Computer Science and Information Systems 22(1):345–368 https://doi.org/10.2298/CSIS231101013L

Improved Session Recommendation Using Contrastive Learning based Tail Adjusted Repeat Aware Graph Neural Network

Daifeng Li[1], Tianjunzi Tian[2], Zhaohui Huang[1], Xiaowen Lin[1,⋆], Dingquan Chen[1,⋆], and Andrew Madden[3]

[1] School of Information Management, Sun Yat-sen University, 51006 Guangzhou, China
lidaifeng@mail.sysu.edu.cn
huangzhaohui27@mail2.sysu.edu.cn
linxw26@mail2.sysu.edu.cn
gzchendq@163.com
[2] Department of Information Management, Nanjing University, 210023 Nanjing, China
tiantjz@smail.nju.edu.cns
[3] University of Sheffield, S10 2TN South Yorkshire, England
admadden@hotmail.com

Abstract. Session-based recommendation using graph neural networks (GNN) is a popular approach to model users' behaviors and attributes of items from the perspective of user-item interaction sequence. However, current researches seldom incorporate the unique attributes of items to delve into a comprehensive analysis of user behaviors. In addition, GNN faces three problems when encounting complex modeling scenarios: long-range dependencies, order information loss, and data sparsity, which are essential to modeling long-tail items. We study the interactions between users and items from a new perspective. A novel Contrastive Learning based Tail Adjusted Repeat Aware Graph Neural Network (CLTAR-GNN) is proposed to tackle the problems. A Tail Adjusted Repeat (TAR) mechanism captures users' repeat-explore behaviors in both short-head and long-tail session items based on graph neural networks. Through the TAR, we are able to further understand the underlying graph-based mechanisms that influence user-item interactions. A Self-Attention (SA) network with position embedding is incorporated to overcome the sequence information loss issues, which may be caused by the complex user behaviors and item characteristics modeling. Finally, a mutli-task learning framework is employed to combine TAR, SA and a contrastive learning model into a unified framework to enhance model performance by collaboratively training graph and sequence-based embeddings. Experimental results show that CLTAR-GNN outperforms the state-of-the-art session-based recommendation methods significantly. The average improvement compared with all baselines are 17.5% (HR@20) and 22.5% (MRR@20) on both experimental datasets.

Keywords: Session-based Recommendation, Contrastive Learning, Self-Attention Networks, Tail Adjusted Repeat.

⋆ Corresponding authors

1. Introduction

Recommendation systems alleviate the issue of information overload by providing users with relevant information. These systems usually rely on user identity and historical behavior data, but for anonymous browsing sessions, recommendations must be based on short-term behavior records. The challenge of session-based recommendation is to obtain high-quality sequence representations. Traditional classic methods such as Markov chain-based methods can model the intrinsic correlations of sequence data [7]. However, there is still significant room for improvement in traditional methods regarding memory enhancement, learning and generalization capabilities, discovery of complex nonlinear sequence patterns, and computational efficiency [9]. With the rapid development of deep learning technology, approaches such as recurrent neural networks (RNN) [10] and graph neural networks (GNN) have applied deep learning methods to session-based recommendations, RNN-based methods that incorporating attention mechanism could be used to model a long session sequence [11], but RNN-based methods have two significant limitations [26]: could not accurately estimate user representations and seldom consider complex interactions between session items. GNN-based methods model sessions as graph-structured data and are able to capture complex transitions of items [18,13], which can be used to investigate users' behavior patterns in sessions by constructing user-item sequence-based graphs [19,16]. For example, Ren et al [19] focus on investigating users' different behaviors in session-based recommendation scenarios. Liu et al [16] constructed new networks to improve the performance of session-based recommendation by considering different characteristics of items.

Different from previous researches, we mainly focus on studying how patterns of repeat-explore behavior differ depending on whether they are associated with long-tail or short-head items. We define repeat-explore, long-tail and short-head as follows: Repeat behavior means the next item which user will click during a session process has already existed in the current session, otherwise it is Explore behavior. Long-tail means unpopular items, the number of which may account for a large proportion, and generate a long-tail effect [31]. In contrast, short-head means popular items, which occupies the head position of sales. The main idea of modelling repeat-explore behavior, long-tail and short-head items into a unified framework is essential important for further understanding users' future actions based on his/her historical behaviors. More importantly, incorporating the attributes of long-tail and short-head items can help better understand users' potential demands by explicitly considering users' preference patterns towards items with different popularity. Besides, long-tail items can provide more diversified information to further satisfy users' demands. It is also meaningful to promote the transformation of high-quality long-tail items into short-head items in an e-commerce system.

However, there exists two challenging problems when constructing a unified model framework: The first problem is How to mine patterns from session sequence while modelling user behavior and the intrinsic associations of items. Existing session recommendation models mainly focus on using GNN with L layers to capture $L - hop$ relations of items in a graph. The use of GNN-based methods has led to significant improvements in session recommendations. However, this solution may cause over-fitting and over-smoothing because of stacking too many layers [2]. In another aspect, when sessions are converted to graphs, the information within item orders in a session may be lost. Although combining sequence attention model can reduce the negative influence of in-

formation loss, existing methods are ineffective in capturing sequence patterns due to the complexity of user behavior modelling. Another important factor affecting the quality of session representations is the problem of data sparsity, especially for long-tail items. A deep learning model usually has a large number of parameters that need to be optimized. However, in two public datasets for session-based recommendations (Yoochoose and Diginetica), there are items that appear many times but there are also long-tail items that rarely appear. Due to lack of training data, the accuracy of session recommendations with long-tail items will be relatively low, and there will exist big biases towards popular short-head items. To address these problems, we propose a novel model called Contrastive Learning based Tail Adjusted Repeat Aware Graph Neural Network (CLTAR-GNN), which is designed to improve recommendations by capturing more of the rich information contained in sessions. The main contributions of our work can be summarized as follows:

- A novel Tail Adjusted Repeat Aware Graph Neural Network (TAR-GNN) is proposed to investigate user repeat-explore patterns in both short-head and long-tail session items. These can improve model performance by further understanding users' behaviors towards different items, while can increase the item diversity of the recommendation list.
- The re-designed Repeat-Explore and Factor Generating module consider both long-term global and short-term local dependencies between items in a session.
- By integrating self-attention with position embedding, the model can consider both order information and complex high-order relations between session items, which can further optimize the modelling of users' repeat-explore behaviors by mining sequence patterns from sessions.
- A multi-task learning framework is proposed to collaboratively learn users' repeat-explore behaviors towards items with different level of popularity and item sequence patterns. In addition, a contrastive learning framework is also Incorporated to deal with data sparsity problem caused by long-tail items.

The code could be found in: `https://github.com/Linxw718/CLTAR-GNN`

2. Related Work

2.1. Conventional Recommendation Methods

In traditional recommendation systems, where users can be identified, neighborhood-based methods have been widely used [22]. Such methods do not directly optimize the ranking of items, though Rendle et al [20] present a generic optimization criterion derived from the maximum posterior estimator for optimal Bayesian personalized ranking based on matrix factorization. However, the method seldom considers the context of a session item. Early methods for session-based recommendation borrow the idea of neighborhood-based methods. For instance, Davidson et al [4] calculate the similarity between items using the co-occurrence between them. They recommend items that are most similar to those in the current session. Neighborhood-based methods are limited by the problem of data sparsity, and do not take the order information of the session into consideration. Methods based on Markov chains [21] are capable of capturing order information, but when more preceding items are considered, the state size becomes unmanageable, making such methods unsuitable for capturing complex high-order sequential information within a session.

2.2. Deep Learning Based Methods

With the rapid development of deep learning technology, many new methods for session-based recommendation have been proposed. Hidasi et al [8] use RNN to model session sequences. These are able to leverage historical information from the session and take item order into account. Nevertheless, in long sessions, RNN-based methods cannot capture the complex dependencies between items. Li et al [12] point out that previous studies only model users' sequential behavior, and do not emphasize the user's main purpose. They propose a model based on attention mechanism, which uses a local encoder and a global encoder to model the users' behavior sequences with a view to infer the user's main purpose. Liu et al [15] consider both global and current interests of the user in the session.

In recent years, many researches have applied GNN to session-based recommendation. Wu et al [27] transformed sessions into a directed graph, in which nodes represent items and edges represent the sequential relationship of items. Item vectors are obtained through a gated GNN, after which, session vectors can be obtained by combing the item vectors using an attention network. On the basis of this research, Yu et al [32] proposed a target attentive GNN model able to generate different session representations for the same session with respect to different target items. Gupta et al [6] point out that GNN-based methods are subject to popularity bias, causing these methods to recommend popular items over long-tail items which is related to the norm of the learned items and their session-graph representations. Chen and Wong [2] spotted two information loss problems in the GNN-based methods: namely loss of long-range dependencies of items and loss of order information. They addressed these problems by adding EOPA to preserve order-information and SGAT to capture long-term dependencies. Instead of GNN, Fang [5] used self-attention networks to encode sessions that capture long-range dependencies.

The aforementioned research does not thoroughly explore the modeling of user behavior and item characteristics. In addition, enhancing the model with more detailed analysis would improve the complexity of the model, which could inadvertently compromise its inherent ability to capture and represent sequential patterns effectively. In this research, we model users' repeat-explore behaviors and item popularity into a unified framework, and incorporate self-attention with POS embedding to enhance the model capability in mining sequence patterns from sessions.

2.3. Contrastive Learning

As a kind of self-supervised learning, contrastive learning has achieved excellent performance in Computer Vision (CV) area. SimCLR [3], which stands for Similarity Constrastive Learning Representation, is a typical contrastive learning framework that generates different views of images using data-augmentation methods such as random rotation. It is designed to learn useful representations from unlabeled data by maximizing the similarity between augmented views of the same image and minimizing the similarity between different images. It produces different views of the same image close and keep views of different images away in the feature space by optimizing a loss function called NT-Xent loss. The feature representations of images obtained from SimCLR can then be applied to other downstream tasks.

In recommendation area, SimCLR can effectively learn representations of items or users without relying heavily on labeled data. Some researchers have tried to apply the

idea of contrastive learning. Zhou et al [34] used a self-supervised learning method to learn the internal relationship among items, item attributes and item sequences by designing different contrastive loss functions. Xie et al [28] applied contrastive learning to sequential recommendations. Unlike sequential recommendation, user identity is usually unknown in session-based recommendations, and more attention is paid to a user's short-term preferences, which is apparent from the shorter session sequences.

In this paper, we apply contrastive learning to session-based recommendation by mining self-supervised signals, we can capture intrinsic features of the session by considering both order-information and complex correlations between items. Therefore, the model could obtain high-quality session representations, especially long-tail item representations, for making more accurate recommendations based on repeat-explore behaviors.

3. Model Descriptions

3.1. Model Overview

Session-based recommendation aims to predict the next item that a user may click based on the sequence of items clicked earlier in the current session. Let $V = \{v_1, v_2, \ldots, v_{|V|}\}$ denotes the set of $|V|$ items that have appeared in all sessions. Assume SS is the set of N sessions, a session $S \in SS$ can be represented as a list $S = [v_{s1}, v_{s2}, \ldots, v_{sn}]$, where $v_{si} \in V$ denotes an item has been clicked in the ith position or at the ith timestamp by the same user in session S. n is the length of session S, and for different sessions S, the values of n are different. For the current session S at the nth timestamp, the objective of session-based recommendation is to predict $v_{s(n+1)}$ at the $n+1$ timestamp. The main idea is that we use the CLTAR-GNN model to calculate the score $\hat{y}_i$ (a real number) of each candidate item $v_i \in V$ based on the session S. The formula can be seen as below:

$$\begin{aligned} &\hat{y}_i = \mathrm{CLTAR} - \mathrm{GNN}(v_i, S), i \leq |V| \\ &\hat{y} = \{\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_{|V|}\} \end{aligned} \tag{1}$$

All the scores $\hat{y}_i$ ($i \leq |V|$) generate a score vector $\hat{y}$, where $\hat{y}_i$ is the score of item v_i, which is calculated by CLTAR-GNN. The top K items with the highest score from $\hat{y}$ will be recommended to the user. The architecture of the proposed CLTAR-GNN is shown in Figure 1. There are eight major components in CLTAR-GNN, including a Data Augmentation module, a Gated Graph Neural Network (GGNN) embedding layer, a Session Generation module, a Repeat-Explore component, an Item Factor Generating module, a Self-Attention network, a Contrastive Loss (CL) framework and a Multi-Task Learning strategy. The Data Augmentation module can generate new sessions by leveraging existing ones, thereby enhancing the training dataset for sessions involving long-tail items. The initial session embedding are generated by using GGNN model. The Session Generation module, Repeat-Explore module and Item Factor Generating module are collaboratively incorporated to model users' repeat-explore behaviors based on short-head and long-tail items. We define the model consists of the three modules as Tail Adjust Repeat Aware (TAR) mechanism. In addition, a Self-Attention module is also combined with TAR module to enhance the model capability in mining sequence patterns. Contrastive losses are calculated for different session augmentations. The contrastive task and recommendation task are jointly training using a Multi-Task Learning framework.

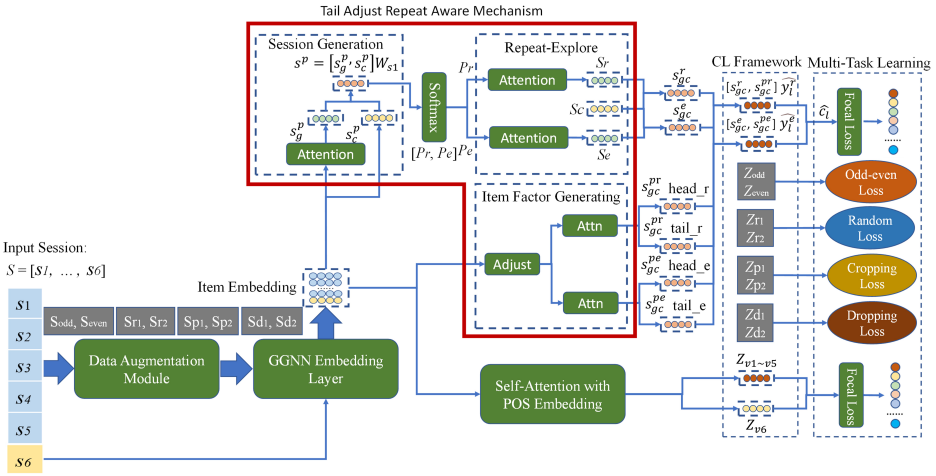


Fig. 1. A representation of the architecture of CLTAR-GNN

3.2. Data Augmentation Module

The Data Augmentation module can generate two new sessions for each existing session through augmentation operations. The two new sessions are similar to the existing ones, and will be used as a positive pair in Contrastive Learning, while any two augmented sessions generated from different sessions are used as a negative pair. CLTAR-GNN adopts four data augmentation operations, including Odd-Even augmentation, Random augmentation, Cropping augmentation and Dropping augmentation [28,34]. Assume there exists N sessions, then each augmentation can generate $2N$ new sessions, which are prepared for the Contrastive Learning module.

As seen in Figure 1, the output of Odd-even augmentation is S^{odd} and S^{even}, which can be treated as a positive pair for odd-even contrastive learning (Odd-Even loss); the output of Random Augmentation is S^{r1} and S^{r2}, which can be treated as a positive pair for random contrastive learning (Random loss); the output of Cropping Augmentation is S^{p1} and S^{p2}, which can be treated as a positive pair for cropping contrastive learning (Cropping loss); the output of Dropping Augmentation is S^{d1} and S^{d2}, which can be treated as a positive pair for dropping contrastive learning (Dropping loss).

3.3. GGNN Embedding Layer

In this research, Gate Graph Neural Network (GGNN) is adopted as the embedding layer [13]. The advantage of GGNN over traditional sequence-based embedding methods, such as LSTM or GRU, is that it can capture high-order graph-based correlations among items. For a session $S = [s_1, s_2, \ldots, s_n]$, its in-degree and out-degree adjacency matrices are defined as E^{in} and $E^{out} \in R^{n \times n}$. We define $E \in R^{n \times 2n}$ as the concatenate of E^{in} and E^{out}. Assuming that there exists t iterations, then for node i in session S at t, its node

embedding at $t+1$ could be represented as:

$$m_i^{t+1} = \sum_{v \in N_i} M_t(h_i^t, h_v^t | E, S) \tag{2}$$

$$h_i^{t+1} = U_t(h_i^t, z_i^t, r_i^t | m_i^{t+1}) \tag{3}$$

where M_t is message propagation operation, and indicates the correlation between i and its neighbors $v \in N_i$. h_i^t and h_v^t denote item embedding at t. z_i^t controls forget information and r_i^t controls the influence of newly generated information. U_t is similar to GRU, and is used to update the embedding of item i at $t+1$. The length of h_i^t is defined as d.

3.4. Tail Adjust Repeat Aware Mechanism (TAR)

In this section, a TAR mechanism is proposed to collaboratively model users' repeat-explore behavior patterns on short-head and long-tail items. The mechanism consists of three modules: A Session Generation module is designed to calculate the probabilities [P_r,P_e] of users selecting different behaviors. The Repeat-Explore module calculates the preference score for each candidate item under different behaviors of a user. The Item Factor Generating module is used to encode items with different level of popularity (short-head and long-tail). Finally, a Focal Loss based Repeat-Tail loss function is designed to provide optimization strategy for TAR.

Session Generation Module. The main target of the session generation module is to adopt an attention mechanism that captures a user's main preference in session S. This mechanism can then be used to discover the probability distributions of users' behavior patterns. Existing researches often use probability distributions to describe users' preferences towards different behaviors [19,30]. In this article, we focus mainly on two types of user behavior pattern: repeat and explore. We define the probability of users' repeat patterns as P_r and explore patterns as P_e. Formulas for P_r and P_e are shown below:

$$s_g^p = \sum_{i=1}^{n} a_i^g h_i^t; \quad s_c^p = h_n^t \tag{4}$$

$$s^p = [s_g^p, s_c^p] W_{s1} \tag{5}$$

$$[P_r, P_e] = softmax(W_{s2} \times s^p) \tag{6}$$

where a_i^g is the attention weight of item i in session S, which is utilized to determine the influence of a user's historical click behaviors on items to his/her current state at timestamp n. $W_{s1} \in R^{2d \times d}$, $W_{s2} \in R^d$ means weight matrices. Different from previous studies [19], inspired by long-short term modules which are widely applied in sequence processing [15], we consider both long-term global dependencies s_g^p and short-term local dependencies s_c^p by calculating h_n^t. In formula 6, the P_r and P_e of users' repeat-explore behaviors for selecting the $(n+1)$th item in session S are mainly determined by s^p, which is the concatenate of s_g^p and s_c^p. P_r and P_e are two probabilities, which are used to determine whether the user will adopt repeat or explore behaviors at $(n+1)$th timestamp.

Repeat-Explore Module. The Repeat-Explore module is used to re-calculate session representations based on two behavior patterns: repeat and explore. Assume $I_s \in V$ is the item set, and that each item appears in session S. x represents item embedding and $T(x)$ represents item id which x represents. Then the indicator function $H(x, y)$ is defined in formula 7. $y = 0$ represents repeat mode, $y = 1$ represents explore mode. The idea of using indicator function to encode different user behaviors are discussed in existing researches [1,17,33]. We design a repeat-explore function: for an arbitrary item l in V, its score of whether a user will click it at $n + 1$ in session S is represented in formula 8.

$$H(x, y) = \begin{cases} 1 - y, T(x) \in I_s \\ y, \quad T(x) \notin I_s \end{cases} \tag{7}$$

$$\begin{aligned} &\textbf{Repeat:} \quad \hat{y}_l^r = H(e_l, 0) e_l (s_{gc}^r)^T \\ &\textbf{Explore:} \quad \hat{y}_l^e = H(e_l, 1) e_l (s_{gc}^e)^T + \gamma \cdot \frac{\sum_{i \in S} sim(e_l, e_i)}{len(S)} \end{aligned} \tag{8}$$

where s_g^r and s_g^e are global session representation of repeat and explore with attention weight [15,5] as a_i^r and a_i^e respectively (Similar to formula 4, $s_g^r = \sum_{i=1}^n a_i^r h_i^t$ and $s_g^e = \sum_{i=1}^n a_i^e h_i^t$). s_{gc}^r and s_{gc}^e are combination of global and local representations of Repeat and Explore patterns respectively (similar to formula 5, $s_{gc}^r = [s_g^r, s_c^p] W_{r1}$ and $s_{gc}^e = [s_g^e, s_c^p] W_{e1}$). $\hat{y}_l^r$ is the score of item l in a repeat pattern, $\hat{y}_l^e$ is the score of item l in a explore pattern. e_l is the embedding of item l with its size as d, the initial value of which is pre-trained by neural network embedding (pytorch nn.Embedding). In our study, explore pattern led to item recommendations that have fewer interactions (ex. Co-occurrence in all sessions) with items in session S, while ignores that in many cases, users prefer to click new items that have high co-occurrences with existing items in a session [17,32,33]. To resolve this issue, we added $\sum_{i \in S} sim(e_l, e_i)/len(S)$ to the **Explore** formula in formula 8, and γ is the weight to control the influence of item correlations.

Item Factor Generating. The main target of item factor is to generate session embeddings and their influence factors by consider both long-tail and short-head items. Similar to previous studies, which model items according to their unique characteristics [1,16,25], we used Pareto rules to identify short-head popular and long-tail less popular items by counting each item's click frequency. Assume I_S^T, I_S^H represent the set of long-tail and short-head items which appear in session S. We re-define item embedding as:

$$F(x) = \begin{cases} x + [1, 1, 1, \ldots, 1], T(x) \in I_S^T \\ x + [0, 0, 0, \ldots, 0], T(x) \in I_S^H \end{cases} \tag{9}$$

where $F(x)$ is adjust function. x represents item embedding and $T(x)$ represents item id of x. $F(x)$ could adjust x based on its short-head or long-tail characters. Vector $[1, 1, 1, \ldots, 1]$ and $[0, 0, 0, \ldots, 0]$ have the same length as x. Based on the adjust function $F(x)$, attention mechanisms [23,29] are then adopted to obtain long-term global session representations s_g^{pr} and s_g^{pe}, which are corresponding to repeat and explore behaviors respectively. The expressions of global sessions could be seen in formula 10. Short-term local representations of s_c^{pr} and s_c^{pe}, which are corresponding to repeat and explore behaviors, could be

seen in formula 11. Finally, based on formulas 10 and 11, the combinations of long-short term global and local session representations s^{pr} and s^{pe}, which are corresponding to repeat and explore behaviors, could be seen in formula 12.

$$s_g^{pr} = \sum_{i=1}^{n} a_i^{pr} F(h_i^t); \quad s_c^{pr} = F(h_n^t) \tag{10}$$

$$s_g^{pe} = \sum_{i=1}^{n} a_i^{pe} F(h_i^t); \quad s_c^{pe} = F(h_n^t) \tag{11}$$

$$s_{gc}^{pr} = [s_g^{pr}, s_c^{pr}]W_{pr1}; \quad s_{gc}^{pe} = [s_g^{pe}, s_c^{pe}]W_{pe1} \tag{12}$$

where a_i^{pr} and a_i^{pe} are attention weights. W_{pr1} and $W_{pe1} \in R^{2d \times d}$ are weight matrices. s_{gc}^{pr} and s_{gc}^{pe} are used to generate four item factors: R_{head}^r and R_{head}^e are for short-head items, which are associated with repeat and explore behaviors. R_{tail}^r and R_{tail}^e are for long-tail items, which are associated with repeat and explore behaviors (formula 13, 14).

$$R_{head}^r = sigmoid(W_{p2} \cdot s_{gc}^{pr}); \quad R_{tail}^r = 1 - R_{head}^r \tag{13}$$

$$R_{head}^e = sigmoid(W_{p3} \cdot s_{gc}^{pe}); \quad R_{tail}^e = 1 - R_{head}^e \tag{14}$$

where W_{p2} and $W_{p3} \in R^d$ are the weight matrices. R_{head}^r and R_{tail}^r are item factors which are used to calculate the probabilities of users' selections for short-head or long-tail items when engaging in repeat behaviors. R_{head}^e and R_{tail}^e are item factors which are used to calculate the probabilities of users' selections for short-head or long-tail items when engaging in explore behaviors.

Repeat-Tail Loss. We design repeat-tail loss l_{rt-rec} to investigate users' different repeat-explore patterns for both long-tail and short-head items. For an arbitrary item $v_l \in V$, we have already obtained its repeat and explore pattern probability for session S, which are P_r and P_e respectively, through formula 6; while the item's recommendation score under repeat and explore patterns, which are $\hat{y}_l^r$ and $\hat{y}_l^e$, could be obtained through formula 8. Then the item l's recommendation score $\hat{c}_l$ could be represented as:

$$\hat{c}_l = P_r \hat{y}_l^r T(l, 1) + P_e \hat{y}_l^e T(l, 0) \tag{15}$$

where $T(x, y)$ is an indicator function to determine which factors should be adopted to item l according to its unique characters and its correlations with session S. $T(x, y)$ could be represented as in formula 16.

$$T(x, y) = \begin{cases} R_{head}^r, x \in I^H, y = 1 \\ R_{tail}^r, x \in I^T, y = 1 \\ R_{head}^e, x \in I^H, y = 0 \\ R_{tail}^e, x \in I^T, y = 0 \end{cases} \tag{16}$$

where I_H and I_T are the set of short-head and long-tail items. For all candidate items $[v_1, v_2, \ldots, v_{|V|}]$, the recommendation score could be $\hat{c} = [\hat{c}_1, \hat{c}_2, \ldots, \hat{c}_{|V|}]$, then vector $\hat{y}_c = softmax(\hat{c})$ is the vector of final recommendation scores based on probability

distributions. It denotes the probability of each candidate item to be clicked next at $n+1$ for session S. The cross-entropy loss $l_{rt-ce}(\hat{y}_c)$ between the predicted probability and the ground-truth is calculated as:

$$\begin{aligned} \hat{y}_c &= softmax(\hat{c}) = [\hat{y}_{c1}, \hat{y}_{c2}, \ldots, \hat{y}_{c|V|}] \\ l_{rt-ce}(\hat{y}_c) &= -\sum_{l=1}^{|V|}(y_l \times log(\hat{y}_{cl}) + (1-y_l) \times log(1-\hat{y}_{cl})) \end{aligned} \tag{17}$$

where y_l is the one-hot embedding denoting the ground-truth item (It denotes which item was truly clicked at $n+1$ timestamp). $\hat{y}_{cl}$ is the lth element in $\hat{y}_c$. It denotes the probability of item l to be clicked at timestamp $n+1$ in session S. Due to the imbalanced data of sessions, we use focal loss [14] in CLTAR-GNN. Focal loss could increase the impact of hard samples and decrease the impact of easy samples in the process of model optimization. Hence, focal loss can help make more accurate recommendations of long-tail items. The focal loss based repeat-tail loss $l_{rt-rec}(\hat{y}_c)$ is formulated as:

$$l_{rt-rec}(\hat{y}_c) = (1 - exp(-l_{rt-ce}(\hat{y}_c)))^{\gamma} l_{rt-ce}(\hat{y}_c) \tag{18}$$

where γ is the focusing parameter and we set it as to 2. The repeat-tail loss $l_{rt-rec}(\hat{y}_c)$ can predict the next likely-to-be-clicked short-head or long-tail item within a session when operating under the repeat-explore behaviors, with the objective of aligning the predicted value as closely as feasible to the actual value (the item that is genuinely clicked).

3.5. Self-Attention Network (SA)

The TAR mechanism mainly focuses on modelling user different behaviors towards items with different popularity, which can detect more complex interaction patterns from a session. However, this mechanism is chiefly concerned with the design of theoretical models that address intricate relationships between users and items, which may consequently reduce the model's inherent capability for sequence modeling. In another aspect, GGNN is adopted as embedding layer for capturing high-order correlations among items, while its limitations would be that GGNN could not capture order information and long-range dependencies in a session. Though there exists related researches optimized existing GGNN-based methods to solve the problems mentioned above to a certain extent [2], it is challenge to incorporate more complex user-item interaction models, such as TAR, into existing researches, because modelling complex interactions between user-item may compromise the sequential modelling capabilities of the model.

In this research, we find that sequential modelling is very important to the accuracy of recommendation results. Thus, we design a Self-Attention (SA) network specifically for sequential modelling, and then adopt a multi-task learning framework to collaboratively train the model and TAR in a unified framework. For a session $S = [v_{s1}, v_{s2}, \ldots, v_{sn}]$, its t iteration item embedding through GGNN could be represented as $E' = [h_1^t, h_2^t, \ldots, h_n^t]$ with dimension size as d. In order to take the order information of the session into account, we stack position embeddings $P = [p_1, p_2, \ldots, p_n]$ for each item in the session, where p_i denotes the position embedding of item v_{si} . The input E for self-attention networks is:

$$E = E' + P \tag{19}$$

Self-attention [23] are composed of multi-head attention and feed-forward networks. Multi-head attention could capture different information from the session. Assume H heads are assigned, then the output F of multi-head attention could be seen in formula 20. $W^O \in R^{Hd \times d}$ is the projection matrix. Then F is passed through a two-layer fully connected feed-forward network (FFN) with activation functions (σ) to obtain output as $H_S = [h_1, h_2, \ldots, h_n]$. According to experiments, we find that using the average of the last two item vectors (h_{n-1} and h_n) as session representation z can produce better results.

$$\begin{aligned} F &= Concat(head_1(E), \ldots, head_H(E))W^O + E \\ H_S &= \sigma(\mathrm{FFN}(\mathrm{FFN}(F))) = [h_1, h_2, ..., h_n] \\ z &= \frac{h_{n-1} + h_n}{2} \end{aligned} \tag{20}$$

After getting the session representation z, we can calculate the items that should be recommended to users. For each candidate item $v_l \in V$, its corresponding recommendation score is calculated as $g_l = z \cdot e_l$, where e_l is the vector of item v_l (e_l is calculated by using pytorch nn.Embedding). Let $g = [g_1, g_2, \ldots, g_{|V|}]$ denotes the recommendation scores of all items. $\hat{y}_s = softmax(g)$ is the vector of final recommendation scores. It denotes the probability of each candidate item to be clicked next for session S. The cross-entropy between the predicted probability and the ground-truth is calculated as:

$$\begin{aligned} &g = [g_1, g_2, \ldots, g_{|V|}] \quad where \quad g_l = z \cdot e_l \quad and \quad l \leq |V| \\ &\hat{y}_s = softmax(g) = [\hat{y}_{s1}, \hat{y}_{s2}, \ldots, \hat{y}_{s|V|}] \\ &l_{sa-ce}(\hat{y}_s) = -\sum_{l=1}^{|V|}(y_l \times log(\hat{y}_{sl}) + (1 - y_l) \times log(1 - \hat{y}_{sl})) \end{aligned} \tag{21}$$

where y_l is the one-hot embedding denoting the ground-truth item. $\hat{y}_{sl}$ is the lth element in $\hat{y}_s$. $\hat{y}_{sl}$ denotes the probability of item l to be clicked next at $n+1$ for session S. Similar to formula 18, the focal loss based self-attention loss $l_{sa-rec}(\hat{y}_s)$ is formulated in 22, where γ is the focusing parameter and we set it as to 2.

$$l_{sa-rec}(\hat{y}_s) = (1 - exp(-l_{sa-ce}(\hat{y}_s)))^{\gamma} l_{sa-ce}(\hat{y}_s) \tag{22}$$

3.6. Contrastive Loss Function

Session embeddings of S from different modules are summarized as: **Repeat-Explore**: s^r_{gc}, s^e_{gc} (Formula 8). **Item Factor**: s^{pr}_{gc}, s^{pe}_{gc} (Formula 12). **Self-Attention**: z (Formula 20). We merge all representations into a unified vector as $sz = [s^r_{gc}, s^e_{gc}, s^{pr}_{gc}, s^{pe}_{gc}, z]$. Assume a training batch contains N sessions $S_1, S_2, \ldots, S_N$. Data augmentation strategies from Odd-even, Random, Cropping, Dropping are used to generate positive instances for each session S_i. For each session S_i, each strategy could generate two positive instances, which are S^{odd}_i and S^{even}_i for Odd-even, S^{r1}_i and S^{r2}_i for Random, S^{p1}_i and S^{p2}_i for Cropping, S^{d1}_i and S^{d2}_i for Dropping. The representations of all instances are sz^{odd}_i and sz^{even}_i, sz^{r1}_i and sz^{r2}_i, sz^{p1}_i and sz^{p2}_i, sz^{d1}_i and sz^{d2}_i respectively. Then for all sessions, we adopt SimCLR framework for contrastive learning and use **NT-Xent** loss (Normalized Temperature-Scaled Cross-Entropy Loss) to generate contrastive loss [3], which are odd-

even loss $l_{cl}^{odd-even}$, random loss l_{cl}^{r1-r2}, cropping loss l_{cl}^{p1-p2} and dropping loss l_{cl}^{d1-d2}:

$$\begin{aligned} l_{cl}^{odd-even} &= \sum_{i=1}^{N} \mathrm{NT-Xent}(S_i, S_i^{odd}, S_i^{even}) \\ l_{cl}^{r1-r2} &= \sum_{i=1}^{N} \mathrm{NT-Xent}(S_i, S_i^{r1}, S_i^{r2}) \\ l_{cl}^{p1-p2} &= \sum_{i=1}^{N} \mathrm{NT-Xent}(S_i, S_i^{p1}, S_i^{p2}) \\ l_{cl}^{d1-d2} &= \sum_{i=1}^{N} \mathrm{NT-Xent}(S_i, S_i^{d1}, S_i^{d2}) \end{aligned} \tag{23}$$

3.7. Predictions

In previous sections, we introduced the calculation methods for repeat-tail loss $l_{rt-rec}(\hat{y}_c$ (formula 18), self-attention loss $l_{sa-rec}(\hat{y}_s)$ (formula 22), contrastive loss $l_{cl}^{odd-even}$, l_{cl}^{r1-r2}, l_{cl}^{p1-p2} and l_{cl}^{d1-d2} (formula 23). In this section, a multi-task learning strategy is adopted to obtain the final loss l_{cltar} of CLTAR-GNN. The loss function could be seen as below in formula 24:

$$l_{cltar} = \alpha \cdot l_{rt-rec} + \beta \cdot l_{sa-rec} + \gamma(l_{cl}^{odd-even} + l_{cl}^{r1-r2} + l_{cl}^{p1-p2} + l_{cl}^{d1-d2}) \tag{24}$$

where α, β and γ are weight parameters to control the contributions of each loss. Based on grid search of hyperparameter, we assign α as 0.3, β as 0.7 and γ as 1. Finally, for a session $S = [v_{s1}, v_{s2}, \ldots, v_{sn}]$, the probability score $\hat{y}_i$ for each candidate item v_i ($i \leq |V|$) that could be clicked at timestamp $n+1$ is as follows:

$$\hat{y}_i = \alpha \times \hat{y}_{ci} + \beta \times \hat{y}_{si} \tag{25}$$

4. Experiments

4.1. Experiment Settings

Two widely-used public datasets Yoochoose and Diginetica are selected to test the performance of the model. Yoochoose is from RecSys Challenge 2015, which contains the click sequence of users on an e-commerce website within six months. Because the Yoochoose dataset is too large, only the latest 1/64 part of it is used, as done in [27]. Diginetica is from CIKM Cup 2016 challenge. The statistics of datasets are shown in Table 1.

Evaluation Metrics In order to compare with other models for session-based recommendation, two evaluation metrics HR@20 and MRR@20 are used.

HR@20 (Hit Rate): This metric indicates the proportion of correctly predicted sessions in all testing sessions. Correctly predicted n_{hit} means that the ground-truth item is among the top 20 items with the highest recommendation scores calculated by the model.

Table 1. Statistics of datasets used in the experiment

Statistics	Yoochoose 1/64	Diginetica
# of clicks	557,248	982,961
# of training sessions	369,859	719,470
# of test sessions	55,898	60,858
Average length	6.16	5.12

In formula 26, n_{hit} represents sessions that have been correctly predicted, and N denotes the number of all testing sessions.

$$HR@20 = \frac{n_{hit}}{N} \tag{26}$$

MRR@20 (Mean Reciprocal Rank): This metric denotes the average of reciprocal ranks of the ground-truth items. When the ground-truth item is not among the top-20 recommended items, the reciprocal rank is set to 0. In formula 27, S denotes the set of correctly predicted sessions, and $rank_i$ denotes the rank of ground-truth item for session i.

$$MRR@20 = \frac{1}{N}(\sum_{i \in S} \frac{1}{rank_i}) \tag{27}$$

Baseline Methods The proposed model CLTAR-GNN will be compared with the following representative baseline methods on Yoochoose and Diginetica datasets.

- Item-KNN [22] recommends similar items of the previous clicked item in the session based on cosine similarity.
- BPR-MF [20] optimizes a pairwise ranking objective function via stochastic gradient descent.
- FPMC [21] models next-basket recommendation. The user feature is removed since it is unavailable in session-based recommendation.
- GRU4REC [8] models user sequences for session-based recommendation using RNN.
- NARM [12] employs a local encoder and a global encoder with an attention mechanism to model the user's sequential behavior and capture the user's main purpose.
- STAMP [15] captures users' general interests of session context and user's current interests of last click.
- RepeatNet [19] takes repeat-explore consumption behaviors into account and uses GNN to model users' repeat and explore behaviors.
- CSRM [24] incorporates collaborative modeling into session-based recommendation with an end-to-end model.
- SR-GNN [27] models session sequences into graph-structure data and uses graph neural networks to capture complex item transitions.
- GC-SAN [29] integrates self-attention layers with graph neural networks to learn long-range dependencies.
- TAGNN [32] proposes a target attentive network which could discover the relevance of target item with graph neural networks.
- LESSR [2] designs two new graph neural network based layers, which are EOPA and SGAT to solve the information loss problems.

- SR-SAN [5] captures long-range dependencies between items using self-attention networks.
- NISER [6] normalizes the item and session-graph representations to improve the recommendation accuracy of long-tail items.

Hyperparameter Setup Following [16], for sessions with length ¿ 10, we consider only the most recently clicked 10 items. The dimensions of both item embeddings and position embeddings are set as 300 (Hyper-parameter studies of embedding size could be seen in Figure 2). The batch size is set to 100. Following [13], The Adam optimizer is adopted with the initial learning rate 0.001. Decay rate is set as 0.1 per 3 epochs. We use dropout probability of 0.1 and attention heads of 4 on self-attention networks. The focusing parameter for focal loss is set to 2. The weight of contrastive loss in multi-task learning is set to 1, which make it have the same impact for the model optimization with the loss of recommendation task. We use the normalization strategy as introduced in [16] and set the scale factors as 16 following the paper. The number of training epochs is set to 30, and we adopt the early stopping strategy. When the performance does not improve after 10 consecutive epochs, the training will be terminated.

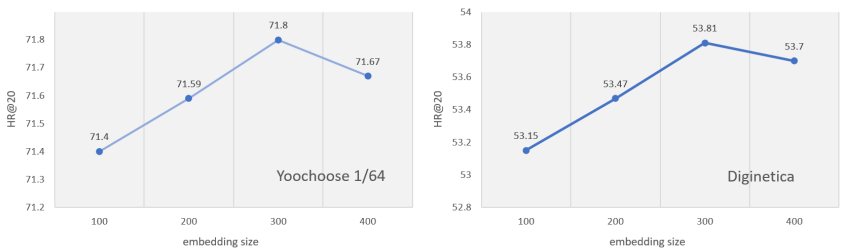


Fig. 2. The performance of CLTAR-GNN with different embedding sizes

Another important hyper-parameter is split-ratio, which decides the proportion of long-tail items in training dataset. We assign the split-ratio as $sr \in 0.5, 0.6, 0.7, 0.8, 0.9$, and use sr to divide all items in V into long-tail and short-head groups. Then we test different sr to find the best results, which is evaluated by HR@20, MRR@20. Experimental results of selecting sr could be seen in Figure 3. The proposed CLTAR-GNN could obtain the best HR@20 and MRR@20 on Yoochoose 1/64 when sr = 0.7; while for Diginetica, the best sr is 0.6. The training process could be seen in Figure 4, the loss of the CLTAR-GNN is small compared with other three baselines during the training process, while the performance on validation datasets in terms of HR@20 and MRR@20 keeps a relatively high score.

4.2. Overall Performance Comparison

Results and Observations To evaluate the performance of CLTAR-GNN, we compare it with 14 state-of-the-art baselines introduced above. Table 2 summarizes the best results

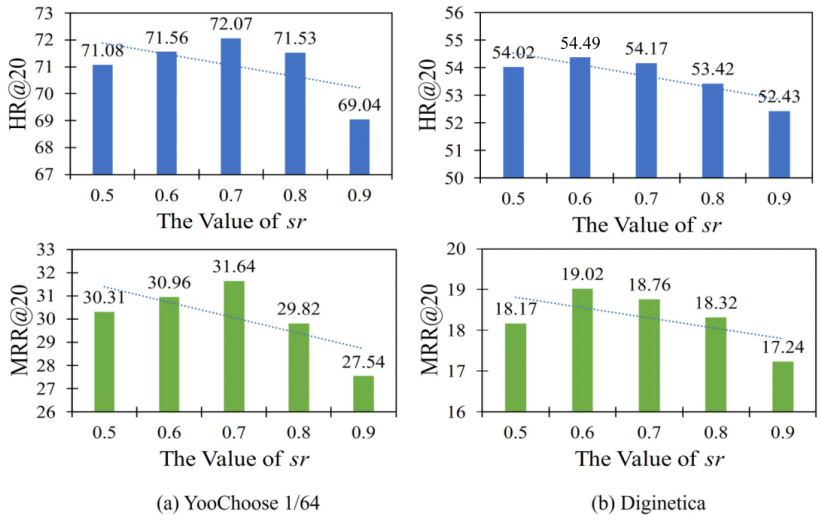


Fig. 3. The Performance of sr on Yoochoose 1/64 and Diginetica

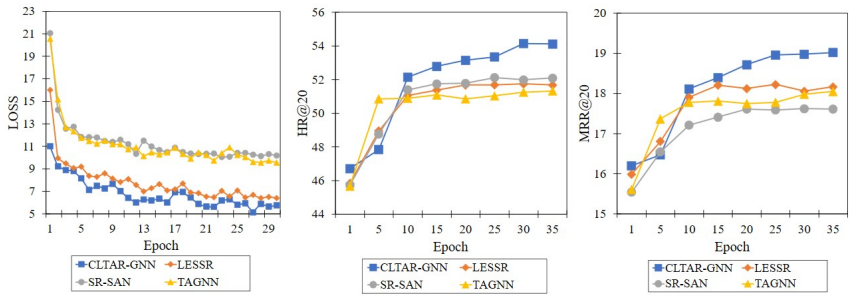


Fig. 4. The Training Process of CLTAR-GNN in terms of LOSS, HR@20 and MRR@20

of all models. As seen in Table 2, Traditional recommendation methods including Item-KNN, BPR-MF and FPMC obtain the lowest performance score. These methods make recommendations only based on the similarities or transitions between items, which are unable to leverage other important information such as the order of items. Compared with traditional methods, GRU4REC applies RNN to session-based recommendation and outperforms traditional methods. Both NARM and STAMP consider the current interest and global interest of the user, and thus get better results. RepeatNet considers users' repeated-explore patterns, and has better performance than other RNN-based methods. CSRM applies collaborative neighbor information to current session, and it's performance on Yoochoose dataset is very competitive.

The GNN-based methods generally perform better than the methods mentioned above. SR-GNN firstly model the session sequences as graphs and then consider the transitions of items, obtaining better results than RNN-based methods. TAGNN is an improved version

Table 2. The performance of CLTAR-GNN and its variants compared with 14 state-of-the-art Baseline

Methods	Yoochoose 1/64		Diginetica	
	HR@20	MRR@20	HR@20	MRR@20
Item-KNN	51.60	21.81	35.75	11.57
BPR-MF	31.31	12.08	5.24	1.98
FPMC	45.62	15.01	26.53	6.95
GRU4REC	60.64	22.89	29.45	8.33
NARM	68.32	28.63	49.70	16.17
STAMP	68.74	29.67	45.64	14.32
RepeatNet	70.71	31.03	47.79	17.66
CSRM	71.45	30.06	50.55	16.38
SR-GNN	70.57	30.94	50.73	17.59
GC-SAN	70.66	30.04	51.70	17.61
TAGNN	71.02	31.12	51.31	18.03
LESSR	70.64	30.97	51.71	18.15
SR-SAN	71.74	31.58	52.04	17.61
NISER	71.27	<u>31.61</u>	53.39	18.72
TAR-GNN w/o sa	69.85	30.76	51.73	18.22
TAR-GNN w/o $sa - ls$	69.46	30.45	51.52	18.03
TAR-GNN	<u>71.73</u>	31.17	<u>53.92</u>	<u>18.98</u>
CLTAR-GNN w/o sa	70.31	30.98	52.65	18.66
CLTAR-GNN w/o $sa - ls$	70.12	30.82	52.49	18.31
CLTAR-GNN	**72.07**	**31.67**	**54.49**	**19.02**

of SRGNN. By adding a target attentive module, its performance is better than SRGNN in all the metrics. LESSR proposes two new layers on the basis of graph neural networks to solve the information loss problem, which outperforms TAGNN on Diginetica dataset. GC-SAN outperforms SR-GNN since it applies self-attention mechanism and combines it with graph neural network to capture long-range dependencies, however, the information of global dependencies between items may be lost during the process of neighbor item aggregations. SR-SAN mainly uses self-attention networks to replace GNN for session encodings, and achieves the second place on Yoochoose dataset in term of HR@20. Though NISER uses GNN to get session encodings, it adopts optimal normalization representation method to obtain more accurate item and session embeddings, it greatly improves the performance of long-tail items recommendations and achieves the second place on Yoochoose dataset in term of MRR@20. The proposed method CLTAR-GNN outperforms all baselines on both datasets in terms of all metrics, proving its effectiveness in the session-based recommendations task. Specifically, CLTAR-GNN outperforms the best baseline by average 1.2% (HR@20) and 0.9% (MRR@20) on Yoochoose and Diginetica datasets. The average improvement compared with all baselines are 17.5% (HR@20) and 22.5% (MRR@20) on both datasets. In order to better illustrate the effectiveness of the proposed model, five variants are proposed, which are TAR-GNN, TAR-GNN w/o sa, TAR-GNN w/o $sa - ls$, CLTAR-GNN w/o sa, CLTAR-GNN w/o $sa - ls$.

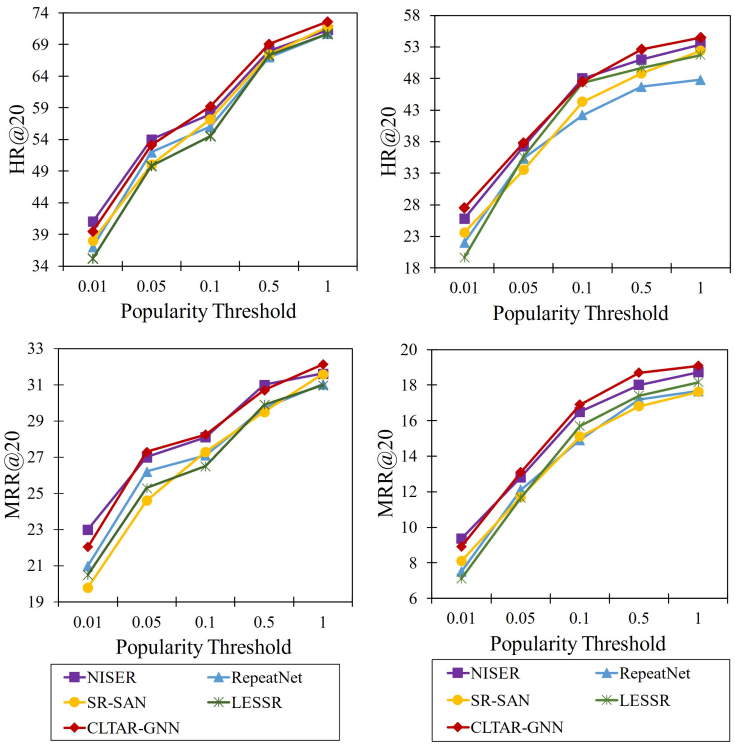
- **TAR-GNN**: CLTAR-GNN does not contain contrastive learning.
- **TAR-GNN w/o** sa: TAR-GNN does not contain self-attention.

- **TAR-GNN w/o** $sa - ls$: TAR-GNN does not contain both self-attention and long-short term mechanism.
- **CLTAR-GNN w/o** sa: CLTAR-GNN does not contain self-attention.
- **CLTAR-GNN w/o** $sa - ls$: CLTAR-GNN does not contain both self-attention and long-short term mechanism.

Based on the performance analysis of the proposed CLTAR-GNN and its variants, our findings are summarized in four aspects:

- **Considering users' repeat-explore patterns in both short-head and long-tail items could improve the performance compared with RepeatNet, GC-SAN and TA-GNN.** TAR-GNN w/o $sa - ls$ could make an average improvement of 7.8% and 2.1% in terms of HR@20 and MRR@20 on Diginetica compared with RepeatNet, while the performance on Yoochoose 1/64 is not as good as RepeatNet. One main reason is that the number of sessions in Diginetica is bigger than Yoochoose 1/64, so Diginetica could provide more training instances with long-tail items and reduce the negative influence of data sparsity. Compared with GC-SAN and TA-GNN, the average improvements of TAR-GNN are 3.8% (HR@20) and 4.1% (MRR@20) on two datasets, which indicates that on the premise of incorporating self-attention network at the same time, the investigation of behavior patterns could bring more advantages in session recommendations.
- **Incorporating long-short term module into TAR could make further improvement.** Compared with TAR-GNN w/o $sa - ls$, the average improvements of TAR-GNN w/o sa are 0.5% and 1.1% on two datasets. Experimental results indicate the effectiveness of the proposed long-short term module. Besides, compared with NARM and STAMP, which considers both general and current interests of users, TAR-GNN w/o sa could also have an average improvement of 7% and 12% in terms of HR@20 and MRR@20.
- **Self-attention module is important for capturing order-information, which could not be well captured by only using GNN based methods.** Compared with TAR-GNN w/o sa, TAR-GNN could obtain a more significant improvement in terms of two metrics. Besides, TAR-GNN outperformances SR-GNN and TAGNN by average 3.5% (HR@20) and 3% (MRR@20) on two datasets. Experimental results illustrate that TAR-GNN could better leverage order information derived from self-attention to improve the performance. Compared with LESSR, which adopts optimized GNN and GRU for capturing long-term dependencies and order information, TAR-GNN incorporating self-attention could also obtain a better result.
- **Data sparsity limits the ability of the proposed model to process long-tail items, while the contrastive learning could better solve the problem to a certain extent.** SR-SAN and NISER exhibits strong competitiveness in the task of session recommendation, and outperformance TAR-GNN in terms of two metrics. We consider the potential reasons that limit the ability of TAR-GNN could be the lack of training samples of long-tail items. Adding contrastive learning validates our motivations that it is necessary to use data augmentation strategies to improve the performance. Experimental results show that incorporating CL could significantly improve the performances of TAR-GNN and CLTAR-GNN outperformances the two competitive baselines by average 1.5% (HR@20) and 1.2% (MRR@20) on both datasets.

Performance based on Popularity Threshold As illustrated in [6], popularity threshold is an indicator to evaluate whether an item is popular or not. The indicator could help better evaluate the performance of CLTAR-GNN on the task of long-tail item recommendations. The popularity of an item could be defined as $popularity = \frac{\varphi(i)}{max\varphi(i)}$, where $\varphi(i)$ is the number of times item i appears in all sessions. In order to evaluate the performance of long-tail recommendations, we should firstly construct a long-tail testing dataset L. For a session $S = [s_1, s_2, \ldots, s_n]$ in the testing dataset, if its $n+1$th clicked item is non-popular ($popularity \leq threshold$), then we set $S \in L$. We set the threshold as 0.01, 0.05, 0.1, 0.5, 1, and obtain five long-tail testing datasets, the performances of the proposed model and four other baselines on the five datasets are summarized in Figure 5.



(a) YooChoose 1/64 (b) Diginetica

Fig. 5. The performance of long-tail item recommendations on Yoochoose 1/64 and Diginetica

In Figure 5, when the threshold is small, the target items of all sessions in the long-tail testing dataset are non-popular. The proposed CLTAR-GNN outperforms the other four baselines significantly for those sessions filtered by threshold with long-tail items as target. While for RepeatNet, SR-SAN and LESSR, there are no mechanisms for them to

identify whether current session is a long-tail or short-head recommendation task. Though NISER is a competitive model for the task of long-tail item recommendations, CLTAR-GNN outperformances NISER by average 1.2% and 0.8% improvement on two datasets in terms of HR@20 and MRR@20. One main reason is that CLTAR-GNN uses item factor to decide whether short-head or long-tail items are taken as recommendation targets.

Performance based on Repeat-explore Patterns In this section, a new experiment is conducted to investigate the models' performance on users' repeat and explore behaviors separately. Repeat behavior means repeating click a item that has already existed in a session, explore means a new item will be clicked in the session. According to the definition of repeat and explore, we could divide all sessions in the testing dataset into repeat sessions and explore sessions, and evaluate the performance of the proposed model on the two types of sessions. Experimental results could be seen as in Table 3.

Table 3. The performance of repeat-explore

Methods		Yoochoose 1/64		Diginetica	
		HR@20	MRR@20	HR@20	MRR@20
CLTAR-GNN	R	**93.55**	**64.85**	**91.87**	**58.28**
	E	**64.35**	19.26	**44.60**	8.82
RepeatNet	R	91.71	57.65	87.43	49.33
	E	61.27	27.81	39.69	**8.89**
SR-GNN	R	92.25	63.08	86.14	52.26
	E	62.55	19.04	42.10	8.74
SR-SAN	R	92.93	64.37	86.86	51.53
	E	64.11	**19.31**	43.02	8.45

As seen in Table 3, "R" represents Repeat sessions, and "E" represents Explore sessions. In both Yoochoose 1/64 and Diginetica datasets, the percentage of Repeat sessions is about 30% and that of Explore sessions is about 70%. RepeatNet is a research to investigate users' repeat-explore behaviors, CLTAR-GNN obtains better performance compared with RepeatNet because the proposed model could capture more accurate repeat-explore patterns. The proposed CLTAR-GNN also outperformances SR-GNN and SR-SAN significantly. The two baselines represent GNN-based and Attention-based session recommendation models respectively.

4.3. Ablation Test

An ablation study was conducted to help determine the contribution of each component of the proposed model. Variations of CLTAR-GNN are tested, in which specific components were removed or replaced. These assignments were as follows:

- **-CL**: Contrastive learning removed.
- **-GGNN**: Use neural network embeddings to replace GGNN.
- **-RE**: Repeat-explore removed.
- **-HR**: Item factor generating removed.

- **-LS**: Long-short term removed.
- **-SA**: Self-attention removed.
- **-PE**: Position embedding removed.
- **-FL**: Use entropy loss to replace focal loss.
- **+PH**: Add projection head to CLTAR-GNN.

As for +PH, the main idea is from the study of SimCLR, which adds a projection head (PH) to improve the performance. The projection head is a MLP layer which projects session vectors into a different feature space to calculate the contrastive loss. The test results are evaluated using HR@20 and MRR@20, and are summarized in Table 4.

Table 4. Ablation results for CLARE

Methods	Yoochoose 1/64		Diginetica	
	HR@20	MRR@20	HR@20	MRR@20
CLTAR-GNN	**72.07**	**31.67**	**54.49**	**19.02**
-CL	71.73	31.17	53.92	<u>18.98</u>
-GGNN	71.97	31.26	54.33	18.89
-RE	72.01	31.14	54.31	18.91
-HR	71.95	31.15	54.25	18.82
-LS	71.95	31.21	<u>54.36</u>	18.88
-SA	70.31	30.98	52.65	18.66
-PE	70.69	29.40	53.54	18.65
-FL	71.82	31.04	54.23	18.88
+PH	71.66	<u>31.55</u>	53.75	18.96

As seen in Table 4, self-attention (-SA) and position embedding (-PE) have the highest contribution on the performance of CLTAR-GNN, which is consistent with our previous experimental conclusion. Besides, contrastive learning (-CL) also has a great impact on the performance of CLTAR-GNN. The impact of focal loss (-FL) is relatively smaller, but also make positive contributions to the performance, which indicate that focal loss could help resolve data imbalance problem. Projection head (+PH) cannot improve the performance in session-based recommendation.

4.4. Comparison with Different Model Variant

Different variants of the proposed CLTAR-GNN are also verified in this section. Six variants are summarized as below:

- **CLTAR-GNN-AVG**: Use the average of self-attention output as the representation of session embedding.
- **CLTAR-GNN-N1**: Use the last item embedding of self-attention output as the representation of session embedding.
- **CLTAR-GNN-N2**: Use average of the last two item embeddings of self-attention output as the representation of session embedding.
- **CLTAR-GNN-S**: Only use single attention mechanism in item generating module.
- **CLTAR-GNN-M:** The session embeddings in session generation layer will share weights with adjust function in item factor generating module.

– **CLTAR-GNN-SM**: The structure considers both single attention design and weight sharing strategy.

CLTAR-GNN-S is used to evaluate whether the design of two attention mechanisms in item factor generating module is necessary. CLTAR-GNN-M investigates whether incorporating repeat-explore and head-tail modules in early stage will obtain better performances. In order to verify those assumptions, experiments are conducted based on CLTAR-GNN and its six variants, and the results could be seen in Table 5.

Table 5. Experimental results for different variants

Methods	Yoochoose 1/64		Diginetica	
	HR@20	MRR@20	HR@20	MRR@20
CLTAR-GNN-AVG	71.46	30.54	53.61	18.55
CLTAR-GNN-N1	71.72	31.43	53.54	18.66
CLTAR-GNN-N2	**72.07**	**31.67**	**54.43**	**19.02**
CLTAR-GNN-S	71.56	30.02	53.49	18.37
CLTAR-GNN-M	68.36	27.62	51.98	17.44
CLTAR-GNN-SM	68.64	27.36	51.35	17.10

As shown in Table 5, CLTAR-GNN-N2 is the best choice for calculating session embeddings. CLTAR-GNN-AVG may be unable to make accurate recommendations because it focuses on the global interest, and CLTAR-GNN-N1 only uses one item vector which may be unrepresentative. Therefore, CLTAR-GNN-N2 is a better choice which doesn't have these problems. Besides, different structure-based variants of CLTAR-GNN (S, M, SM) could not obtain better results in terms of HR@20 and MRR@20 on both datasets, which further indicates the effectiveness of the proposed CLTAR-GNN.

5. Conclusion

In this paper, we propose a novel model called Contrastive Learning based Tail Adjusted Repeat Graph Neural Network (CLTAR-GNN) for Session-based Recommendation. We design a series of innovations, which include: incorporate self-attention, position embedding and graph neural network into a unified framework to consider both long-term order information and high-order complex correlations among items; incorporate repeat-explore and head-tail into a unified framework to consider users' different behavior patterns in long-tail and short-head items; consider long-short term correlations for all session representation modelling; use contrastive learning to extract self-supervised signals from raw data, thus get high-quality session representations and make more accurate recommendations. Extensive experiments conducted on two public datasets Yoochoose 1/64 and Diginetica show that CLTAR-GNN evidently outperforms the state-of-the-art session-based recommendation methods. Besides, Experiments also exhibit that self-attention with position embedding is essential important for the proposed model to capture long-term dependencies and order information. The negative influence of data sparsity and data imbalance problems in session recommendations have also been confirmed through extensive ex-

periments, and contrastive learning framework is verified as a good solution to solve the above problems to a certain extent.

In future, it would be worth exploring and employing different contrastive learning frameworks in CLTAR-GNN. In another aspect, users' behavior patterns will be further investigated from the perspective of consumer behavior theory. With the guidance of domain theory, prior knowledge of users and sessions will be also taken into considerations to further improve the performance and make the results more explainable.

Acknowledgments. This research was supported by Guangdong Natural Science Foundation General Project under Grant No. 2024A1515011793.

References

1. Benson, A., Kumar, R., Tomkins, A.: Modeling user consumption sequences. In: WWW '16: In Proceedings of the 25th International Conference on World Wide Web. pp. 519–529 (2016)
2. Chen, T., Wong, R.C.W.: Handling information loss of graph neural networks for session-based recommendation. In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp. 1172–1180 (2020)
3. Chen, T., Kornblith, S., Norouzi, M., Hinton, G.: A simple framework for contrastive learning of visual representations. In: International conference on machine learning. pp. 1597–1607. PMLR (2020)
4. Davidson, J., Liebald, B., Liu, J., Nandy, P., Van Vleet, T., Gargi, U., Gupta, S., He, Y., Lambert, M., Livingston, B., et al.: The youtube video recommendation system. In: Proceedings of the fourth ACM conference on Recommender systems. pp. 293–296 (2010)
5. Fang, J.: Session-based recommendation with self-attention networks. arXiv preprint arXiv:2102.01922 (2021)
6. Gupta, P., Garg, D., Malhotra, P., Vig, L., Shroff, G.: Niser: Normalized item and session representations to handle popularity bias. arXiv preprint arXiv:1909.04276 (2019)
7. He, R., McAuley, J.: Fusing similarity models with markov chains for sparse sequential recommendation. In: In Proceedings of 2016 IEEE 16th International Conference on Data Mining (ICDM'16), pp. 191–200 (2016)
8. Hidasi, B., Karatzoglou, A., Baltrunas, L., Tikk, D.: Session-based recommendations with recurrent neural networks. arXiv preprint arXiv:1511.06939 (2015)
9. Hidasi, B., Karatzoglou, A., Baltrunas, L., Tikk, D.: Session-based recommendations with recurrent neural networks. In: In Proceedings of 2016 International Conference on Learning Representation (ICLR'16) (2016)
10. Hidasi, B., Quadrana, M., Karatzoglou, A., Tikk, D.: Parallel recurrent neural network architectures for feature-rich session-based recommendations. In: Proceedings of the 10th ACM conference on recommender systems, pp. 241–248 (2016)
11. Li, J., Ren, P., Chen, Z., Ren, Z., Lian, T., Ma, J.: Neural attentive session-based recommendation. In: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, pp. 1419–1428 (2017)
12. Li, J., Ren, P., Chen, Z., Ren, Z., Lian, T., Ma, J.: Neural attentive session-based recommendation. In: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management. pp. 1419–1428 (2017)
13. Li, Y., Tarlow, D., Brockschmidt, M., Zemel, R.: Gated graph sequence neural networks. arXiv preprint arXiv:1511.05493 (2015)
14. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. In: Proceedings of the IEEE international conference on computer vision. pp. 2980–2988 (2017)

15. Liu, Q., Zeng, Y., Mokhosi, R., Zhang, H.: Stamp: short-term attention/memory priority model for session-based recommendation. In: Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining. pp. 1831–1839 (2018)
16. Liu, S., Zheng, Y.: Long-tail session-based recommendation. In: Fourteenth ACM conference on recommender systems. pp. 509–514 (2020)
17. Pang, Y., Wu, L., Shen, Q., Zhang, Y., Wei, Z., Xu, F., Chang, E., Long, B., Pei, J.: Heterogeneous global graph neural networks for personalized session-based recommendation (2022)
18. Perozzi, B., Al-Rfou, R., Skiena, S.: Deepwalk: Online learning of social representations. In: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 701–710 (2014)
19. Ren, P., Chen, Z., Li, J., Ren, Z., Ma, J., De Rijke, M.: Repeatnet: A repeat aware neural recommendation machine for session-based recommendation. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 33, pp. 4806–4813 (2019)
20. Rendle, S., Freudenthaler, C., Gantner, Z., Schmidt-Thieme, L.: Bpr: Bayesian personalized ranking from implicit feedback. arXiv preprint arXiv:1205.2618 (2012)
21. Rendle, S., Freudenthaler, C., Schmidt-Thieme, L.: Factorizing personalized markov chains for next-basket recommendation. In: Proceedings of the 19th international conference on World wide web. pp. 811–820 (2010)
22. Sarwar, B., Karypis, G., Konstan, J., Riedl, J.: Item-based collaborative filtering recommendation algorithms. In: Proceedings of the 10th international conference on World Wide Web. pp. 285–295 (2001)
23. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in neural information processing systems 30 (2017)
24. Wang, M., Ren, P., Mei, L., Chen, Z., Ma, J., de Rijke, M.: A collaborative session-based recommendation approach with parallel memory modules. In: Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval. pp. 345–354 (2019)
25. Wang, S., Zhang, Q., Hu, L., Zhang, X., Wang, Y., Aggarwal, C.: Sequential/session-based recommendations: Challenges, approaches, applications and opportunities (2022)
26. Wu, S., Tang, Y., Zhu, Y., Wang, L., Xie, X., Tan, T.: Session-based recommendation with graph neural networks. In: Proceedings of the AAAI conference on artificial intelligence, vol. 33, pp. 346–353 (2019)
27. Wu, S., Tang, Y., Zhu, Y., Wang, L., Xie, X., Tan, T.: Session-based recommendation with graph neural networks. In: Proceedings of the AAAI conference on artificial intelligence. vol. 33, pp. 346–353 (2019)
28. Xie, X., Sun, F., Liu, Z., Wu, S., Gao, J., Ding, B., Cui, B.: Contrastive learning for sequential recommendation. In: In Proceedings of IEEE 38th International Conference on Data Engineering. pp. 1259–1273 (2022)
29. Xu, C., Zhao, P., Liu, Y., Sheng, V.S., Xu, J., Zhuang, F., Fang, J., Zhou, X.: Graph contextualized self-attention network for session-based recommendation. In: IJCAI. vol. 19, pp. 3940–3946 (2019)
30. Yang, L., Luo, L., Li, F., Zhang, X., Zhang, X.: Dagnn: Demand-aware graph neural networks for session-based recommendation. In: SIGIR (2022)
31. Yin, H., Cui, B., Li, J., Yao, J., Chen, C.: Challenging the long tail recommendation. In: In Proceedings of the VLDB Endowment. vol. 5 (2012)
32. Yu, F., Zhu, Y., Liu, Q., Wu, S., Wang, L., Tan, T.: Tagnn: target attentive graph neural networks for session-based recommendation. In: Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval. pp. 1921–1924 (2020)
33. Zhong, S., Qin, J., Huang, Z., Li, D.: Cem:machine-human chatting handoff via causal-enhance module (2022)

34. Zhou, K., Wang, H., Zhao, W.X., Zhu, Y., Wang, S., Zhang, F., Wang, Z., Wen, J.R.: S3-rec: Self-supervised learning for sequential recommendation with mutual information maximization. In: Proceedings of the 29th ACM International Conference on Information & Knowledge Management. pp. 1893–1902 (2020)

Daifeng Li is a professor at School of Information Management, Sun Yat-sen University (SYSU). He has working experience at Tsinghua University as post phd and Baidu as Senior Engineer. His research interests include Information Sciences, Data Science, Information Systems and Artificial Intelligent.

Tianjunzi Tian is a master student at Department of Information Management at Nanjing University, majoring in information science. Her research interests include data mining, data analysis and management science.

Zhaohui Huang is an undergraduate student at School of Information Management, Sun Yat-sen Univeristy (SYSU), majoring in information science. His research interests include software engineering, data mining, data analysis and management science.

Xiaowen Lin is a master student at Department of Information Management at Nanjing University, majoring in information science. Her research interests include data mining, data analysis and management science.

Dingquan Chen is a professor at School of Information Management, Sun Yat-sen University (SYSU). His research interests include Library and Information Science.

Andrew Madden began his research career as an agricultural ecologist, but became interested in Information Science after working on a project to develop computer-based materials for teaching about land use. He has worked as a researcher at the UK's Natural Resources Institute, Sheffield Hallam University, Sheffield University, and a senior researcher in the School of Information Management at Sun Yat-sen University. His current research interests are in information behavior, the evolution of decision-making systems and information literacy.

Received: November 01, 2024; Accepted: December 02, 2024.