Attribute-enhanced Data Generator. Different from other data augmentation tasks, our focus is to improve user representations based on the sequential behaviors, thus users' attributes are important to consider in our data augmentation strategies. For this purpose, we firstly develop and train an attribute-enhanced data generator based on the Transformer architecture in Section 3.2, which can produce the augmented sequences retaining the attribute semantics of users. The base model in Section 3.2.1 has involved user attributes in the embedding layer, and we further propose a pre-training strategy inspired by masked language model [31] to fuse attribute information with sequential semantics. Given an item sequence $s_u = \{i_1, \cdots, i_k, \cdots, i_n\}$ and the associated attributes $\mathcal{A}_u$ for user u, we randomly mask a proportion of items in the sequence and then train the augmentation model by recovering the original sequence based on its contextual information. For example, if we mask the item i_k, we replace it with a special token "[*mask*]" and denote the masked item sequence as $s_{u,\neg k} = \{i_1, \cdots, [mask], \cdots, i_n\}$.

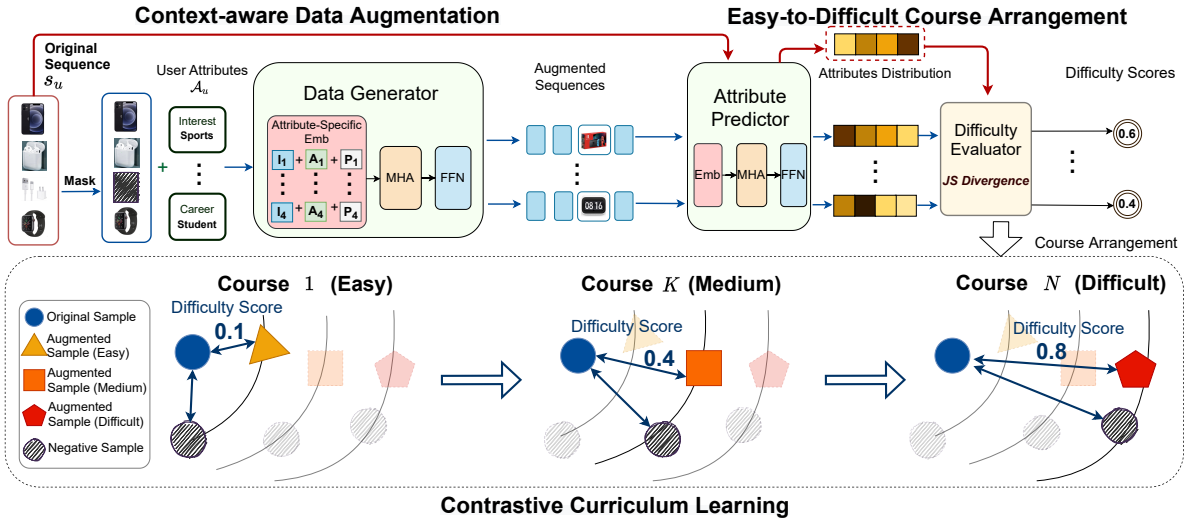


Figure 1: The overall architecture of our proposed approach. The distance of the blue left-right arrow represents the difficulty score between the original sample and the augmented samples at the lower part.

For the masked item i_k, we predict its original ID based on its contextual information, consisting of contextual items $s_{u,\neg k}$ and attribute information $\mathcal{A}_u$. Formally, following BERT [6], we leverage the bidirectional contextual information in the masked item sequence for predicting the masked item as:

$$\mathrm{P}(i_k|s_{u,\neg k},\mathcal{A}_u) = \sigma(\boldsymbol{f}_k^\top \mathbf{W}_1 \boldsymbol{e}_k), \tag{6}$$

where $\boldsymbol{f}_k$ is the representation at the k-th position using the bidirectional Transformer based on contextual information (Eq. 4), $\boldsymbol{e}_k$ is the embedding of the masked item i_k and $\sigma(.)$ is the sigmoid function to obtain the probability. Note that the above training strategy in Eq. 6 can be directly extended to mask a proportion of items. In this way, the base model is able to learn the correlations between the masked item and contextual information (*i.e.,* item sequence and user attribute), so that it can be used to generate instances confirming to the original attribute information of users.

4.1.2 Mask-and-Fill Augmentation. Based on the augmentation model, we propose a mask-and-fill strategy to conduct data augmentation, which is able to generate high-quality sequences retaining original attribute semantics of users. For each user interaction sequence s, we randomly mask a proportion γ of items, and let $\mathcal{M}$ denote all indices of the masked items. Formally, the mask operation can be formulated as:

$$s' = g_{\text{mask}}(s_u) = [i'_1, i'_2, \ldots, i'_n], \tag{7}$$

$$i'_t = \begin{cases} i_t \in s_u, & t \notin \mathcal{M}; \\ [\text{mask}], & t \in \mathcal{M}. \end{cases} \tag{8}$$

For each item i'_t in the interaction sequence s', we replace it with the special token "[*mask*]" if $t \in \mathcal{M}$ or keep as it is. Next, we obtain the augmented sequences by filling in the masked positions.

$$z = g_{\text{fill}}(s') = [i'_1, i'_2, \ldots, i'_n], \tag{9}$$

$$i'_t = \begin{cases} i_t \in s, & t \notin \mathcal{M}; \\ i \sim \mathrm{P}(\cdot|s_{u,\neg k},\mathcal{A}_u), & t \in \mathcal{M}. \end{cases} \tag{10}$$

Here, we utilize the pre-trained generator (Eq. 6) to fill in the masked position, and we sample an item according to the probability in $P(i|s_{u,\neg k},\mathcal{A}_u)$ (Eq. 6). Recall that the pre-training for the generator considers both sequential patterns and user's attributes. Our generator can retain important semantic information from original sequences. So, the augmented sequences are highly relevant to the original sequence but also provide different views for learning user representations.

Based on the above context-aware data generator, for each user, we can obtain a set of augmented sequences.

4.2 Easy-to-Difficult Course Arrangement

In contrastive learning [34], it relies on the contrastive difference or similarity to construct the learning objective between different views of a data instance. Intuitively, for two augmented sequences from the same sequence, the smaller difference there is, the easier it is to model the "*agreement*" on their representations. In other words, it will be less difficult to learn with more similar augmented sequences which have fewer modifications in semantics. Actually, it has been shown the schedule strategy of learning tasks in varied difficulty levels directly affect the performance of the learner [1, 27]. To quantitatively measure the difficulty levels in our task, we design a difficulty evaluator for augmented sequences based on the user attribute recovering task, and then apply it to schedule augmented sequences into courses from easy to difficult.

4.2.1 Difficulty Evaluator. To measure the difficulty level of each augmented instance, a straightforward method is to compute the surface similarity in sequences. However, user behaviors tend to be very complicated. Even for the same user, she/he might show different behavior patterns at different time. Considering this issue, we design the difficulty evaluator based on a novel task to recover user attributes. To be specific, we first encode an augmented sequence z into a d-dimensional embedding $\boldsymbol{v}_z$ with the base Transformer

architecture (Section 3.2), and then fed it into an attribute predictor according to a softmax layer:

$$\boldsymbol{\phi}_z = \Pr(\mathcal{A}_u|z) = \text{softmax}(\mathbf{W}_2^\top \cdot \boldsymbol{v}_z), \tag{11}$$

where $\mathbf{W}_2$ is a learnable matrix for softmax classification. Here, $\boldsymbol{\phi}_z \in \mathbb{R}^{|\mathcal{A}|}$ is a probability distribution over all the attributes, reflecting the confidence scores of the actual attributes in $\mathcal{A}_u$. If one augmented sequence is more capable of recovering the original user attributes, we assume that it is with a low difficulty level. To characterize such an idea, we represent the real attributes $\mathcal{A}_u$ as a one-hot vector, denoted by $\boldsymbol{\theta}_u$, where only the attribute dimensions in $\mathcal{A}_u$ are set to 1 and the rest entries are set to 0.

Formally, we use the Jensen-Shannon (JS) divergence between the probability distributions in original data and augmented data as the difficulty score:

$$d(z,u) = \frac{1}{2}KL(\boldsymbol{\theta}_u \| \frac{\boldsymbol{\phi}_z + \boldsymbol{\theta}_u}{2}) + \frac{1}{2}KL(\boldsymbol{\phi}_z \| \frac{\boldsymbol{\phi}_z + \boldsymbol{\theta}_u}{2}), \tag{12}$$

where $\boldsymbol{\phi}_z$ denotes the attribute distribution of the augmented data from Eq. 11, and $\boldsymbol{\theta}_u$ denotes the attribute distribution of the original data. The larger $d(z,u)$ is, the larger difficulty score of the augmented sequence is, which indicates the augmented sequence includes more semantic modifications.

4.2.2 Course Design. Based on the difficulty evaluator, we next distribute the augmented data into a set of courses according to their difficulty scores from easy to difficult. For each augmented data, we first calculate its difficulty score via the difficulty evaluator. Then, all the augmented sequences are sorted in an order of ascending difficulty scores. We divide the entire augmented data (with duplicate removal) into N bins consisting of equal-sized augmented sequences, where a smaller bin corresponds to smaller difficulty scores. To incorporate local stochastics, we further shuffle the augmented data within each bin. In our approach, we consider a bin of augmented data as a *course*. Based on these N courses, we can construct the learning curriculum with increasing difficulty levels. The model capacity will be gradually improved with such an elaborately designed curriculum.

4.3 Contrastive Curriculum Learning

Based on the generated courses, we propose a contrastive curriculum learning method to learn user representations via an easy-to-difficult process. Next, we first introduce the contrastive curriculum loss function, and then present the detailed learning process.

4.3.1 Contrastive Curriculum Loss Function. In order to learn user representations, we propose a contrastive curriculum learning method on the designed courses, via an easy-to-difficult process. Specifically, our model learns the above course one-by-one, in which we utilize a contrastive learning objective for minimizing the difference between augmented sequences and the original sequence given a user. Considering a mini-batch of samples in each learning stage, we regard original data and augmented data from the same user as *positive samples*, and treat other augmented examples within the same mini-batch as *negative samples*. We utilize the cosine similarity to measure the representation similarity:

$$\text{sim}(\boldsymbol{v}_u, \boldsymbol{v}_z) = \cos(\boldsymbol{v}_u, \boldsymbol{v}_z) = \frac{\boldsymbol{v}_u^\top \boldsymbol{v}_z}{\|\boldsymbol{v}_u\| \|\boldsymbol{v}_z\|}, \tag{13}$$

Algorithm 1 The overall learning process of our approach.

Input: User interaction sequence $\{s_u\}$ and user attribute $\{\mathcal{A}_u\}$.
Output: The learned user representations $\boldsymbol{v}_u$.
1: Pre-train the augmented data generator by Eq. 6.
2: **for** $j = 1 \rightarrow |\mathcal{U}|$ **do**
3: Randomly mask a proportion of items in s_u (Eq. 7).
4: Fill in the masked positions by the augmented data generator (Eq. 9).
5: **end for**
6: Pre-train the difficulty evaluator by Eq. 11.
7: Assign the difficulty score for all the augmented data $\{\mathcal{Z}_u\}$ by Eq. 12.
8: Divide $\{\mathcal{Z}_u\}$ into N curriculums based on difficulty scores.
9: Initialize the parameters in base model.
10: **for** $k = 1 \rightarrow N$ **do**
11: Train the base model with the k-th curriculum until convergence using the annealing contrastive loss as Eq. 14.
12: **end for**
13: Fine-tune the model according to downstream tasks.

where $\boldsymbol{v}_u$ denotes the representation learned with original sequence s_u for user u, and $\boldsymbol{v}_z$ denotes the representation for an augmented sequence z. Following [34], the loss function for a positive pair $(\boldsymbol{v}_u, \boldsymbol{v}_z)$ can be defined similar to the softmax cross-entropy loss:

$$\mathcal{L} = -\lambda * \log \frac{\exp(\text{sim}(\boldsymbol{v}_u \cdot \boldsymbol{v}_{z^+})/\tau)}{\exp(\text{sim}(\boldsymbol{v}_u \cdot \boldsymbol{v}_{z^+})/\tau) + \sum_{z^- \in \mathcal{N}} \exp(\text{sim}(\boldsymbol{v}_u \cdot \boldsymbol{v}_{z^-})/\tau)}, \tag{14}$$

where z^+ is a positive augmented sequence for user u and z^- is a negative augmented sequence for other users, $\mathcal{N}$ is a negative set of augmented sequences, and τ is a hyper-parameter for softmax temperature. To avoid forgetting easy instances and overfitting to difficult ones, we add the annealing mechanism to adjust the contrastive objective: $\lambda = \rho * \delta^k$, where ρ is the initial weight, δ denotes the decay ratio, and k denotes the current curriculum number. In this way, as the learned courses becomes difficult, the importance of the scheduled courses is also reduced.

Since our model learns with the augmented instances from easy to difficult, the user representations can be gradually improved. Finally, after pre-training, we fine-tune the learned user representations $\boldsymbol{v}_u$ in various downstream tasks.

4.3.2 Learning process. Algorithm 1 presents the training algorithm for our model. The entire procedure of our approach consists of four important stages, namely context-aware data augmentation, course design, contrastive curriculum learning and fine-tuning stages. At the context-aware data augmentation stage (line 1), we adopt a context-aware data generator to produce a set of augmented instances, which first masks a proportion of items in the original user interaction sequence and then samples "close-to-real" items based on the predicted probability distribution (Eq. 6). At the course design stage (line 6-8), we utilize a difficulty evaluator to produce the difficulty scores of all the augmented instances. Then we rank them based on scores and divide them into N bins, where each bin corresponds to a curriculum. At the contrastive curriculum learning stage (line 9-12), we pre-train our base model with the above curriculums via an annealing contrastive learning objective (Eq. 14). Since our model learns from easy curriculums to difficult ones, the user representations can be gradually enhanced. Finally, we fine-tune the user representations in downstream applications.

4.4 Discussion and Analysis

Next, we discuss our framework, and further analyze time complexity in details.

4.4.1 Discussion on Overall Framework. The basic idea of our approach is to generate high-quality augmented data for contrastive curriculum learning. We do not simply generate augmented sequences by heuristic strategies [42, 45]. Instead, we train a context-aware data generator that is able to generate sequences conforming to users' attribute information. Such a model-based augmentation approach has seldom been considered in user behavior modeling, while it is particularly suitable for user-oriented applications [39]. Besides, recent studies have adopted contrastive learning [42, 49] for user behavior modeling. As the primary difference, our approach schedules the augmented data in an easy-to-difficult process with curriculum contrastive learning. As shown by Hacohen et al. [10], curriculum learning leverages a Bayesian prior for data sampling. It adds a covariance term for regularization, which encourages the hyper-parameters to better converge to the optimal point of the contrastive loss. Therefore, the contrastive curriculum learning approach can further enhance the user representation. To our knowledge, it is the first time that curriculum learning and contrastive learning are integrated for improving sequential user behavior modeling.

4.4.2 Time Complexity. Once the user representation $\boldsymbol{v}_u$ (in Eq. 14) has been learned, the online inference time is similar to other representation-based methods [51]. Here, we mainly discuss the offline time cost of model training, including data augmentation, course arrangement and contrastive curriculum learning. For data augmentation, it roughly takes a time of $O((m+a)(d^2))$ to generate a augmented sequences for d-length sequence from m original data. For course arrangement, the time cost mainly lies in the sort and partition of all augmented data. If the score range can be pre-defined, we can use the bucket sort algorithm [20] to accelerate the process, which costs $O(a)$. For contrastive curriculum learning, we iterate all the N bins containing augmented sequences together with original sequences. Since the number of instances in each bin is a/N, the corresponding time complexity is $O(k * a)$, where k denotes the average training epoch until convergence. The total time complexity is $O((a(d^2 + k + 1) + md^2)$. Actually, the learning within each course can be largely efficiently parallelized [5]. Meanwhile, when handling long sequences, we can further adopt locality sensitive hashing [19] to accelerate the procedure of attention calculation.

5 EXPERIMENTS

In this section, we first set up the experiments, and then present the results and analysis.

5.1 Experimental Setup

5.1.1 Datasets. We conduct experiments on six datasets collected from four real-world platforms with various domains and sparsity levels. The statistics of these datasets after preprocessing are summarized in Table 1.

(1) **QQBrowser, Kuaibao**: we collect the anonymous behavior data of users from two news apps in Tencent[1], namely *Kuaibao* and *QQBrowser*. The data period spans a given month. In the two apps, news articles are considered as *items*, and user authorized registration information (*i.e.,* hobbies and city) is considered as *user attribute*. Each interaction record is one click on a news article by a user on an app (with a timestamp). The research process is tightly regulated for avoiding any disclosure of user privacy.

(2) **Amazon Beauty, Sports, and Toys**: these three datasets are obtained from Amazon review datasets in [23]. In this work, we select three subcategories: "Beauty", "Sports and Outdoors", and "Toys and Games", and utilize the brands of the goods as attributes.

(3) **Yelp**[2]: this is a dataset for business recommendation. As it is very large, we use the transaction records after *January 1st, 2019*.

For all datasets, we group the interaction records by users and sort them by the interaction timestamps ascendingly. Following [49], we only keep the 5-core datasets, and filter out unpopular items and inactive users with fewer than five interaction records. It should be noted that in these datasets, Yelp and Amazon are public datasets, while QQBrowser and Kuaibao are industry datasets.

Table 1: Statistics of the datasets after preprocessing. "QQB" and "Acts" denote QQBrowser and Actions respectively.

Dataset	Beauty	Sports	Toys	Yelp	QQB	Kuaibao
# Users	22,363	25,598	19,412	30,431	18,405	19,863
# Items	12,101	18,357	11,924	20,033	76,269	92,607
# Avg. Acts / User	8.9	8.3	8.6	10.4	27.8	32.3
# Avg. Acts / Item	16.4	16.1	14.1	15.8	6.7	6.9
# Actions	747,827	296,337	167,597	316,354	511,664	642,312
Sparsity	99.93%	99.95%	99.93%	99.95%	99.97%	99.97%
# Attributes	1,221	2,277	1,027	1,001	620	695

5.1.2 Evaluation Tasks. We conduct the experiments on three downstream tasks, namely sequential recommendation, user attribute prediction and user activity prediction. The task of sequential recommendation is that given the historical behaviors and attributes of a user, we need to predict the next item that the user is likely to interact with at the next step. User attribute prediction is a classification task, in which we aim to predict the profession domain of a user. We consider 15 first-level profession domains (*e.g.,* education, IT and finance) derived from the authorized registration information from apps. For user activity prediction, we aim to infer the user activity level in the next week. We treat it as a classification task with four different activity levels, including *high* (7 days), *medium* (4-6 days), *low* (1-3 days) and *churn* according to the total number of logging days by a user on an app.

5.1.3 Evaluation Settings. For *user activity prediction* and *user attribute prediction*, we adopt the evaluation metrics including AUC, Accuracy, Precision, Recall and F1. we split the dataset into train, validation, and test sets with an approximate ratio of 8:1:1. For *sequential recommendation*, we employ top-k Hit Ratio (HR@k), top-k Normalized Discounted Cumulative Gain (NDCG@k) to evaluate the performance, which are widely used in related works [2, 46]. We report results on HR@$\{5, 10\}$ and NDCG@$\{5, 10\}$. Following

[1] https://www.tencent.com

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

previous works [2, 47], we apply the *leave-one-out strategy* for evaluation. 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.

5.1.4 Comparison Methods. We compare our proposed approach with the following ten baseline methods:

(1) **GRU4Rec** [13] applies GRU to model user click sequence for session-based recommendation. We represent the items using embedding vectors rather than one-hot vectors;

(2) **Caser** [32] is a CNN-based method capturing high-order Markov Chains by applying horizontal and vertical convolutional operations for sequential recommendation;

(3) **SASRec** [17] is a self-attention based sequential recommendation model, which uses the multi-head attention mechanism to recommend the next item;

(4) **BERT4Rec** [31] uses a Cloze objective loss for sequential recommendation by the bidirectional self-attention mechanism;

(5) **GAT** [36] is a homogeneous graph neural network which considers the attention mechanism on the user-item bipartite graph;

(6) **SR-GNN** [41] models per-session records with GCN and then construct the sequential model over session representations. In our work, we consider a day as a session;

(7) **VirtualTB** [28] proposes GAN-SD for user feature generation with matched distribution and Multi-agent Adversarial Imitation Learning for generating better generalizable interaction data;

(8) **Mixup** [45] proposes a generic vicinal distribution and samples from mixup vicinal distribution producing virtual vectors;

(9) **EDA** [40] consists of three simple data augmentation operations: random insertion, random swap, and random deletion;

(10) **CP4Rec** [42] utilizes the contrastive pre-training framework to extract meaningful user patterns and proposes three data augmentation approaches to construct pre-training tasks.

We reproduce most of the baseline methods with the open source library RecBole [48], and implement those that are not in RecBole and our approach in PyTorch [3]. The dimensionality of embeddings (including items and apps) is set to 200. We set the number of Transformer layers is set as 2. The batch size is set to 256. We use Adam [18] optimization with its default parameter setting. Early stopping is used with a patience of 5 epochs. The gradient clipping restricts the norm of gradients within $[0, 0.1]$. Our code is available at this link: https://github.com/RUCAIBox/Contrastive-Curriculum-Learning.

5.2 Experimental Results

In this section, we compare the proposed method with several baselines methods in different tasks on both public datasets and industry datasets.

In Table 2, we report the performance of different methods in sequential recommendation on public datasets. Among sequential recommendation baseline methods, SASRec and BERT4Rec utilize the self-attention mechanism respectively, and achieve better performance than GRU4Rec and Caser. It indicates that self-attentive architecture is particularly suitable for modeling sequential data. However, their improvements are not stable when training with the conventional next-item prediction loss. Within four data augmentation based methods, VTB uses generative methods to augment data, Mixup and EDA use heuristic methods to rewrite and augment data. For sequential data, CP4Rec achieves the best results on most datasets and tasks, since CP4Rec uses heuristic methods and considers contrastive learning to enhance the representations. Finally, by comparing our approach CCL with all the baselines, it is clear to see that our method performs consistently better than them by a large margin on all datasets. Different from these baselines, we adopt an attribute-enhanced data generator to produce high-quality instances for data augmentation, and utilize contrastive curriculum learning strategy to model user behavior via an easy-to-difficult learning process. This result shows that our approach is effective to improve the performance for sequence modeling.

Since industry datasets usually contain more user portraits and online evaluation indicators of users, we can use sequential user behavior data for more downstream tasks. In Table 3, we report the performance of different methods on three tasks (sequential recommendation, user attribute prediction and user activity prediction) on industry datasets. Among non-sequential baseline methods, in user attribute prediction and user activity prediction, SR-GNN achieves relatively better performance than GAT, which captures sequential behavior characteristics to some extent. In general, non-sequential recommendation methods perform worse than sequential recommendation methods, since the sequential pattern is important to our task. Besides, the experimental results of other baselines are similar to those on public datasets. Our proposed framework performs consistently better than all the baselines under different metrics. It indicates that our proposed method is able to learn effective user representations, and is suitable for various downstream tasks.

5.3 Further Analysis

Next, we continue to study whether our approach works well in more detailed analysis.

5.3.1 Ablation Study. To effectively utilize the user behavior data, our approach has made several technical extensions. Here, we examine how each of them affects the final performance. We consider the following variants of our approach for comparison:

- $\text{CCL}_{\neg\text{Aug}}$: Removing the data augmentation module.
- $\text{CCL}_{\neg\text{Attr}}$: Removing the attribute embeddings in the data augmentation module.
- $\text{CCL}_{\neg\text{Curriculum}}$: Removing the curriculum learning module.
- $\text{CCL}_{\neg\text{Annealing}}$: Removing the annealing methods in the curriculum learning module.
- $\text{CCL}_{\text{Reorder}}$: Arranging the curriculums randomly.

In Table 4, we report the results of these comparison methods in the sequential recommendation task on Beauty dataset. Similar conclusions can be drawn on the other datasets or tasks. As can see, the variants derived from removing operation perform worse than our proposed CCL method, it indicates that the proposed techniques are useful to improve the performance. Among them, $\text{CCL}_{\neg\text{Aug}}$ performs worst, it shows that our proposed context-aware data augmentation method is more important for our approach. Besides, our model outperforms $\text{CCL}_{\text{Reorder}}$. It is because the easy-to-difficult learning strategy is more helpful to improve the user behavior modeling.

[3] https://pytorch.org/

Table 2: Performance comparisons of different methods for sequential recommendation task on public datasets. The best performance and the second best performance methods are denoted in bold and underlined fonts respectively. "$*$" indicates the statistical significance for $p < 0.01$ compared to the best baseline method.

Datasets	Metric	GRU4Rec	Caser	SASRec	BERT4Rec	VTB	Mixup	EDA	CP4Rec	CCL
Beauty	HR@5	0.3125	0.3032	0.3741	0.3640	0.3656	0.3802	0.3893	0.3932	**0.4156***
	NDCG@5	0.2268	0.2219	0.2848	0.2622	0.2641	0.2813	0.2889	0.2904	**0.3125***
	HR@10	0.4106	0.3942	0.4696	0.4739	0.4744	0.4920	0.4975	0.4995	**0.5234***
	NDCG@10	0.2584	0.2512	0.3156	0.2975	0.2995	0.3188	0.3231	0.3247	**0.3466***
Sports	HR@5	0.3055	0.2866	0.3466	0.3375	0.3382	0.3567	0.3634	0.3658	**0.3879***
	NDCG@5	0.2126	0.2020	0.2497	0.2341	0.2352	0.2456	0.2534	0.2556	**0.2723***
	HR@10	0.4299	0.4014	0.4622	0.4722	0.4725	0.4603	0.4678	0.4695	**0.4903***
	NDCG@10	0.2527	0.2390	0.2869	0.2775	0.2783	0.2867	0.2922	0.2943	**0.3178***
Toys	HR@5	0.2795	0.2614	0.3682	0.3344	0.3348	0.3887	0.3923	0.3942	**0.4153***
	NDCG@5	0.1919	0.1885	0.2820	0.2327	0.2330	0.2790	0.2864	0.2876	**0.2971***
	HR@10	0.3896	0.3540	0.4663	0.4493	0.4502	0.4811	0.4867	0.4889	**0.5102***
	NDCG@10	0.2274	0.2183	0.3136	0.2698	0.2705	0.3147	0.3202	0.3212	**0.3365***
Yelp	HR@5	0.5437	0.5111	0.5745	0.5976	0.5985	0.5923	0.5997	0.5956	**0.6203***
	NDCG@5	0.3784	0.3696	0.4113	0.4252	0.4254	0.4281	0.4337	0.4203	**0.4404***
	HR@10	0.7265	0.6661	0.7373	0.7597	0.7603	0.7587	0.7621	0.7639	**0.7832***
	NDCG@10	0.4375	0.4198	0.4642	0.4778	0.4782	0.4729	0.4756	0.4735	**0.4912***

Table 3: Performance comparisons of different methods for three tasks on industry datasets. The best performance and the second best performance methods are denoted in bold and underlined fonts respectively. "$*$" indicates the statistical significance for $p < 0.01$ compared to the best baseline method.

Task (Datasets)	Metric	GAT	SR-GNN	GRU4Rec	Caser	SASRec	BERT4Rec	VTB	Mixup	EDA	CP4Rec	CCL
Sequential Recommendation (QQBrowser)	HR@5	0.2267	0.2353	0.2517	0.2982	0.3385	0.3073	0.2872	0.3202	0.3519	0.3424	**0.3623***
	NDCG@5	0.1645	0.1703	0.1728	0.1960	0.2330	0.1766	0.1896	0.2301	0.2113	0.2409	**0.2656***
	HR@10	0.3589	0.3685	0.3917	0.4431	0.4706	0.4055	0.4193	0.4670	0.4569	0.4991	**0.5135***
	NDCG@10	0.2071	0.2122	0.2150	0.2428	0.2755	0.2225	0.2324	0.2775	0.2594	0.2871	**0.3083***
Sequential Recommendation (Kuaibao)	HR@5	0.2662	0.3218	0.3382	0.3812	0.4524	0.3985	0.4110	0.3799	0.4386	0.4595	**0.4725***
	NDCG@5	0.2039	0.2170	0.2303	0.2619	0.3207	0.2713	0.2887	0.2639	0.3098	0.3236	**0.3427***
	HR@10	0.4377	0.4709	0.4881	0.5267	0.6053	0.5514	0.5573	0.5378	0.5962	0.6164	**0.6368***
	NDCG@10	0.2394	0.2651	0.2787	0.3090	0.3700	0.3208	0.3359	0.3149	0.3607	0.3743	**0.3994***
User Activity Prediction (QQBrowser)	ACC	0.6665	0.6624	0.6574	0.6693	0.6712	0.6699	0.6734	0.6741	0.6757	0.6778	**0.6845***
	Precsion	0.6726	0.6593	0.6662	0.6785	0.6793	0.6782	0.6802	0.6811	0.6823	0.6840	**0.6896***
	Recall	0.6632	0.6540	0.6562	0.6696	0.6723	0.6717	0.6745	0.6753	0.6758	0.6773	**0.6843***
	F1	0.6665	0.6566	0.6584	0.6726	0.6755	0.6762	0.6774	0.6798	0.6805	0.6815	**0.6880***
User Activity Prediction (Kuaibao)	ACC	0.6525	0.6482	0.6391	0.6502	0.6518	0.6511	0.6534	0.6545	0.6547	0.6556	**0.6689***
	Precsion	0.6578	0.6522	0.6436	0.6590	0.6603	0.6617	0.6634	0.6647	0.6654	0.6662	**0.6768***
	Recall	0.6493	0.6475	0.6383	0.6517	0.6521	0.6525	0.6578	0.6612	0.6623	0.6636	**0.6743***
	F1	0.6535	0.6493	0.6412	0.6579	0.6594	0.6604	0.6613	0.6625	0.6632	0.6639	**0.6752***
User Attribute Prediction (QQBrowser)	ACC	0.6554	0.6772	0.6945	0.7005	0.7102	0.7135	0.7147	0.7156	0.7255	0.7298	**0.7401***
	Precsion	0.6675	0.6663	0.7052	0.7112	0.7137	0.7164	0.7221	0.7243	0.7260	0.7282	**0.7367***
	Recall	0.6601	0.6725	0.6822	0.6904	0.6915	0.6934	0.6967	0.7013	0.7115	0.7129	**0.7221***
	F1	0.6685	0.6901	0.6956	0.6980	0.7012	0.7067	0.7128	0.7156	0.7190	0.7225	**0.7315***
User Attribute Prediction (Kuaibao)	ACC	0.6520	0.6482	0.6391	0.6567	0.6612	0.6635	0.6678	0.6723	0.6789	0.6802	**0.6912***
	Precsion	0.6573	0.6525	0.6437	0.6621	0.6645	0.6668	0.6690	0.6721	0.6745	0.6713	**0.6855***
	Recall	0.6492	0.6475	0.6386	0.6567	0.6612	0.6624	0.6682	0.6711	0.6756	0.6725	**0.6875***
	F1	0.6531	0.6494	0.6417	0.6610	0.6632	0.6656	0.6678	0.6718	0.6750	0.6719	**0.6865***

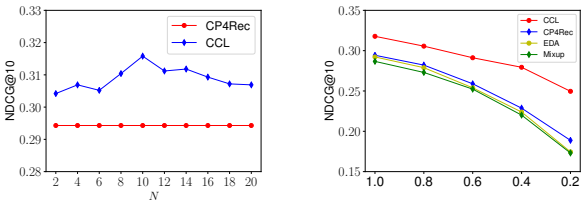
5.3.2 Parameter Tuning. In this part, we examine the robustness of our model, and analyze the influence of hyperparameters and the sparsity of training data on model performance. Course design is a key step of our approach, in which we set the number of courses N for contrastive curriculum learning. To examine the influence of different course number, we vary the course number from 2 to 20 incremented by 2 and report the corresponding performance on Sports dataset for sequential recommendation task. As shown in Figure 2(a), our approach outperforms the baseline under all settings. It indicates that our approach is very robust with the settings of

Table 4: Ablation analysis on Beauty dataset (sequential recommendation).

Models	HR@5	NDCG@5	HR@10	NDCG@10
$CCL_{\neg Aug}$	0.3796	0.2855	0.4898	0.3136
$CCL_{\neg Attr}$	0.4060	0.3035	0.5152	0.3389
$CCL_{\neg Curriculum}$	0.3937	0.2915	0.5014	0.3277
$CCL_{\neg Annealing}$	0.4044	0.3035	0.5126	0.3401
$CCL_{Reorder}$	0.3998	0.3002	0.5085	0.3354
CCL	**0.4156**	**0.3125**	**0.5234**	**0.3466**

course number. Besides, when N is set to 10, our model achieves the best performance. It demonstrates that a proper number of courses is promising for user behavior modeling and the proposed approach can better utilize the augmented data to enhance user representation.

We simulate the data sparsity scenarios by using different proportions of the full dataset, *i.e.,* 20%, 40%, 60%, 80%, and 100%. Figure 2(b) shows the evaluation results of sequential recommendation task on Sports dataset. The performance substantially drops when less training data is used. While, the proposed contrastive curriculum learning approach is consistently better than baselines in all cases, especially in an extreme sparsity level (20%). This observation implies that the contrastive curriculum learning framework is able to make better use of the augmented data, which alleviates the influence of data sparsity problem for user behavior modeling to some extent.



(a) The number of courses. (b) The ratio of training data.

Figure 2: Performance tuning with different number of courses and different sparsity levels on the Sports dataset.

5.4 Online A/B Test

To further examine the effectiveness of our approach, we conduct online A/B tests on the Click Through Rate (CTR) metrics in real application scenario of the QQ browser app. Specifically, we select four real business scenarios from File Search, File Download, Video Player and News Search. These are four functional modules with high user traffic in the QQ browser app. Among them, File Search module is for the search of local files, File Download module is for the download of files on the Internet, Video Player module is to play videos on the Internet, and News Search module is to look for news that users are interested in.

Then, we randomly split them into control group (A) and treatment group (B) with the same size. For comparison, we apply our model to generate augmented data and learn user representations. Then, we adopt the same strategy for the selected users of two groups. Finally, we compute CTR for both groups, defined as *CTR= (Number of Clicks / Number of Impressions)*, CTR is a commonly-used metric to measure the success of an online advertising campaign. A group with a larger CTR value indicates that the corresponding algorithm is better in capturing user interests in the cold start stage and improving user retention with a high probability in various application scenarios.

Table 5: Comparison of CTR metrics in online A/B test. Among them, File Tab and File Download each have 500,000 new users, and Video Player and News Search each have 1,000,000 new users.

Models	File Search	File Download	Video Player	News Search
Origin	0.0594	0.0636	0.0640	0.0678
Origin+CCL	**0.0650**	**0.0650**	**0.0676**	**0.0696**
Improvement	+2.6%	+2.2%	+4.7%	+2.7%

Here, we consider a 7-day period of the QQ browser app for online A/B tests. Table 5 presents the CTR comparison between the original method and our method. As we can see, our method is consistently better than the compared baseline. Meanwhile, the time cost of our method is similar to the original method with the same order of magnitude in the real-time environment. Our online A/B test further demonstrates the effectiveness of the proposed method.

6 CONCLUSION

In this paper, we present a novel curriculum contrastive learning framework for effectively modeling the sequential user behavior data. Our contributions can be summarized in two points. First, we designed a context-aware data augmentation approach to produce the augmented sequences confirming to users' attribute information. Second, we proposed a curriculum learning strategy to conduct contrastive learning via an easy-to-difficult learning process. Experimental results on both public and industry datasets have demonstrated the effectiveness of our approach.

It is worth noting that the contrastive curriculum learning method can be easily adapted to other tasks. As future work, we will consider extending the current framework for data modeling in other kinds of applications.

ACKNOWLEDGEMENT

This work was partially supported by the National Natural Science Foundation of China under Grant No. 61872369 and 61832017, Beijing Academy of Artificial Intelligence (BAAI) under Grant No. BAAI2020ZJ0301, Beijing Outstanding Young Scientist Program under Grant No. BJJWZYJH012019100020098, the Fundamental Research Funds for the Central Universities, the Research Funds of Renmin University of China under Grant No. 18XNLG22 and 19XNQ047, and the Outstanding Innovative Talents Cultivation Funded Programs 2020 of Renmin Univertity of China. Xin Zhao is the corresponding author.

REFERENCES

[1] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *ICML 2009*. 41–48.

[2] Shuqing Bian, Wayne Xin Zhao, Kun Zhou, Xu Chen, Jing Cai, Yancheng He, Xingji Luo, and Ji-Rong Wen. 2021. A Novel Macro-Micro Fusion Network for User Representation Learning on Mobile Apps. In *WWW 2021*. 3199–3209.

[3] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. 2020. A Simple Framework for Contrastive Learning of Visual Representations. In *ICML 2020*. 1597–1607.

[4] Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators. In *ICLR 2020*.

[5] Jeffrey Dean, Greg Corrado, Rajat Monga, Kai Chen, Matthieu Devin, Quoc V. Le, Mark Z. Mao, Marc'Aurelio Ranzato, Andrew W. Senior, Paul A. Tucker, Ke Yang, and Andrew Y. Ng. 2012. Large Scale Distributed Deep Networks. In *NIPS 2012*. 1232–1240.

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL-HLT 2019*. 4171–4186.

[7] Tim Donkers, Benedikt Loepp, and Jürgen Ziegler. 2017. Sequential User-based Recurrent Neural Network Recommendations. In *RecSys 2017*. 152–160.

[8] Bin Fu, Jialiu Lin, Lei Li, Christos Faloutsos, Jason I. Hong, and Norman M. Sadeh. 2013. Why people hate your app: making sense of user feedback in a mobile app store. In *KDD 2013*. 1276–1284.

[9] Michael Gutmann and Aapo Hyvärinen. 2012. Noise-Contrastive Estimation of Unnormalized Statistical Models, with Applications to Natural Image Statistics. *J. Mach. Learn. Res.* 13 (2012), 307–361.

[10] Guy Hacohen and Daphna Weinshall. 2019. On The Power of Curriculum Learning in Training Deep Networks. In *ICML 2019*.

[11] Ruining He and Julian J. McAuley. 2016. Fusing Similarity Models with Markov Chains for Sparse Sequential Recommendation. In *ICDM 2016*. 191–200.

[12] Xiangnan He and Tat-Seng Chua. 2017. Neural Factorization Machines for Sparse Predictive Analytics. In *SIGIR 2017*. 355–364.

[13] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2016. Session-based Recommendations with Recurrent Neural Networks. In *ICLR 2016*.

[14] Jin Huang, Wayne Xin Zhao, Hongjian Dou, Ji-Rong Wen, and Edward Y. Chang. 2018. Improving Sequential Recommendation with Knowledge-Enhanced Memory Networks. In *SIGIR 2018*. 505–514.

[15] Lu Jiang, Deyu Meng, Qian Zhao, Shiguang Shan, and Alexander G. Hauptmann. 2015. Self-Paced Curriculum Learning. In *AAAI 2015*. 2694–2700.

[16] Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. 2018. MentorNet: Learning Data-Driven Curriculum for Very Deep Neural Networks on Corrupted Labels. In *ICML 2018*. 2309–2318.

[17] Wang-Cheng Kang and Julian J. McAuley. 2018. Self-Attentive Sequential Recommendation. In *ICDM 2018*. 197–206.

[18] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *ICLR 2015*.

[19] Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. 2020. Reformer: The Efficient Transformer. In *ICLR 2020*.

[20] Fang Liu, Meng-Cheng Huang, Xuehui Liu, and Enhua Wu. 2009. Efficient depth peeling via bucket sort. In *ACM SIGGRAPH/EUROGRAPHICS 2009*. 51–57.

[21] Lajanugen Logeswaran and Honglak Lee. 2018. An efficient framework for learning sentence representations. In *ICLR 2018*.

[22] Jiasen Lu, Jianwei Yang, Dhruv Batra, and Devi Parikh. 2016. Hierarchical Question-Image Co-Attention for Visual Question Answering. In *NeurIPS 2016*. 289–297.

[23] Julian J. McAuley, Christopher Targett, Qinfeng Shi, and Anton van den Hengel. 2015. Image-Based Recommendations on Styles and Substitutes. In *SIGIR 2015*. 43–52.

[24] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. In *NeurIPS 2013*. 3111–3119.

[25] Emmanouil Antonios Platanios, Otilia Stretcu, Graham Neubig, Barnabás Póczos, and Tom M. Mitchell. 2019. Competence-based Curriculum Learning for Neural Machine Translation. In *NAACL-HLT 2019*. 1162–1172.

[26] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing personalized Markov chains for next-basket recommendation. In *WWW 2010*. 811–820.

[27] Mrinmaya Sachan and Eric P. Xing. 2016. Easy Questions First? A Case Study on Curriculum Learning for Question Answering. In *ACL 2016*.

[28] Jing-Cheng Shi, Yang Yu, Qing Da, Shi-Yong Chen, and Anxiang Zeng. 2019. Virtual-Taobao: Virtualizing Real-World Online Retail Environment for Reinforcement Learning. In *AAAI 2019*. 4902–4909.

[29] Ajit Paul Singh and Geoffrey J. Gordon. 2008. Relational learning via collective matrix factorization. In *SIGKDD 2008*. 650–658.

[30] Weiping Song, Chence Shi, Zhiping Xiao, Zhijian Duan, Yewen Xu, Ming Zhang, and Jian Tang. 2019. AutoInt: Automatic Feature Interaction Learning via Self-Attentive Neural Networks. In *CIKM 2019*. 1161–1170.

[31] 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 *CIKM 2019*. 1441–1450.

[32] Jiaxi Tang and Ke Wang. 2018. Personalized Top-N Sequential Recommendation via Convolutional Sequence Embedding. In *WSDM 2018*. 565–573.

[33] Yi Tay, Shuohang Wang, Anh Tuan Luu, Jie Fu, Minh C. Phan, Xingdi Yuan, Jinfeng Rao, Siu Cheung Hui, and Aston Zhang. 2019. Simple and Effective Curriculum Pointer-Generator Networks for Reading Comprehension over Long Narratives. In *ACL 2019*. 4922–4931.

[34] Aäron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation Learning with Contrastive Predictive Coding. *CoRR* abs/1807.03748 (2018).

[35] 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 *NeurIPS 2017*. 5998–6008.

[36] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. In *ICLR 2018*.

[37] Petar Velickovic, William Fedus, William L. Hamilton, Pietro Liò, Yoshua Bengio, and R. Devon Hjelm. 2019. Deep Graph Infomax. In *ICLR 2019*.

[38] Jun Wang, Lantao Yu, Weinan Zhang, Yu Gong, Yinghui Xu, Benyou Wang, Peng Zhang, and Dell Zhang. 2017. IRGAN: A Minimax Game for Unifying Generative and Discriminative Information Retrieval Models. In *SIGIR 2017*. 515–524.

[39] Qinyong Wang, Hongzhi Yin, Hao Wang, Quoc Viet Hung Nguyen, Zi Huang, and Lizhen Cui. 2019. Enhancing Collaborative Filtering with Generative Augmentation. In *KDD 2019*. 548–556.

[40] Jason W. Wei and Kai Zou. 2019. EDA: Easy Data Augmentation Techniques for Boosting Performance on Text Classification Tasks. In *EMNLP-IJCNLP 2019*. 6381–6387.

[41] Shu Wu, Yuyuan Tang, Yanqiao Zhu, Liang Wang, Xing Xie, and Tieniu Tan. 2019. Session-Based Recommendation with Graph Neural Networks. In *AAAI 2019*. 346–353.

[42] Xu Xie, Fei Sun, Zhaoyang Liu, Jinyang Gao, Bolin Ding, and Bin Cui. 2020. Contrastive Pre-training for Sequential Recommendation. *CoRR* abs/2010.14395 (2020).

[43] Yi-Ting Yeh and Yun-Nung Chen. 2019. QAInfomax: Learning Robust Question Answering System by Mutual Information Maximization. In *EMNLP-IJCNLP 2019*. 3368–3373.

[44] Feng Yu, Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. 2016. A Dynamic Recurrent Model for Next Basket Recommendation. In *SIGIR 2016*. 729–732.

[45] Hongyi Zhang, Moustapha Cissé, Yann N. Dauphin, and David Lopez-Paz. 2018. mixup: Beyond Empirical Risk Minimization. In *ICLR 2018*.

[46] Tingting Zhang, Pengpeng Zhao, Yanchi Liu, Victor S. Sheng, Jiajie Xu, Deqing Wang, Guanfeng Liu, and Xiaofang Zhou. 2019. Feature-level Deeper Self-Attention Network for Sequential Recommendation. In *IJCAI 2019*. 4320–4326.

[47] Wayne Xin Zhao, Junhua Chen, Pengfei Wang, Qi Gu, and Ji-Rong Wen. 2020. Revisiting Alternative Experimental Settings for Evaluating Top-N Item Recommendation Algorithms. In *CIKM 2020*. 2329–2332.

[48] Wayne Xin Zhao, Shanlei Mu, Yupeng Hou, Zihan Lin, Kaiyuan Li, Yushuo Chen, Yujie Lu, Hui Wang, Changxin Tian, Xingyu Pan, Yingqian Min, Zhichao Feng, Xinyan Fan, Xu Chen, Pengfei Wang, Wendi Ji, Yaliang Li, Xiaoling Wang, and Ji-Rong Wen. 2021. RecBole: Towards a Unified, Comprehensive and Efficient Framework for Recommendation Algorithms. In *CIKM 2021*.

[49] Kun Zhou, Hui Wang, Wayne Xin Zhao, Yutao Zhu, Sirui Wang, Fuzheng Zhang, Zhongyuan Wang, and Ji-Rong Wen. 2020. S3-Rec: Self-Supervised Learning for Sequential Recommendation with Mutual Information Maximization. In *CIKM 2020*. 1893–1902.

[50] Kun Zhou, Wayne Xin Zhao, Shuqing Bian, Yuanhang Zhou, Ji-Rong Wen, and Jingsong Yu. 2020. Improving Conversational Recommender Systems via Knowledge Graph based Semantic Fusion. In *KDD 2020*. 1006–1014.

[51] Meizi Zhou, Zhuoye Ding, Jiliang Tang, and Dawei Yin. 2018. Micro Behaviors: A New Perspective in E-commerce Recommender Systems. In *WSDM 2018*. 727–735.

[52] Hengshu Zhu, Hui Xiong, Yong Ge, and Enhong Chen. 2014. Mobile app recommendations with security and privacy awareness. In *KDD 2014*. 951–960.

[53] Yin Zhu, Erheng Zhong, Sinno Jialin Pan, Xiao Wang, Minzhe Zhou, and Qiang Yang. 2013. Predicting user activity level in social networks. In *CIKM 2013*. 159–168.