Dual-Ganularity Contrastive Learning for Session-based Recommendation

Zihan Wang[1], Gang Wu[1], and Haotong Wang[1]

Northeastern University, China
{2101816@stu.neu.edu.cn, wugang@mail.neu.edu.cn, 2171931@stu.neu.edu.cn}

Abstract. Session-based recommendation systems(SBRS) are more suitable for the current e-commerce and streaming media recommendation scenarios and thus have become a hot topic. The data encountered by SBRS is typically highly sparse, which also serves as one of the bottlenecks limiting the accuracy of recommendations. So Contrastive Learning(CL) is applied in SBRS owing to its capability of improving embedding learning under the condition of sparse data. However, existing CL strategies are limited in their ability to enforce finer-grained (e.g., factor-level) comparisons and, as a result, are unable to capture subtle differences between instances. More than that, these strategies usually use item or segment dropout as a means of data augmentation which may result in sparser data and thus ineffective self-supervised signals.
By addressing the two aforementioned limitations, we introduce a novel multi-granularity CL framework. Specifically, two extra augmented embedding convolution channels with different granularities are constructed and the embeddings learned by them are compared with those learned from original view to complete the CL tasks. At factor-level, we employ Disentangled Representation Learning to obtain finer-grained data(e.g. factor-level embeddings), with which we can construct factor-level convolution channels. At item-level, the star graph is deployed as the augmented data and graph convolution on it can ensure the effectiveness of self-supervised signals.
Compare the learned embeddings of these two views with the learned embeddings of the basic view to achieve CL at two granularities. Finally, the more precise item-level and factor-level embeddings obtained are referenced to generate personalized recommendations for the user.
The proposed model is validated through extensive experiments on two benchmark datasets, showcasing superior performance compared to existing methods.

Keywords: Session-based Recommendation· Contrastive Learning· Disentangled Representation Learning

1 Introduction

Session-based recommendation (SBR) has gained significant attention recently due to its ability to provide recommendations solely based on information from an anonymous session, and it is highly sensitive to users' short-term interests.

The items that a user interacts with in a short period of time are arranged in chronological order to form a session. Session data bears some resemblance to text data in Natural Language Processing (NLP), and as a result, some classic NLP frameworks including Recurrent Neural Network(RNN)[1,2,3,6], Attention Mechanism[4,5,12,13] and Graph Neural Network(GNN)[7,8,9,10,11,15] have been adapted to SBR.

GNN-based models have been found to be more effective than other models because they can better model the complex relationships that exist between items. However, even with GNN-based models, SBR still faces the challenge of insufficient data to train accurate item embeddings, which can lead to suboptimal recommendation performance.

Contrastive Learning (CL) in Self-Supervised Learning[14,16,18] is widely regarded as a solution to the problem of data sparsity. The process of CL can be divided into three steps. Typically, CL models first create a new view through artificial data augmentation based on the original view and then learn embeddings from both the original view and the augmentation view. Finally, the embeddings learned from the latter are partitioned into positive and negative samples relative to the embeddings learned from the former. By comparing the differences and similarities between them, models can adjust the embeddings during training following the principle that positive samples are close to each other and negative samples are mutually exclusive. As a result, the model can learn more accurate embeddings than with supervised training alone. Therefore, some researchers have applied CL to the SBR and have achieved promising results[17,20,21,19]. Nevertheless, we have observed that existing CL models often exhibit two flaws. First, existing CL methods are typically limited to coarse-grained comparisons at the item-level and/or session-level, often overlooking finer-grained relationships between instances. Second, these methods often rely on item or segment dropout as a form of data augmentation, which can exacerbate data sparsity and consequently, less effective self-supervised signals.

By addressing the two aforementioned limitations, we introduce a novel dual-granularity CL framework. Our approach typically involves incorporating two additional augmentation views, one at the item-level and the other at the factor-level, besides the original item-level view. The embeddings acquired from two augmentation views are compared with those acquired from the original view to finish CL tasks under two granularities.

At factor-level, to address the issue of fine-grained factor (e.g. brand and color) labels being often absent in SBR data, we propose the use of Disentangled Representation Learning(DRL) to obtain independent latent factor-level embeddings corresponding to items, which can replace the missing labels. So factor-level convolution channels that operate independently can be constructed and then can be compared with the factor-level embeddings converted from the embeddings learned in the original view to finish the factor-level CL. At item-level, the star graph has been planted as an augmentation of the basic view. Star graph has the inclusion of an extra satellite node and non-adjacent nodes can communicate with each other via the satellite node, thereby enabling the ac-

quisition of more information. Unlike traditional data augmentation techniques, it avoids further exacerbating data sparsity by promoting nodes to learn more information. Likewise, we compared the item-level embedding learned from the original view and star graph to finish item-level CL.

Moreover, we leverage the learned item-level and factor-level embeddings to model the user's overall and specific latent factor interests, respectively, in order to predict the user's next interaction at two granularities. In the end, we propose our model Dual-Ganularity Contrastive Learning for Session-based Recommendation(DGCL-GNN) and summarize our main contributions as follows:

- We identify and address two challenges in existing contrastive learning methods for SBR, and propose a dual-granularity contrastive learning framework to improve the model's ability to learn embeddings.
- We innovatively introduce disentangled representation learning and star map augmentation to help us complete two granular comparative learning tasks.
- Eventually, we propose out model DGCL-GNN, and extensive experiments show that the proposed model has achieve statistically significant improvements on benchmark datasets.

This paper is organized as follows. 2 describes the related work for SBR and CL. 3 give formal definitions of the SBR. 4 presents the details of our model DGCL-GNN. 5 includes various experiments to demonstrate the effectiveness of our model. Finally, 6 presents our conclusions and suggestions for future studies.

2 Related Work

2.1 Session-based Recommendation

Session-based recommendation(SBR) has gained considerable research interest in recent years. Its main objective is to provide personalized recommendations for a user's immediate needs, based on the current session context, without the reliance on any past user-item interaction history.

Session-based recommendation models in their early days relied on markov chains for predictions[**?**]. Nevertheless, these models were constrained by their inability to transfer user interest beyond the previous item, thereby rendering the model's performance unsatisfactory. To overcome this limitation, some researchers have turned to neural network models inspired by NLP, which have shown promise in improving the effectiveness of SBR. Recurrent Neural Network(RNN)[22] was first noticed because it can capture sequential dependencies in a session, which is essential for SBR. However, RNN-based models[1,2,3,6] overlooks the global information that exists within the entire session, which can lead to performance limitations. Some scholars have attempted to incorporate the attention mechanism[24] to capture global information and proposed some attention-based models[4,5,12,13]. Then Graph Neural Network(GNN)[23,25] model [7,8,9,10,11,15]emerged as a promising solution, showing a significant performance advantage. This is due

to the GNN's strong ability to capture complex relationships between items, enabling enhanced representation learning capabilities of the model. Disen-GNN[26] has introduced DRL, which has become popular in other fields[27,28,31], into SBR for the first time. This creative approach refines the SBR problem to a fine-grained level, prompting a deeper analysis of the SBR problem.

Although GNN has shown great advantages in SBR, it still faces a significant performance bottleneck. This is due to the sparsity of the SBR data, resulting in a sparsity of the graph and fewer connections between nodes. As a result, the information transmitted between nodes is limited, making it difficult for nodes to learn accurate embeddings.

2.2 Contrastive Learning

Contrastive learning(CL) is a type of unsupervised learning that aims to learn a representation space where similar samples are mapped to nearby points, while dissimilar samples are mapped to distant points. In CL, the model learns by contrasting pairs of samples, where one sample is considered as a "positive" sample and the other as a "negative" sample. The objective is to minimize the distance between the positive pair while maximizing the distance between the negative pair. CL has been successfully applied in various fields[30,**?**,34], such as computer vision and natural language processing, to improve the quality of learned embeddings and enhance the performance of downstream tasks.

In the field of recommendation systems like collaborative filtering recommendation and sequential recommendation, CL has also been widely used in recent years[36,**?**,29,35]. Some studies have also applied CL to the session-based recommendation scenario[17,20,21,19], where it has shown promising results in improving the performance of session-based recommendation models.

The CL task aims to address a pain point of SBR, as the data in this area is often sparse. Therefore, we firmly believe that improving CL is one of the key directions for enhancing SBR's performance in the future.

3 Preliminaries

In this section, we introduce the formal definitions of the session-based recommendation problem. Let $\mathcal{I} = \{v_1, ..., v_N\}$ denote the set of all items in the dataset and $\boldsymbol{N}$ is the number of items. Session-based recommendation is to predict the next-item of an anonymous session, consisting of a sequence of interacted items. Formally speaking, a session is represented as $\boldsymbol{s} = \left\{v_{(s,1)}, ..., v_{(s_n)}\right\}$, where $\boldsymbol{n}$ is the total length of the session. The task of the SBR models is to analyze the user's interests revealed by interactions in $\boldsymbol{s}$ and predict the next item $v_{(s,n+1)}$ that the user most likely interacts with. Finally, we compared the simulation of user interests with the characteristics of each item to calculate the probability of its potential occurrence $p(v_i|s)$, which represents the score of v_i in this recommendation. We selected the Top-K items with the highest scores and recommended them to the user.

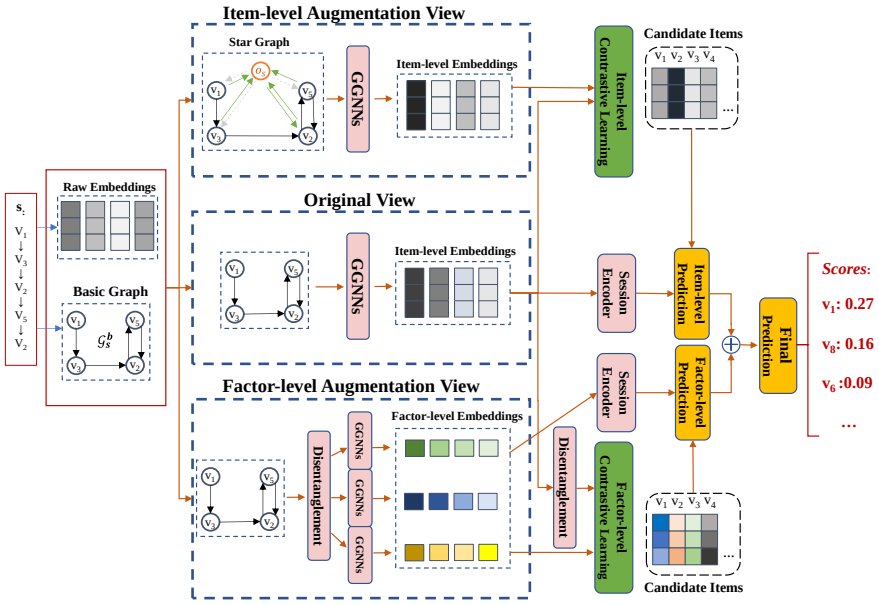


Fig. 1: Overview of DGCL-GNN

4 Methodology

Next, we describe the framework of our proposed model DGCL-GNN. In general, we elaborate on three components of our model mainly including the basic recommendation module, the factor-level CL module and the item-level CL module.

4.1 Initialization

Basic Session Graph . As in the previous work, we use a directed session graph $\mathcal{G}_s^o =< \mathcal{V}_s^o, \mathcal{E}_s^o >$ to store the sequential relationship of each item in the session $\boldsymbol{s}$. That is, when the next interaction of v_i is v_j, we set the $\boldsymbol{i}$ row $\boldsymbol{j}$ column of the adjacency matrix $\mathcal{A}_s^o$ to 1, which means that there is a pointing relationship.

Item-level Embeddings . We first embed items in $\boldsymbol{s}$ into the same space to obtain the corresponding item-level embeddings $\boldsymbol{e}^s = \left\{e_{(s,1)}, ..., e_{(s,n)}\right\}$.

Factor-level Embeddings . We adopt DRL to acquire the representations of the independent latent factors underlying the observed data, which corresponds to the fine-grained, factor-level embeddings of the items.

The input of DRL is the item-level embeddings. Let $\boldsymbol{e}_i \in \mathbb{R}^d$ represents a $\boldsymbol{d}$ dimension item-level embedding. Then DRL computes $\mathcal{K}$ factor-level embeddings from $\boldsymbol{e}_i$ by re-embedding it into corresponding spaces. The factor-level embedding $\boldsymbol{f}_{(k,i)}$ on factor k is computed with Equation 1.

$$\boldsymbol{f}_i^k = \sigma(W_f^k e_i) + \boldsymbol{b}_f^k (1 \leq k \leq \mathcal{K}) \tag{1}$$

Here, $\boldsymbol{W}_t^f \in \mathbb{R}^{d \times d_f}$ and $\boldsymbol{b}_t^f \in \mathbb{R}^{d_f}$ are the weight matrix and bias on factor k. And $d_f = \lfloor \frac{d}{k} \rfloor$ is the dimension of a factor-level embedding.

In order to avoid redundant information between factors, DRL uses the following loss function as the learning objective to generate independent factor-level embeddings.

$$\mathcal{L}_d = \sum_{k}^{\mathcal{K}} \sum_{t \neq k}^{\mathcal{K}} dCor(f^k, f^t) \tag{2}$$

dCor is a formula to measure the correlation between variables in different spaces. For other details, please refer to [37].

4.2 Embedding Learning in Three Channels

We introduce convolutions in three channels in our model, which aim to learn embeddings for three different views.

Original view . The input to this view is the raw item-level embedding $\boldsymbol{e}^s = e_{(s,1)}, ..., e_{(s,n)}$ and the raw session adjacency matrix $\mathcal{A}_s^o$. In Original view, we use the most commonly used Graph Gated Neural Network(GGNN)[38] method to update the item-level embedding.

$$\begin{aligned} c_{(s,i)}^l = Concat(&W_{in} \mathcal{A}_{(s,in)}^o ([x_{(s,1)}^l, ..., x_{(s,n)^l}]^\top) + b_{in}, \\ &W_{out} \mathcal{A}_{(s,out)}^o ([x_{(s,1)}^l, ..., x_{(s,n)^l}]^\top) + b_{out}) \end{aligned} \tag{3}$$

$$z_{(s,i)}^l = \sigma(W_z c_{(s,i)}^l + U_z x_{(s,i)}^{l-1}) \tag{4}$$

$$r_{(s,i)}^l = \sigma(W_r c_{(s,i)}^l + U_r x_{(s,i)}^{l-1}) \tag{5}$$

$$\tilde{x}_{(s,i)}^l = \tanh(W_h c_{(s,i)}^l + U_h (r_{(s,i)}^l \odot x_{(s,i)}^{l-1})) \tag{6}$$

$$x_{(s,i)}^l = (1 - z_{(s,i)}^l) \odot h_i^{l-1} + z_{(s,i)}^l \odot \tilde{x}_{(s,i)}^l \tag{7}$$

Where $\boldsymbol{c}_{(s,i)}^l$ is the information that $v_{(s,i)}$ can learn from its adjacent nodes in $\boldsymbol{l}$-th layer, $A_{(s,in}^o$ and $A_{(s,out)}^o$ are the in-degree and out-degree matrices corresponding to the adjacency matrix A_s^o. $x_{(s,i)^l}$ represents the embedding of node $v_{(s,i)}$ at layer l and $x_{(s,i)}^0 = e_{(s,i)}$. $z_{(s,i)}^l$ and $r_{(s,i)}^l$ are respectively the update gate and reset gate in the gating mechanism. $\boldsymbol{\sigma}$ and **tanh** are activation functions. Note that, W_{in}, W_{out}, W_z, W_r, W_h, U_z, U_r, $U_h \in \mathbb{R}^{d \times d}$ are all learnable weights and b_{in}, $b_{out} \in \mathbb{R}^d$ are biases.

Finally, we get the updated item-level embedding $\left\{ e_{(s,1)}^o, ..., e_{(s,n)}^o \right\}$.

Factor-level Augmentation View This view is subdivided into $\mathcal{K}$ independent sub-channels inside, which are respectively responsible for learning the embeddings corresponding to $\mathcal{K}$ hidden factors. In order to reduce the redundant information between sub-channels. According to 2, measure the distance similarity between factor-level embeddings of each item to generate loss $\mathcal{L}_d$, and finally we hope that each embedding is as independent as possible.

Since items may be similar in one latent factor but not in another, we transform the original session graph $\mathcal{G}_s^o$ to represent the relationship between items on different latent factors. We replace the weight in $\mathcal{A}_s^o$ with the corresponding similarity value to generete factor-level adjacency matrixes $\left\{\mathcal{A}_s^{(f,1)}, ..., \mathcal{A}_s^{(f,K)}\right\}$. $\mathcal{A}_s^{(f,k)}$ is the factor-level adjacency matrix on factor $\boldsymbol{k}$.

$$a_{(s,ij)}^{(f,k)} = \frac{e_{(s,i)}^k e_{(s,j)}^k}{||e_{(s,i)}^k|| \cdot ||e_{(s,j)}^k||} \tag{8}$$

$a_{(s,ij)}^{(f,k)}$ corresponds to the value of row $\boldsymbol{i}$ and column $\boldsymbol{j}$ in $\mathcal{A}_s^{(f,k)}$.

The input of $\boldsymbol{k}$-th sub-channel is ,$\left\{e_{(s,1)}^k, ..., e_{(s,n)}^k\right\}$, the raw factor-level embeddings of items on hidden factor $\boldsymbol{k}$ and the factor-level adjacent matrix $\mathcal{A}_s^{(f,k)}$. Similar to the original channel, we perform convolution to learn the factor-level embedding based on the adjacency relationships stored in $\mathcal{A}_s^{(f,k)}$ to generate the final output of the sub-channel $\left\{e_{(s,1)}^{(f,k)}, ..., e_{(s,n)}^{(f,k)}\right\}$.

Item-level Augmentation View We decide to choose the star graph as the augmentation graph of the original graph at the item-level. In details, we set a satellite node $\boldsymbol{O}_s$ for session $\boldsymbol{s}$ and the embedding of it $\boldsymbol{e}_{(s,O)}$ is set as the average pooling of the items in $\boldsymbol{s}$.

$$\boldsymbol{e}_{(s,O)} = \frac{\sum_1^n e_{(s,i)}}{n} \tag{9}$$

For the connections between the satellite node and other nodes, we adopt a completely random method that satellite node has an equal probability θ of pointing or being pointed to with other nodes. By implementing the above operations, we can initialize a star graph $\mathcal{G}_s^*$. Furthermore, we have the corresponding adjacency matrix $\mathcal{A}_s^*$. Then $\boldsymbol{e}^s = \left\{e_{(s,1)}, ..., e_{(s,n)}\right\}$ and $\mathcal{A}_s^*$ are sent into the convolution channel and item-level embeddings are updated as $\left\{e_{(s,1)}^*, ..., e_{(s,n)}^*\right\}$.

4.3 Contrastive Learning

Once we have learned three embeddings from three views, we can leverage them to accomplish dual-granularity CL tasks. Combining two Contrastive Learning (CL) methods can enhance the embedding learning ability of our model.

Factor-level Contrastive Learning To begin, we re-embed the output of the original view $\left\{e^o_{(s,1)}, ..., e^o_{(s,n)}\right\}$ into the corresponding factor-level spaces to get factor-level embeddings $\left\{\left\{e^{(o,1)}_{(s,1)}, ..., e^{(o,1)}_{(s,n)}\right\}, ..., \left\{e^{(o,K)}_{(s,1)}, ..., e^{(o,K)}_{(s,n)}\right\}\right\}$. Then we use the factor-level embeddings learned by the factor-level augmentation view $\left\{\left\{e^{(f,1)}_{(s,1)}, ..., e^{(f,1)}_{(s,n)}\right\}, ..., \left\{e^{(f,K)}_{(s,1)}, ..., e^{(f,K)}_{(s,n)}\right\}\right\}$ to compare with the former.

To perform CL on $\mathcal{K}$ latent factors, we utilize K sub-channels. Likewise, we describe the operations on the $\boldsymbol{k}$-th channel as an example. A standard binary cross-entropy (BCE) loss function has been chosen as our learning objective to measure the difference between the two.

$$\mathcal{L}^{(f,k)}_c = -\log\ \sigma(H(e^{(o,k)}_{(s,i)}, e^{(f,k)}_{(s,i)}) - \log\ \sigma(1 - H(e^{(o,k)}_{(s,i)}, e^{(o,k)}_{(s,j)})(i \neq j) \tag{10}$$

The first half of the comparison involves positive examples, while the second half involves negative examples. and $\boldsymbol{H} : \mathbb{R}^{d_f} \times \mathbb{R}^{d_f} \to \mathbb{R}^{d_f}$is the discriminator function that takes two vectors as the input and then scores the agreement. between them.

Finally, we aggregate the losses generated by the K channels to obtain the total loss for factor-level CL.

$$\mathcal{L}^F_c = \sum_k^K L^{(f,k)}_c \tag{11}$$

Item-level Contrastive Learning We utilize the two item-level embeddings learned from the basic channel $\left\{e^o_{(s,1)}, ..., e^o_{(s,n)}\right\}$ and the item-level augmentation channel $\left\{e^*_{(s,1)}, ..., e^*_{(s,n)}\right\}$ to conduct item-level CL. Similarly, we adopt BCE loss as the learning object of item-level CL.

$$\mathcal{L}^I_c = -\log\ \sigma(H(e^o_{(s,i)}, e^o_{(s,i)}) - \log\ \sigma(1 - H(e^*_{(s,i)}, e^*_{(s,j)})(i \neq j) \tag{12}$$

The total loss of CL is set as follows:

$$\mathcal{L}_c = \alpha * \mathcal{L}^I_c + (1 - \alpha) * \mathcal{L}^F_c \tag{13}$$

α is a hyperparameter controlling the ratio of item-level and factor-level CL losses.

4.4 Session Embedding

The item-level session embedding need to be generated to represent the user's overall preference for items. Inspired by Disen-GNN, we generate factor-level session embeddings in addition, which represent users' interests for specific latent factors. We enrich the modeling of user interests with session embeddings at two granularities.

Item-level Session Embedding The soft attention is employed as the session encoder.

$$\alpha_{(s,i)} = q^{\top}\sigma(W_s^1 e^o_{(s,i)} + W_s^2 e^o_{(s,n)}) \tag{14}$$

$$e_s^l = \sum_{i=1}^{n} \alpha_{(s,i)} e^o_{(s,i)} \tag{15}$$

$$e_s^o = \boldsymbol{W}_s^3[e_s^l, e_s^g] \tag{16}$$

where $\boldsymbol{q} \in \mathbb{R}^d$, $\boldsymbol{W}_s^1 \in \mathbb{R}^{d\times d}$, $\boldsymbol{W}_s^2 \in \mathbb{R}^{d\times d}$ and $\boldsymbol{W}_s^3 \in \mathbb{R}^{d\times 2d}$ are learnable parameters. e_s^l and e_s^g represent the session's local and global preferences for factor t respectively. And e_s^l is $e^o_{(s,n)}$, the last item's embedding, such setting can make the model pay more attention to the last clicked item, because usually the last item is the most related to the item that the user finally needs.

Factor-level Session Embedding Similarly, we use the same soft attention to compute the user's preference for $\mathcal{K}$ independent latent factors. The process will not be repeated, and finally we concat the user's preferences for $\mathcal{K}$ latent factors as $e_s^f = [e_s^1, ..., e_s^K]$.

4.5 Prediction and Optimization

As mentioned earlier, we score candidate items separately based on the item-level and factor-level interests, and then combine the results of them to make the recommendation. We use the inner-product as the scoring criterion. Note that, in order to calculate the factor-level scores, the item-level embeddings of the candidate items need to be converted into their corresponding factor-level embeddings, and then the inner-product is computed with the corresponding user's factor-level interests. The final scores of all items are as follows:

$$\hat{y_i} = \frac{\hat{y_i^I} + \hat{y_i^F}}{2} \tag{17}$$

$\hat{y_i^I}$ and $\hat{y_i^F}$ represent the scoring results of item-level and factor-level respectively. $\hat{y_i}$ stores the final scores. For the prediction loss, we use the cross-entropy as the loss function, which has been extensively used in the recommendation system:

$$\mathcal{L}_p = -\sum_{i=1}^{N} y_i log(\hat{\boldsymbol{y}}_{\boldsymbol{i}}) + (1 - y_i) log(1 - \hat{\boldsymbol{y}}_i) \tag{18}$$

So, the loss of the whole model consists of three parts: CL, the prediction and DRL. β_1 and β_2 are controlling their proportions.

$$\mathcal{L} = \mathcal{L}_p + \beta_1 \cdot \mathcal{L}_c + \beta_2 \cdot \mathcal{L}_d \tag{19}$$

Adam is adopted as the optimization algorithm to analyze the loss $\mathcal{L}$.

5 Experiments

In this section, we introduce the rich experimental content and outline some of the basic experimental settings.

5.1 Experiments Settings

Datasets To verify the effectiveness of DGCL-GNN, we conducted experiments on two commonly used datasets in a session-based recommendation system, *Yoochoose 1/64*[1], and *Diginetica*[2]. We follow the commonly adopted procedures to preprocess the datasets. Specifically, we removed the sessions with only one item and infrequent items that appear less than 5 times in each dataset. Similar to previous work, we divided training data and test data by time. The statistics of the three datasets are exhibited in Table 1.

Table 1: Statistical results of datasets

Statistics	Yoochoose1/64	Diginetica
#interactions	557,248	982,961
#training sess	369,859	719,470
#test sess	55,898	60,858
#items	16,766	43,097
#avg. length	6.16	5.12

Baselines To demonstrate the comparative performance of DGCL-GNN, we choose several representative and/or state-of-the-art models. They can be categorized into three types: (1) **Non-GNN models**: NARM, GRU4Rec, and STAMP; (2) **Normal GNN models**: SR-GNN and Disen-GNN; (3) **GNN models with CL**: DHCN and COTREC.

- **GRU4REC**[1] adapts GRU from NLP to SBR. As an RNN-based model, it only cares about sequential relationships between items.
- **NARM**[4] combined attention mechanism with Gated Recurrent Unit(GRU) to consider both global relationships and sequential relationships to make recommendations.
- **STAMP**[5] emphasizes the impact of the short time and it designed a special attention mechanism with MLP.
- **SR-GNN**[11] introduced GGNN to obtain item embeddings by information propagation and also employs a soft-attention mechanism to get the session embedding for making recommendations.

[1] http://2015.recsyschallenge.com/challege.html
[2] http://cikm2016.cs.iupui.edu/cikm-cup

- **Disen-GNN**[26] deployed DRL into SBR to learn the latent factor-level embeddings. Then use the GGNN in each factor channel to learn factor-level session embeddings. Its model structure is very simple, but it has become a state-of-the-art model.
- **DHCN**[17] note that the traditional CL strategy has limited effect in SBR due to the extreme data sparsity. Therefore, it proposed a new type of CL network, which realizes data augmentation by learning inter-session information.
- **COTREC**[21] is an improved variant of DHCN, it integrated the idea of co-training into CL by adding divergence constraints to DHCN's CL module to comprehensively learn inter and intra-session information.

Evaluation Metrics Following our baselines, we chose widely used ranking metrics P@K(Precise) and M@K(Mean Reciprocal Rank) to evaluate the recommendation results where K is 10 or 20.

Table 2: Comparing the prediction performance of DGCL-GNN with the baselines. The best results in them are highlighted in bold, and the second-best results are underlined.

Method	Yoochoose1/64				Diginetica			
	P@10	M@10	P@20	M@20	P@10	M@10	P@20	M@20
NARM	0.5920	0.2495	0.6811	0.2855	0.3544	0.1513	0.4970	0.1618
GRU4Rec	0.5011	0.1789	0.6063	0.2288	0.1789	0.0730	0.2939	0.0829
STAMP	0.6190	0.2583	0.6874	0.2967	0.3291	0.1378	0.4539	0.1429
SR-GNN	0.6197	0.2651	0.7055	0.3094	0.3669	0.1538	0.5059	0.1750
Disen-GNN	0.6236	0.2701	0.7141	0.3120	0.3981	0.1769	0.5341	0.1879
DHCN	0.6354	0.2635	0.7078	0.3029	0.3987	0.1753	0.5318	0.1844
COTREC	0.6242	0.2711	0.7113	0.3121	0.4179	0.1812	0.5411	0.1902
DGCL-GNN	**0.6509**	**0.2889**	**0.7469**	**0.3289**	**0.4305**	**0.1824**	**0.5501**	**0.1911**

5.2 Overall Performance

Table 2 shows the overall performance of DGCL-GNN compared to the baseline models. We take the average of 10 runs as the result. By comparing the experimental results, we can make the following four observations:

(1) Compared with RNN-based and Attention-based models, GNN-based models obviously perform better. It exhibits the great capability of GNN in learning more accurate embeddings and modeling session data.

(2) Models trained through self-supervised learning often exhibit superior performance and demonstrate consistent results across different datasets.

(3) DGCL-GNN outperforms all the baseline models in all datasets. Especially in Nowplaying, there is obvious performance improvement.

In summary, we can conclude that effective CL can lead to better model optimization, and the enhanced CL in our model has a positive impact on model's overall performance.

5.3 Impact of Hyperparameters

The hyperparameters with the greatest impact on model performance is $\mathcal{K}$, which is the number of disentangled hidden factors. Having more hidden factors can lead to a finer granularity of factor-level CL, but it can also make it more challenging to train accurate factor-level embeddings. Thus, determining the appropriate value for the parameter $\mathcal{K}$ is a crucial consideration. We set it to 5 on the *Diginetica* and *Yoochoose1/64*. The following experimental results also prove the advantages of this setup.

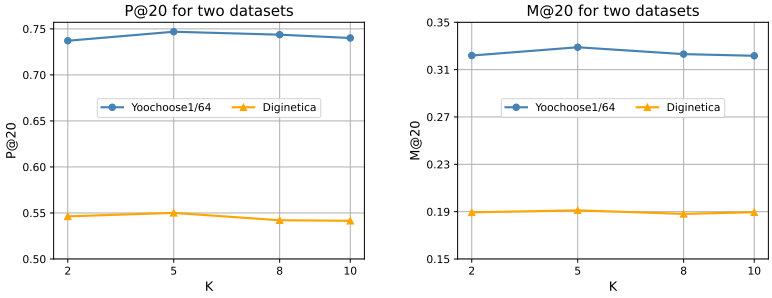


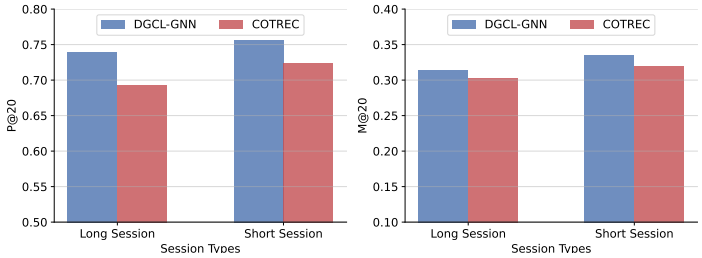
Fig. 2: Impact of the hyperparameter $\mathcal{K}$

5.4 Performance for Different Session Length

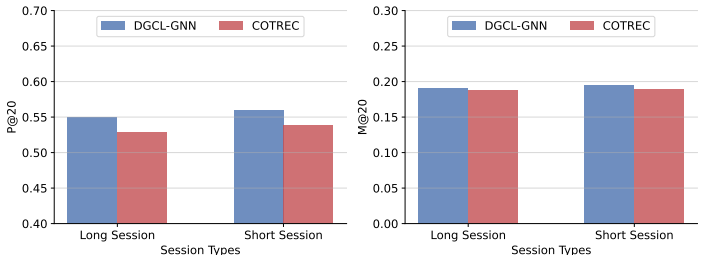
Next, as with most SBR models, we evaluated the performance of our model on both long and short sessions. We performe experiments on Yoochoose1/64 dataset and Nowplaying dataset. We first split the test sets into *long* sessions and *short* sessions. Similar to [11], sessions with a length ≥ 5 are defined to be *long* sessions while the others are *short* ones. After that, we get two sets of long and short sessions and test the performance of MGCL-GNN and a state-of-the-art baseline, *COTREC*. The experimental results show that DGCL-GNN is superior to the other two models in both long and short sessions showing that our model comprehensively improves the accuracy of recommendation.

5.5 Ablation Experiments

In order to verify the effectiveness of the main improvement in our model, the CL module, we have designed three variants of DGCL-GNN. The first one is DGCL-GNN-FCL, in which we remove the factor-level CL but maintain item-level CL.



(a) P@20 and M@20 on Yoochoose1/64 of short and long sessions



(b) P@20 and M@20 on Diginetica of short and long sessions

Fig. 3: Comparison on different lengths of sessions

The second one is DGCL-GNN-STAR, we replace the star graph augmentation with the more conventional approach of deleting edges and points as a data augmentation method. The third one is DGCL-GNN-CL, We remove all CL operations.

The rest parts of the variants keep consistent with the original DGCL-GNN to ensure the fairness of the ablation experiments.

The experimental results prove that any part of our CL module can improve the recommendation performance of DGCL-GNN.

Table 3: Comparing the performance of DGCL-GNN with its three variants.

Model	**Yoochoose 1/64**		**Diginetica**	
	P@20	**M@20**	**P@20**	**M@20**
MGCL	0.7469	0.3289	0.5501	0.1911
MGCL-FCL	0.7391	0.3201	0.5432	0.1895
MGCL-STAR	0.7415	0.3227	0.5459	0.1902
MGCL-CL	0.7288	0.3111	0.5369	0.1859

6 Conclusion and Future Work

Our DGCN-GNN model improves upon the original CL strategy in two ways. First, we implement factor-level CL to capture subtle differences between instances. Second, we utilize star graph for graph data augmentation to avoid obtaining non-ground-truth samples. Results of extensive experiments prove the effectiveness of DGCL-GNN.

Future work includes simplifying the parameters of the proposed CL strategy with the goal of designing it as a modular component that can be easily adapted to other models. Any model can call this component to enhance the embedding learning capabilities of these models and to stabilize the model performance.

References

1. Hidasi, B., Karatzoglou, A., Baltrunas, L., Tikk, D.: Session-based recommendations with recurrent neural networks. arXiv preprint arXiv:1511.06939 (2015)
2. 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)
3. Quadrana, M., Karatzoglou, A., Hidasi, B., Cremonesi, P.: Personalizing session-based recommendations with hierarchical recurrent neural networks. In: proceedings of the Eleventh ACM Conference on Recommender Systems. pp. 130–137 (2017)
4. 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)
5. 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)
6. Tan, Y.K., Xu, X., Liu, Y.: Improved recurrent neural networks for session-based recommendations. In: Proceedings of the 1st workshop on deep learning for recommender systems. pp. 17–22 (2016)
7. 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)
8. Liu, L., Wang, L., Lian, T.: Case4sr: Using category sequence graph to augment session-based recommendation. Knowledge-Based Systems **212**, 106558 (2021)
9. Qiu, R., Li, J., Huang, Z., Yin, H.: Rethinking the item order in session-based recommendation with graph neural networks. In: Proceedings of the 28th ACM international conference on information and knowledge management. pp. 579–588 (2019)
10. Wang, Z., Wei, W., Cong, G., Li, X.L., Mao, X.L., Qiu, M.: Global context enhanced graph neural networks for session-based recommendation. In: Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval. pp. 169–178 (2020)

11. 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)
12. Luo, A., Zhao, P., Liu, Y., Zhuang, F., Wang, D., Xu, J., Fang, J., Sheng, V.S.: Collaborative self-attention network for session-based recommendation. In: IJCAI. pp. 2591–2597 (2020)
13. 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)
14. Wen, Z., Li, Y.: Toward understanding the feature learning process of self-supervised contrastive learning. In: International Conference on Machine Learning. pp. 11112–11122. PMLR (2021)
15. 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)
16. Liu, X., Zhang, F., Hou, Z., Mian, L., Wang, Z., Zhang, J., Tang, J.: Self-supervised learning: Generative or contrastive. IEEE Transactions on Knowledge and Data Engineering **35**(1), 857–876 (2021)
17. Xia, X., Yin, H., Yu, J., Wang, Q., Cui, L., Zhang, X.: Self-supervised hypergraph convolutional networks for session-based recommendation. In: Proceedings of the AAAI conference on artificial intelligence. vol. 35, pp. 4503–4511 (2021)
18. Xin, X., Karatzoglou, A., Arapakis, I., Jose, J.M.: Self-supervised reinforcement learning for recommender systems. In: Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval. pp. 931–940 (2020)
19. Pan, Z., Cai, F., Chen, W., Chen, C., Chen, H.: Collaborative graph learning for session-based recommendation. ACM Transactions on Information Systems (TOIS) **40**(4), 1–26 (2022)
20. Wang, L., Xu, X., Ouyang, K., Duan, H., Lu, Y., Zheng, H.T.: Self-supervised dual-channel attentive network for session-based social recommendation. In: 2022 IEEE 38th International Conference on Data Engineering (ICDE). pp. 2034–2045. IEEE (2022)
21. Xia, X., Yin, H., Yu, J., Shao, Y., Cui, L.: Self-supervised graph co-training for session-based recommendation. In: Proceedings of the 30th ACM International conference on information & knowledge management. pp. 2180–2190 (2021)
22. Medsker, L.R., Jain, L.: Recurrent neural networks. Design and Applications **5**, 64–67 (2001)
23. Scarselli, F., Gori, M., Tsoi, A.C., Hagenbuchner, M., Monfardini, G.: The graph neural network model. IEEE transactions on neural networks **20**(1), 61–80 (2008)
24. 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)
25. Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C., Sun, M.: Graph neural networks: A review of methods and applications. AI open **1**, 57–81 (2020)
26. Li, A., Cheng, Z., Liu, F., Gao, Z., Guan, W., Peng, Y.: Disentangled graph neural networks for session-based recommendation. IEEE Transactions on Knowledge and Data Engineering (2022)

27. Chartsias, A., Joyce, T., Papanastasiou, G., Semple, S., Williams, M., Newby, D.E., Dharmakumar, R., Tsaftaris, S.A.: Disentangled representation learning in cardiac image analysis. Medical image analysis **58**, 101535 (2019)
28. John, V., Mou, L., Bahuleyan, H., Vechtomova, O.: Disentangled representation learning for non-parallel text style transfer. arXiv preprint arXiv:1808.04339 (2018)
29. Lin, Z., Tian, C., Hou, Y., Zhao, W.X.: Improving graph collaborative filtering with neighborhood-enriched contrastive learning. In: Proceedings of the ACM Web Conference 2022. pp. 2320–2329 (2022)
30. Dai, B., Lin, D.: Contrastive learning for image captioning. Advances in Neural Information Processing Systems **30** (2017)
31. Tran, L., Yin, X., Liu, X.: Disentangled representation learning gan for pose-invariant face recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1415–1424 (2017)
32. Xia, L., Huang, C., Xu, Y., Zhao, J., Yin, D., Huang, J.: Hypergraph contrastive collaborative filtering. In: Proceedings of the 45th International ACM SIGIR conference on research and development in information retrieval. pp. 70–79 (2022)
33. Giorgi, J., Nitski, O., Wang, B., Bader, G.: Declutr: Deep contrastive learning for unsupervised textual representations. arXiv preprint arXiv:2006.03659 (2020)
34. Khosla, P., Teterwak, P., Wang, C., Sarna, A., Tian, Y., Isola, P., Maschinot, A., Liu, C., Krishnan, D.: Supervised contrastive learning. Advances in neural information processing systems **33**, 18661–18673 (2020)
35. Chen, Y., Liu, Z., Li, J., McAuley, J., Xiong, C.: Intent contrastive learning for sequential recommendation. In: Proceedings of the ACM Web Conference 2022. pp. 2172–2182 (2022)
36. Qiu, R., Huang, Z., Yin, H., Wang, Z.: Contrastive learning for representation degeneration problem in sequential recommendation. In: Proceedings of the fifteenth ACM international conference on web search and data mining. pp. 813–823 (2022)
37. Székely, G.J., Rizzo, M.L., Bakirov, N.K.: Measuring and testing dependence by correlation of distances (2007)
38. Li, Y., Tarlow, D., Brockschmidt, M., Zemel, R.: Gated graph sequence neural networks. arXiv preprint arXiv:1511.05493 (2015)