# Self-Supervised Graph Co-Training for Session-based Recommendation

Xin Xia
The University of Queensland
x.xia@uq.edu.au

Hongzhi Yin*
The University of Queensland
h.yin1@uq.edu.au

Junliang Yu
The University of Queensland
jl.yu@uq.edu.au

Yingxia Shao
BUPT
shaoyx@bupt.edu.cn

Lizhen Cui
Shandong University
clz@sdu.edu.cn



## ABSTRACT

Session-based recommendation targets next-item prediction by exploiting user behaviors within a short time period. Compared with other recommendation paradigms, session-based recommendation suffers more from the problem of data sparsity due to the very limited short-term interactions. Self-supervised learning, which can discover ground-truth samples from the raw data, holds vast potentials to tackle this problem. However, existing self-supervised recommendation models mainly rely on item/segment dropout to augment data, which are not fit for session-based recommendation because the dropout leads to sparser data, creating unserviceable self-supervision signals. In this paper, for informative session-based data augmentation, we combine self-supervised learning with co-training, and then develop a framework to enhance session-based recommendation. Technically, we first exploit the session-based graph to augment two views that exhibit the internal and external connectivities of sessions, and then we build two distinct graph encoders over the two views, which recursively leverage the different connectivity information to generate ground-truth samples to supervise each other by contrastive learning. In contrast to the dropout strategy, the proposed self-supervised graph co-training preserves the complete session information and fulfills genuine data augmentation. Extensive experiments on multiple benchmark datasets show that, session-based recommendation can be remarkably enhanced under the regime of self-supervised graph co-training, achieving the state-of-the-art performance.

## CCS CONCEPTS

• **Information systems** → **Recommender systems**; • **Theory of computation** → *Semi-supervised learning*.

## KEYWORDS

Self-Supervised Learning, Contrastive Learning, Session-based Recommendation, Co-Training

*Corresponding author and having equal contribution with the first author.


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

ACM ISBN 978-1-4503-8446-9/21/11…$15.00
https://doi.org/10.1145/3459637.3482388

**ACM Reference Format:**
Xin Xia, Hongzhi Yin, Junliang Yu, Yingxia Shao, and Lizhen Cui. 2021. Self-Supervised Graph Co-Training for Session-based Recommendation. In *Proceedings of the 30th ACM International Conference on Information and Knowledge Management (CIKM '21), November 1–5, 2021, Virtual Event, QLD, Australia.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3459637.3482388

## 1 INTRODUCTION

Recommender systems (RS) now have been pervasive and become an indispensable tool to facilitate online shopping and information delivery. Most traditional recommendation approaches share a common assumption that user behaviors are constantly recorded and available for access [50, 52]. However, in some situations, recording long-term user profiles may be infeasible. For example, guests who do not log in, or users who keep personal information private do not have an accessible user profile. Session-based recommendation emerges to tackle this challenge [36], aiming at predicting the next item only with short-term user interaction data generated in a session. Owing to its promising prospect, in the past few years, session-based recommendation has received considerable attention, and a number of models have been successively developed [18, 20, 28, 29].

Early effort in this field brought Markov Chain into session-based scenarios to capture the temporal information [28, 29]. Afterwards, deep learning exhibited overwhelming advantage of modeling sequential data[56], and recurrent neural networks (RNNs) became the dominant paradigm in this line of research [12, 13]. Recently, graph neural networks (GNNs) [42] have sparked heated discussions across multiple fields for its unprecedented effectiveness in solving graph-based tasks. As session-based data can also be modeled as sequence-like graphs, there also have been a proliferation of GNNs-based session-based recommendation models [24, 27, 41, 43, 47], which outperform RNNs-based models and show decent improvements. Despite the achievements, however, these approaches are still compromised by the same issue - data sparsity. Due to the inaccessibility of the long-term user behavior data, session-based recommenders can only leverage very limited user-item interactions generated within a short session to refine the corresponding user/session representations. In most cases, these data is too few to induce an accurate user preference, leading to sub-optimal recommendation performance.

Self-supervised learning (SSL) [21], as an emerging learning paradigm which can discover ground-truth samples from the raw data, is considered to be an antidote to the data sparsity issue. Inspired

by its great success in the areas of graph and visual representation learning [11, 17], recent advances seek to harness SSL for improving recommendation [43, 44, 53, 57]. The typical idea of applying SSL to recommendation is conducting stochastic data augmentations by randomly dropping some items/segments from the raw user-item interaction graph/sequence to create supervisory signals, which is analogous to the strategy used in masked language models like BERT [7]. Following this line of thought, Bert4Rec [30] drives a cloze objective for sequential recommendation by predicting the random masked items in the sequence with their left and right contexts. $S^3$-Rec [57] designs four types of pretexts to derive supervision signals from the segments, items and attributes of sequential data and then utilizes mutual information maximization to refine item representations. Similarly, CL4SRec [44] adopts item cropping, masking and reordering to construct different data augmentations based on sequences for contrastive learning. With such random dropout strategies, SSL is compatible with sequential recommendation. However, when it comes to session-based recommendation, the same idea may not be practicable. It should be noted that, the user interaction data generated in a session is much less than a long-term user profile in sequential recommenders. Accordingly, conducting dropout on session-based data would create sparser sequences, which could be unserviceable for improving recommendation performance. To address this problem, in this paper, we propose a novel framework which combines SSL with semi-supervised learning to create more informative self-supervision signals to enhance session-based recommendation.

Co-training [3], as a classical semi-supervised learning paradigm, exploits unlabeled data to improve classifiers. The basic idea of co-training is to train two classifiers over two different data views, and then predict pseudo-labels of unlabeled instances to supervise each other in an iterative way. In our framework, we first exploit the session-item graph to construct two views (item view and session view) that exhibit the internal and external connections of sessions. Then two asymmetric graph encoders (i.e. graph convolutional networks) are built over these two views and trained under the scheme of co-training. One of them (main encoder) is for recommendation and the other acts as the auxiliary encoder to boost the former. Specifically, given a session, we regard the items as unlabeled data. In each time, one encoder predicts its possible next items and delivers them to the other encoder, respectively. By doing so, both encoders can acquire complementary information from each other. And then a contrastive objective is optimized towards refining the encoders and item representations. Meanwhile, to prevent the mode collapse (i.e. two encoders become very similar and suggest the same item), we exploit adversarial examples to encourage divergence between the two views. As this co-training regime is built upon the graph views derived from the same data source for data augmentation, and is with a contrastive objective, we name it *self-supervised graph co-training*. By iterating this process, the benefits can be two-fold: (1). with the co-training proceeding, the generated item samples become more informative (a.k.a. hard examples), which can bring more useful information to each encoder compared with the dropout strategy that is only for self-discrimination; (2). the complete data of a session is preserved and two different aspects of connectivity information are exploited, generating more practicable self-supervision signals. Finally, the main encoder is significantly improved for recommendation.

Overall, the contributions of this paper are summarized as follows:

- We propose a novel self-supervised framework for session-based recommendation which can generate more informative and practicable self-supervision signals.
- The proposed framework is model-agnostic. Ideally, the architectures of the two used encoders can be diverse, which generalizes the framework to adapt to more scenarios.
- Extensive experiments show that the proposed framework has overwhelming superiority over the state-of-the-art baselines and achieves statistically significant improvements on benchmark datasets. We release the code at https://github.com/xiaxin1998/COTREC.

The rest of this paper is organized as follows. Section 2 summarizes the related work of session-based recommendation and self-supervised learning. Section 3 presents the proposed framework. The experimental results are reported in Section 4. Finally, Section 5 concludes this paper.

## 2 RELATED WORK

### 2.1 Session-based Recommendation

Early studies on session-based recommendation focused on exploiting temporal information from session data with Markov chain [28, 29, 49, 58]. Zimdars *et al.* [58] investigated the order of temporal data based on Markov chain and used a probability decision-tree to model sequential patterns between items. Shan *et al.* [29] developed a novel recommender system based on an Markov Decision Process model with appropriate initialization and generated recommendations based upon the transition probabilities between items. With the boom of deep learning, recurrent neural networks (RNNs) [15] have been applied to session-based recommendation models to capture sequential order between items and achieved great success [19, 55]. Hidasi *et al.* [13] was the first that applied RNNs to model the whole session and introduced several modifications to vanilla RNNs such as a ranking loss function and session-parallel mini-batch training to generate more accurate recommendations. As a follow-up study [33], Tan *et al.* enhanced RNNs by utilizing the technique of data augmentation and handling the temporal shifts of session data. Besides, NARM [18], a neural attentive recommendation algorithm, employs a hybrid encoder with attention mechanism to model the user's sequential behavior and capture the user's main purpose in the current session. In [20], a short-term attention priority model is developed to capture both local and global user interests with simple multilayer perceptrons (MLPs) networks and attention mechanism.

Graph Neural Networks (GNNs) [42] are recently introduced to session-based recommendation and exhibit great performance [24, 27, 32, 37, 41]. Unlike RNN-based approaches, graph structure is an essential factor in graph-based methods, aiming to learn item transitions over session graphs. For example, SR-GNN [41] is the first to model session sequences in session graphs and applies a gated GNN model to aggregate information between items into session representations. MGNN-SPred [37] builds a multi-relational

item graph based on all session clicks to learn global item associations and uses a gated mechanism to adaptively predict the next item. GC-SAN [47] dynamically constructs session-educed graphs and employs self-attention networks on the graphs to capture item dependencies via graph information aggregation. FGNN [27] rethinks the sequence order of items to exploit users' intrinsic intents using GNNs. GCE-GNN [38] aggregates item information from both item-level and session-level through graph convolution and self-attention mechanism. LESSR [5] proposes an edge-order preserving aggregation scheme based on GRU and a shortcut graph attention layer to address the lossy session encoding problem and effectively capture long-range dependencies, respectively. Although these graph-based methods outperform RNN-based methods, they all suffer data sparsity problem due to the limited short-term profiles in session-based scenarios.

## 2.2 Self-Supervised Learning in RS

Recently, self-supervised learning (SSL) [14], as a novel machine learning paradigm which mines free labels from unlabeled data and supervises models using the generated labels, is under the spotlight. The information or intermediate representation learned from self-supervised learning are expected to carry good semantic or structural meanings and can be beneficial to a variety of downstream tasks. SSL was initially applied in the fields of visual representation learning and language modeling [2, 7], where it augments the raw data through image rotation/clipping and sentence masking. Recent advances of SSL start to focus on graphs, and have received considerable attention [16, 35, 40]. DGI [35] maximizes mutual information between the local patch and the global graph to refine node representations, making them as the ground-truth of each other. In InfoGraph [31], graph-level representations encode different aspects of data by encouraging agreement between the representations of substructures with different scales (e.g., nodes, edges, triangles). Hassani *et al.* [10] contrasted multiple views of graphs and nodes to learn their representations. Qiu [26] *et al.* designed a self-supervised graph neural network pre-training framework to capture the structural representations of graphs by leveraging instance discrimination and contrastive learning.

Inspired by the success of SSL in other areas, there are also some studies that integrate self-supervised learning into sequential recommendation [22, 45, 57]. Bert4Rec [30] transfers the cloze objective from language modeling to sequential recommendation by predicting the random masked items in the sequence with the surrounding contexts. $S^3$-Rec [57] utilizes the intrinsic data correlations among attribute, item, subsequence and sequence to generate self-supervision signals and enhance the data representations via pre-training. Xie *et al.* [44] proposed three data augmentation strategies to construct self-supervision signals from the original user behavior sequences, extracting more meaningful user patterns and encoding effective user representation. Ma *et al.* [22] proposed a sequence-to-sequence training strategy based on latent self-supervision and disentanglement of user intention behind behavior sequences. Besides, SSL is also applied to other recommendation paradigms such as general recommendation [48] and social recommendation [51, 53]. Although these self-supervised methods have achieved decent improvements in recommendation performance, they are not suitable for session-based recommendation for the reason that the random dropout strategy used in these models would lead to sparser session data and unserviceable self-supervision signals. The most relevant work to ours is $S^2$-DHCN [43] which conducts contrastive learning between representations learned over different hypergraphs by employing self-discrimination without random dropout. But it cannot learn invariant representations against the data variance for its fixed ground-truths, leading to merely slight improvements. Besides, Yu *et al.* [51] recently proposed a self-supervised tri-training framework that leverages different aspects of social information to generate complementary self-supervision signals to boost recommendation. As the first work to combine SSL with multi-view semi-supervised learning for recommendation, it gives us clues about applying co-training to session-based recommendation.

# 3 PROPOSED METHOD

## 3.1 Preliminaries

*3.1.1 Notations.* Let $H = \{i_1, i_2, i_3, ..., i_N\}$ denote the set of items, where $N$ is the number of items. Each session is represented as a sequence $s = [i_{s,1}, i_{s,2}, i_{s,3}, ..., i_{s,m}]$ ordered by timestamps and $i_{s,k} \in H$ $(1 \le k \le m)$ represents an interacted item of an anonymous user within the session $s$. For learning presentations, we embed each item $i \in I$ into the same space and let $\mathbf{x}_i^{(l)} \in \mathbb{R}^{d^{(l)}}$ denote the representation of item $i$ of dimension $d^{(l)}$ in the $l$-th layer of a deep neural network. The representation of the whole item set is denoted as $\mathbf{X}^{(l)} \in \mathbb{R}^{N \times d^{(l)}}$, and $\boldsymbol{X}^{(0)}$ is randomly initialized with uniform distribution. Each session $s$ is represented by a vector $\mathbf{s}$. The task of session-based recommendation is to predict the next item, namely $i_{s,m+1}$, for any given session $s$. Given $I$ and $s$, the output of session-based recommendation model is a ranked list $y = [y_1, y_2, y_3, ..., y_N]$ where $y_i$ $(1 \le i \le N)$ is the corresponding predicted probability of item $i$. The top-$K$ items $(1 \le K \le N)$ with highest probabilities in $y$ will be selected as the recommendations.

*3.1.2 Co-Training.* Co-Training is a classical semi-supervised learning paradigm to exploit unlabeled data [3, 6, 9]. Under this regime, two classifiers are separately trained on two views and then exchange confident pseudo labels of unlabeled instances to construct additional labeled training data for each other. Typically, the two views are two disjoint sets of features and can provide complementary information to each other. Blum *et al.* [3] first proved that co-training can bring significant benefits when the two views are sufficient and conditionally independent. However, the required conditional dependence of two views is hard to be satisfied in many cases. To relax this assumption, Abney *et al.* [1] found that weak dependence can also enable co-training success, which lifts the dependence restriction and makes co-training easily applied. Furthermore, co-training can also be applied when there is only single data representation if the data is processed by independent prediction models, such as two different classifiers [46]. It should be mentioned that there have been several attempts that combine co-training and recommendation [6, 54]. However, these methods are all based on shallow or KNN-based models, leaving much space to be explored by the graph neural models coupled with SSL.

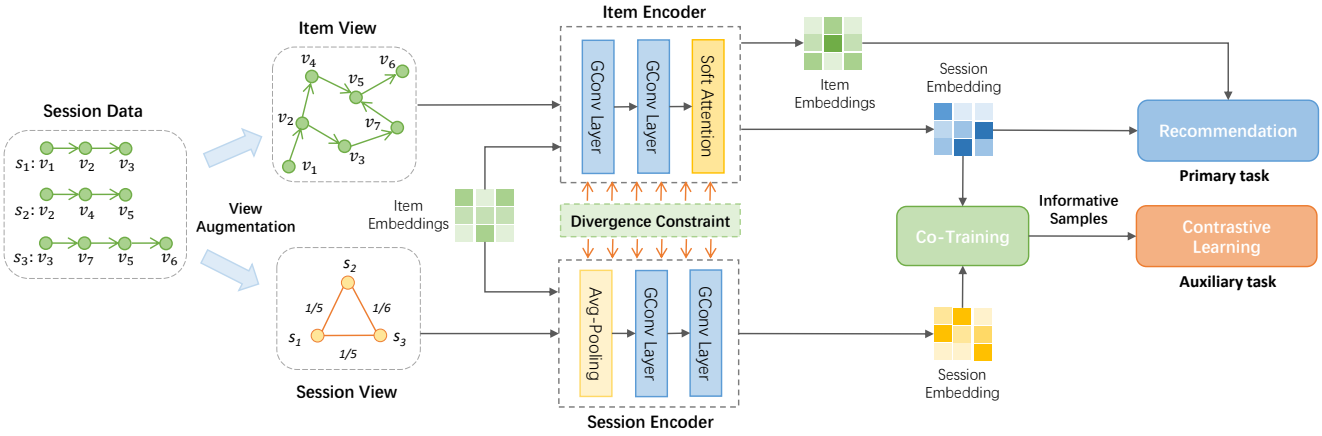


**Figure 1: An overview of the proposed COTREC framework.**

## 3.2 Self-Supervised Graph Co-Training

In this section, the proposed self-supervised graph **CO**-**T**raining framework for session-based **REC**ommendation (**COTREC**) is presented. The overview of COTREC is illustrated in Figure 1.

*3.2.1 View Augmentation.* To conduct co-training, we first derive two different views from the session data, i.e. item view and session view, by exploiting the intra- and inter-connectivity patterns of sessions. The item-view captures the item-level connectivity information while the session view encodes the session-level structural patterns. Concretely, the item view is educed by aligning all sessions. In other words, any two items ($i_a$ and $i_b$) which are connected in a session also get connected as nodes in the item view with a weighted directed edge $E_{ab}$, counting how many times they are adjacent in different sessions in the form of $[i_a, i_b]$. As for the session view, two sessions ($s_j$ and $s_k$) are connected as nodes with a weighted undirected edge $E_{jk}$ obtained by using the number of shared items to divide the number of total items in the two sessions (shown in the left part of Figure 1). These two views are able to provide complementary information for each other while keeping independent and exhibiting divergence to some degree, which are subject to the weak dependence constraint in [1]. To make an analogy, if we intuitively consider the session data presented in the left side of Fig.1 as the complete information, which is analogous to the whole picture in the task of image recognition, then constructing these two views corresponds to the patch clipping in visual self-supervised learning [4]. The augmented parts differ but inherit essential information from the original data, which can help learn more generalizable representations through a self-supervised task.

*3.2.2 Learning Graph Encoders over Augmented Views.* After the view construction, we have two types of graphs. Although we aim to devise a model-agnostic framework that can drive a multitude of session-based neural graph recommendation models, for a concrete architecture than can fulfill the capability of the proposed self-supervised graph co-training, we construct two different graph encoders with graph convolutions over the views as the base. However, the technical details can be modified to adapt to more scenarios.

**Item View Encoding.** The item encoder with a simplified graph convolution layer for the item view is defined as:

$$\mathbf{X}_I^{(l+1)} = \hat{\mathbf{D}}_I^{-1}\hat{\mathbf{A}}_I\mathbf{X}_I^{(l)}\mathbf{W}_I^l, \tag{1}$$

where $\hat{\mathbf{A}}_I = \mathbf{A}_I + \mathbf{I}$ and $\mathbf{I}$ is the identity matrix, $\hat{\mathbf{D}}_{I,p,p} = \sum_{q=1}^{m}\hat{\mathbf{A}}_{I,p,q}$ and $\mathbf{A}_I$ are the degree matrix and the adjacency matrix, $\mathbf{X}_I^{(l)}$ and $\mathbf{W}_I^l$ represent the $l$-th layer's item embeddings and parameter matrix of the item view, respectively. Here we do not use the non-linear function since it has been proved redundant in recommendation [39, 53]. After passing $\mathbf{X}^{(0)}$ through $L$ graph convolution layers, we average the item embeddings obtained from each layer to be the final learned item embeddings $\mathbf{X}_I = \frac{1}{L+1}\sum_{l=0}^{L}\mathbf{X}_I^{(l)}$. Although the graph convolution can perfectly capture item connections, it cannot encode the order of items in a specific session. Following [38], we concatenate the reversed position embeddings with the learned item representations by a learnable position matrix $\mathbf{P}_r = [\mathbf{p_1}, \mathbf{p_2}, \mathbf{p_3}, ..., \mathbf{p_m}]$, where $m$ is the length of the current session and $\mathbf{p_m} \in \mathbb{R}^d$ represents the vector of position $m$. The embedding of $t$-th item in session $s = [i_{s,1}, i_{s,2}, i_{s,3}, ..., i_{s,m}]$ is:

$$\mathbf{x}_I^{t*} = \tanh\left(\mathbf{W}_1\left[\mathbf{x}_I^t \| \mathbf{p}_{m-t+1}\right] + \mathbf{b}\right), \tag{2}$$

where $\mathbf{W}_1 \in \mathbb{R}^{d\times 2d}$, and $b \in \mathbb{R}^d$ are learnable parameters.

Session embeddings can be obtained by aggregating representations of items contained in that session. A soft-attention mechanism is often used in session-based recommendation methods where different items should have different priorities when learning session embeddings. We follow the strategy used in GCE-GNN [38] to refine the embedding of session $s = [i_{s,1}, i_{s,2}, i_{s,3}, ..., i_{s,m}]$:

$$\alpha_t = \mathbf{f}^\top\sigma\left(\mathbf{W}_2\mathbf{x}_s + \mathbf{W}_3\mathbf{x}_I^{t*} + \mathbf{c}\right), \theta_I = \sum_{t=1}^{m}\alpha_t\mathbf{x}_I^{t*} \tag{3}$$

where $\mathbf{x}_s$ is the embedding of session $s$ and here it is obtained by averaging the embeddings of items within the session $s$, i.e. $\mathbf{x}_s = \frac{1}{m}\sum_{t=1}^{m}\mathbf{x}_I^m$. Session representation $\theta_I$ is represented by aggregating item embeddings considering their corresponding importance. $\mathbf{f}, \mathbf{c} \in \mathbb{R}^d$, $\mathbf{W}_2 \in \mathbb{R}^{d\times d}$ and $\mathbf{W}_3 \in \mathbb{R}^{d\times d}$ are attention parameters used to learn the item weight $\alpha_t$.

**Session View Encoding.** The session view depicts item and session relations from the other perspective. Similarly, the session encoder conducts graph convolution on the session graph. As there are no items involved in the session graph, we first initialize the session embeddings $\mathbf{\Theta}_S^{(0)}$ by averaging the corresponding embeddings of items of each session in $\mathbf{X}^{(0)}$. And the graph convolution on session graph is defined as followed:

$$\mathbf{\Theta}_S^{(l+1)} = \hat{\mathbf{D}}_S^{-1}\hat{\mathbf{A}}_S\mathbf{\Theta}_S^{(l)}\mathbf{W}_S^{(l)}, \tag{4}$$

where $\hat{\mathbf{A}}_S = \mathbf{A}_S + \mathbf{I}$, $\mathbf{A}_S$ is the adjacency matrix, and $\hat{\mathbf{D}}_S$ is the corresponding degree matrix, $\mathbf{\Theta}_S^{(l)}$ and $\mathbf{W}_S^{(l)}$ represent the $l$-th layer's session embeddings and the parameter matrix, respectively. Similarly, we pass initialized session embeddings into $L$ graph convolution layers to learn session-level information. The final session representations are obtained by averaging $L$ embeddings learned at different layers, which is formulated as $\mathbf{\Theta}_S = \frac{1}{L+1}\sum_{l=0}^{L}\mathbf{\Theta}_S^{(l)}$.

*3.2.3 Mining Self-Supervision Signals with Graph Co-Training.* In this section, we show how graph co-training mines informative self-supervision signals to enhance session-based recommendation.

Recall that, in the last two subsections, we build two graph encoders over two different views that can provide complementary information to each other. Therefore, it is natural to refine each encoder by exploiting the information from the other view. This can be achieved by following the regime of co-training. Given a session $p$ in the session view, we predict its positive and negative next-item samples using its representation learned over the item view:

$$\mathbf{y}_I^p = \text{Softmax}\left(score_I^p\right), \quad score_I^p = \mathbf{X}_I\theta_I^p \tag{5}$$

where $\theta_I^p$ is the representation of session $p$ in the item view, and $\mathbf{y}_I^p \in \mathbb{R}^N$ denotes the predicted probability of each item being recommended to session $p$ in the item view. $\theta_I^p$ can be seen as a linear classifier, and $\mathbf{X}_I$ is seen as the unlabeled sample set.

With the computed probabilities, we can select items with the top-K highest confidence as the positive samples which act as the augmented ground-truths to supervise the session encoder. Formally, the positive sample selection is as follows:

$$c_S^{p^+} = \text{top-}K\left(\mathbf{y}_I^p\right). \tag{6}$$

As for the way to select negative samples, a straightforward idea is to take the items with the lowest scores. However, such a way can only choose easy samples which contribute little. Instead, we randomly select $K$ negative samples from the items ranked in top 10% in $\mathbf{y}_I^p$ excluding the positives to construct $c_S^{p^-}$. These items can be seen as hard negatives which can contribute enough information, and meanwhile are less likely to fall into the set of false negatives which would mislead the learning. Analogously, we use the similar way to select informative samples for the item encoder,

$$\mathbf{y}_S^p = \text{Softmax}\left(score_S^p\right), \quad score_S^p = \mathbf{X}^{(0)}\theta_S^p \tag{7}$$

where the main difference is that when selecting the top-$k$ item samples, $\boldsymbol{X}^{(0)}$ is used rather than $\boldsymbol{X}_I$ because the session encoder does not output convolved item embeddings.

In each training batch, the positive and negative pseudo-labels for each session in each view are iteratively reconstructed and then are delivered to the other view as the possible next item for refining session and item representations. The intuition behind this process is that, the item samples, which receive high confidence to be the next-item in one view, should also be valuable in the other view. Iterating this process is expected to generate more informative examples (a.k.a. harder examples). In turn, the encoders evolve under the supervision of informative examples as well, which will recursively distill harder examples.

*3.2.4 Contrastive Learning.* With the generated pseudo-labels, the self-supervised task used to refine encoders can be conducted through a contrastive objective. In session-based scenarios, we assume that the last clicked item in a session is the most related to the next item. Therefore, we can maximize (minimize) the agreement between the representations of the last-clicked item and the predicted items samples, accompanied with the given session representation as the session context. Formally, given a session $p$ and the predicted ground-truths, we follow InfoNCE [23], which can maximize the lower bound of mutual information between the item pairs, as our learning objective:

$$\begin{aligned}\mathcal{L}_{ssl} = &-\log\frac{\sum_{i\in c_I^{p^+}}\psi\left(\mathbf{x}_I^{last},\boldsymbol{\theta}_I^p,\mathbf{x}_I^i\right)}{\sum_{i\in c_I^{p^+}}\psi\left(\mathbf{x}_I^{last},\boldsymbol{\theta}_I^p,\mathbf{x}_I^i\right)+\sum_{j\in c_I^{p^-}}\psi\left(\mathbf{x}_I^{last},\boldsymbol{\theta}_I^p,\mathbf{x}_I^j\right)}\\ &-\log\frac{\sum_{i\in c_S^{p^+}}\psi\left(\mathbf{x}_{(0)}^{last},\boldsymbol{\theta}_S^p,\mathbf{x}_{(0)}^i\right)}{\sum_{c\in c_S^{p^+}}\psi\left(\mathbf{x}_{(0)}^{last},\boldsymbol{\theta}_S^p,\mathbf{x}_{(0)}^i\right)+\sum_{j\in c_S^{p^-}}\psi\left(\mathbf{x}_{(0)}^{last},\boldsymbol{\theta}_S^p,\mathbf{x}_{(0)}^j\right)}\end{aligned} \tag{8}$$

where $\mathbf{x}^{last}$ is the embedding of the last-clicked item of the given session, $\psi\left(\boldsymbol{x}_1,\boldsymbol{x}_2,\boldsymbol{x}_3\right) = \exp\left(f\left(\boldsymbol{x}_1+\boldsymbol{x}_2,\boldsymbol{x}_3+\boldsymbol{x}_2\right)/\tau\right)$ where $\tau$ is the temperature to amplify the effect of discrimination (we empirically use 0.2 in our experiments), and $f(\cdot):\mathbb{R}^d\times\mathbb{R}^d\longmapsto\mathbb{R}$ is the discriminator function that takes two vectors as the input and then scores the agreement between them. We simply implement the discriminator by applying the cosine operation. Through the contrastive learning between the positive and negative pairs, the two views can exchange information and the last-clicked item representation can learn to infer related items with a session context, and thus item and session representations are refined.

In the vanilla co-training, the generated pseudo-labels are reused in subsequent training as training labels. However, that way will make our framework less efficient because adding pseudo-labels will lead to adjacency matrix reconstruction in each iteration. Also, it may misguide the training from then on when the pseudo-labels introduce false information because the added pseudo-labels would not be removed. Therefore, in our model, we decide not to add pseudo-labels into training set in view of the above considerations. Besides, compared with the dropout based SSL methods [44, 57] which leverage the fragmentary sequences as self-supervision signals, our idea has the advantage of preserving the complete session information and fulfilling genuine label augmentation, and hence it is more suitable for session-based scenarios.

*3.2.5 Divergence Constraint in Co-Training.* In our framework, the two data views for co-training are derived from the same data

source by exploiting structural information in different aspects. On the one hand, this augmentation does not require two sufficient and independent data sources, which is the advantage. But on the other hand, it somehow might lead to the mode collapse problem, i.e., two encoders become similar and generated the same ground-truths when given the same session after a number of learning iterations. Therefore, it is necessary to make the two encoders differ to some degree. Following [25], we impose the divergence constraint on the self-supervised graph co-training regime by integrating adversarial examples into the training.

Theoretically, the adversarial examples targeting one encoder [8] would mislead it to generate wrong predictions. However, if the two encoders are trained to be resistant to the adversarial examples generated by each other and still output the correct predictions, we can manage to achieve the goal of keeping them different. We define the divergence constraint as follows:

$$\begin{aligned}\mathcal{L}_{\text{diff}} =& KL\left(Prob_I(\mathbf{X}_I), Prob_S\left(\mathbf{X}_I + \Delta^I_{adv}\right)\right) \\ &+ KL\left(Prob_S(\mathbf{X}_I), Prob_I\left(\mathbf{X}_I + \Delta^S_{adv}\right)\right),\end{aligned} \tag{9}$$

where $Prob_I(\cdot)$ and $Prob_S(\cdot)$ represent the probabilities of each item to be recommended to a given session $p$, which are computed by two encoders: $Prob_I(\mathbf{X}_I) = \text{Softmax}(\boldsymbol{X}_I\boldsymbol{\theta}^p_I)$, and $Prob_S(\mathbf{X}_I) = \text{Softmax}(\boldsymbol{X}_I\boldsymbol{\theta}^p_S)$, $\Delta^I_{adv}$ and $\Delta^S_{adv}$ represent the adversarial perturbations on the item embeddings with regard to $\boldsymbol{\theta}^p_I$ and $\boldsymbol{\theta}^p_S$, respectively, and $KL(\cdot)$ denotes the KL divergence. To make it clear, $Prob_S\left(\mathbf{X}_I + \Delta^I_{adv}\right)$ is the probability distribution produced by the session encoder when $\boldsymbol{X}_I$ is perturbed by $\Delta^I_{adv}$. If the session encoder is immune to $\Delta^I_{adv}$ that is destructive to the item encoder, it will output a probability distribution similar to $Prob_I(\mathbf{X}_I)$ due to shared information, resulting in a smaller loss of Eq. (9), otherwise not.

To create adversarial examples, we adopt the FGSM method proposed in [8], which adds adversarial perturbations on model parameters through fast gradient computation. In our paper, we add adversarial perturbations on item embeddings. The perturbations $\Delta$ are updated as:

$$\Delta_{adv} = \epsilon\frac{\Gamma}{\|\Gamma\|} \quad \text{where} \quad \Gamma = \frac{\partial l_{adv}(\hat{y} \mid \mathbf{x} + \Delta)}{\partial \Delta}. \tag{10}$$

$l_{adv}(\hat{y})$ is the loss of aversarial examples and $\epsilon$ is the control parameter ($\epsilon$ is 0.5 on Diginetica and 0.2 on Tmall and RetailRocket in our experiments).

*3.2.6 Model Optimization.* Based on the learned representations, the score of each candidate item $i \in I$ to be recommended for a session $s$ is computed by doing inner product:

$$\hat{\mathbf{z}}_i = \theta^{s\top}_I \mathbf{x}_i. \tag{11}$$

Since the item view can reflect the item connectivity in a finer-grained granularity, we use the encoder over the item view as the main encoder to predict the final candidate items for recommendation. After that, a softmax function is applied:

$$\hat{\mathbf{y}} = \text{softmax}(\hat{\mathbf{z}}). \tag{12}$$

**Algorithm 1:** The whole procedure of COTREC

**Input:** Sessions S, node embeddings $\boldsymbol{X}$ ;
**Output:** Recommendation lists

```
Construct item view and session view ;
for each iteration do
    for each batch do
        Learn item and session representations through Eq. (1) - (4) ;
        for each session s do
            Predict the probabilities of items being the positive examples in different views and obtain positive and negative examples with Eq.(5) - Eq. (7);
            Compute self-supervised learning loss of two views via Eq.(8);
        end
        Add divergence constraint by following Eq. (9) - (10);
        Jointly optimize the overall objective in Eq. (14);
    end
end
```

We then use cross entropy loss function to be the learning objective:

$$\mathcal{L}_r = -\sum_{i=1}^{N} \mathbf{y}_i \log\left(\hat{\mathbf{y}}_i\right) + \left(1 - \mathbf{y}_i\right) \log\left(1 - \hat{\mathbf{y}}_i\right). \tag{13}$$

$\mathbf{y}$ is the one-hot encoding vector of the ground truth. For simplicity, we leave out the $L_2$ regularization terms. Finally, we unify the recommendation task with the auxiliary SSL task. The total loss $L$ is defined as:

$$\mathcal{L} = \mathcal{L}_r + \beta\mathcal{L}_{ssl} + \alpha\mathcal{L}_{\text{diff}}, \tag{14}$$

where $\alpha, \beta$ are hyperparameters to control the scale of the self-supervised graph co-training and view difference constraint. It should be noted that, we jointly optimize the three throughout the training. Finally, the whole procedure of COTREC is summarized in Algorithm 1.

| Dataset | Tmall | RetailRocket | Diginetica |
|---|---|---|---|
| training sessions | 351,268 | 433,643 | 719,470 |
| test sessions | 25,898 | 15,132 | 60,858 |
| # of items | 40,728 | 36,968 | 43,097 |
| average lengths | 6.69 | 5.43 | 5.12 |

**Table 1: Dataset Statistics**

# 4 EXPERIMENTS

## 4.1 Experimental Settings

*4.1.1 Datasets.* We evaluate our model on three real-world benchmark datasets: *Tmall*[1], *RetailRocket*[2] and *Diginetica*[3], which are often used in session-based recommendation methods. *Tmall* dataset

[1] https://tianchi.aliyun.com/dataset/dataDetail?dataId=42
[2] https://www.kaggle.com/retailrocket/ecommerce-dataset
[3] http://cikm2016.cs.iupui.edu/cikm-cup/

| Method | Tmall | | | | RetailRocket | | | | Diginetica | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | P@10 | M@10 | P@20 | M@20 | P@10 | M@10 | P@20 | M@20 | P@10 | M@10 | P@20 | M@20 |
| FPMC | 13.10 | 7.12 | 16.06 | 7.32 | 25.99 | 13.38 | 32.37 | 13.82 | 15.43 | 6.20 | 26.53 | 6.95 |
| GRU4REC | 9.47 | 5.78 | 10.93 | 5.89 | 38.35 | 23.27 | 44.01 | 23.67 | 17.93 | 7.33 | 29.45 | 8.33 |
| NARM | 19.17 | 10.42 | 23.30 | 10.70 | 42.07 | 24.88 | 50.22 | 24.59 | 35.44 | 15.13 | 49.70 | 16.17 |
| STAMP | 22.63 | 13.12 | 26.47 | 13.36 | 42.95 | 24.61 | 50.96 | 25.17 | 33.98 | 14.26 | 45.64 | 14.32 |
| SR-GNN | 23.41 | 13.45 | 27.57 | 13.72 | 43.21 | 26.07 | 50.32 | 26.57 | 36.86 | 15.52 | 50.73 | 17.59 |
| GCE-GNN | 28.01 | 15.08 | 33.42 | 15.42 | - | - | - | - | 41.16 | 18.15 | **54.22** | 19.04 |
| $S^2$-DHCN | 26.22 | 14.60 | 31.42 | 15.05 | 46.15 | 26.85 | 53.66 | 27.30 | 39.87 | 17.53 | 53.18 | 18.44 |
| COTREC | **30.62** | **17.65** | **36.35** | **18.04** | **48.61** | **29.46** | **56.17** | **29.97** | **41.88** | **18.16** | 54.18 | **19.07** |

**Table 2: Performances of all comparison methods on three datasets.**

comes from IJCAI-15 competition, which contains anonymized user's shopping logs on Tmall online shopping platform. *Retail-Rocket* is a dataset on a Kaggle contest published by an e-commerce company, which contains the user's browsing activity within six months. *Diginetica* dataset describes the music listening behavior of users, and *Diginetica* comes from CIKM Cup 2016. For convenience of comparing, we follow the experiment environment in [38, 41]. Specifically, we filter out all sessions whose length is 1 and items appearing less than 5 times. Latest data (such as, the data of last week) is set to be test set and previous data is used as training set. Then, we augment and label the training and test datasets by using a sequence splitting method, which generates multiple labeled sequences with the corresponding labels $([i_{s,1}], i_{s,2}), ([i_{s,1}, i_{s,2}], i_{s,3}), ..., ([i_{s,1}, i_{s,2}, ..., i_{s,m-1}], i_{s,m})$ for every session $s = [i_{s,1}, i_{s,2}, i_{s,3}, ..., i_{s,m}]$. Note that the label of each sequence is the last click item in it. The statistics of the datasets are presented in Table 1.

*4.1.2 Baseline Methods.* We compare COTREC with the following representative methods:

- **FPMC** [28] is a sequential method based on Markov Chain. In order to adapt it to session-based recommendation, we do not consider the user latent representations when computing recommendation scores.
- **GRU4REC** [13] utilizes a session-parallel mini-batch training process and adopts ranking-based loss functions to model user sequences.
- **NARM** [18]: is a RNN-based state-of-the-art model which employs attention mechanism to capture user's main purpose and combines it with the sequential behavior to generate the recommendations.
- **STAMP** [20]: adopts attention layers to replace all RNN encoders in the previous work and employs the self-attention mechanism[34] to enhance the session-based recommendation performance.
- **SR-GNN** [41]: applies a gated graph convolutional layer to obtain item embeddings and also employs a soft-attention mechanism to compute the session embeddings.
- **GCE-GNN** [38]: constructs two types of session-educed graphs to capture local and global information in different levels.
- **$S^2$-DHCN** [43]: constructs two types of hypergraphs to learn inter- and intra-session information and uses self-supervised learning to enhance session-based recommendation.

*4.1.3 Evaluation Metrics.* Following [38, 41], we use P@K (Precision) and MRR@K (Mean Reciprocal Rank) to evaluate the recommendation results where K is 10 or 20.

*4.1.4 Hyper-parameters Settings.* Following previous works, we set the embedding size to 100, the batch size for mini-batch to 100, and the $L_2$ regularization to $10^{-5}$. In our model, all parameters are initialized with the Gaussian Distribution $\mathcal{N}(\mathbf{0}, \mathbf{0.1})$. We use Adam with the learning rate of 0.001 to optimize our model. For the number of layers of graph convolution on the three datasets, a three-layer setting achieves the best performance. For the baseline models, we refer to their best parameter setups reported in the original papers and directly report their results if available, since we use the same datasets and evaluation settings.

## 4.2 Experimental Results

*4.2.1 Overall Performance.* The experimental results of overall performance are reported in Table 2, where we highlight the best results of each column in boldface. From the results, we can draw some conclusions:

- Recent methods that consider temporal information (such as, GRU4REC, NARM, STAMP, SR-GNN) outperform traditional methods (FPMC) that do not, demonstrating the importance of capturing sequential dependency between items in session-based recommendation. Besides, among the methods based on RNNs-like units (RNN, LSTM, GRU), NRAM and STAMP achieve better performance than GRU4REC. This is because NRAM and STAMP not only utilize recurrent neural networks to model sequential behavior, but also utilize an attention mechanism to learn importance of each item when learning session representations. GRU4REC which only uses GRU cannot handle the shift of user preference.
- Graph-based baseline methods all outperform RNN-based methods, showing the great capacity of graph neural networks in modeling session data. Among them, GCE-GNN obtains higher accuracy than SR-GNN. This proves that capturing different levels of information (inter- and intra-session information) helps accurately predict user intent in session-based recommendation. $S^2$-DHCN also utilize both inter- and intra-session information in hypergraph modeling and achieves promising performance. However, compared to GCE-GNN, $S^2$-DHCN has lower results on Tmall and Diginetica, showing that self-discrimination based

SSL method is not so successful in improving session-based recommendation performance, which is in line with our motivation.

- Our proposed COTREC almost outperforms all the baselines on all the datasets. Particularly, it beats other models by a large margin on Tmall, showing the effectiveness of the self-supervised grpah co-training when applied to real e-commerce data. Compared with the other self-supervised model $S^2$-DHCN, the advantage is also obvious. Considering that $S^2$-DHCN and COTREC both have a two-branch architecture, we think that the improvements mainly derive from the multi-instance contrastive learning in Eq. (8) while $S^2$-DHCN only conducts self-discrimination contrastive learning. Compared with another strong baseline GCE-GNN, COTREC is competitive in terms of both performance and efficiency. Although GCE-GNN can achieve comparable results on Diginetica, its more complex structure makes it suffer from the out-of-memory problem when performing on RetailRocket on our RTX 2080 Ti GPU. Besides, its performance is much lower than that of COTREC on Tmall.

*4.2.2 Ablation Study.* In this section, to investigate the contribution of each component in our model, we develop four variant versions of COTREC: **COTREC-base**, **base-NP**, **base-NA**, **COTREC-ND**, and we compare the four variants with the complete COTREC model on *Tmall* and *Diginetica*. In **COTREC-base**, we only use the item view to model session data, removing the session view and the self-supervised graph co-training. In **base-NP**, we remove the reversed position embeddings. **base-NA** means that we remove the soft-attention mechanism and replace it with averaging item representations as the representation of each session. In **COTREC-ND**, we only use self-supervised co-training without the divergence constraint. We show the results of these four variants in Figure 2.

From Figure 2, we can observe that each component consistently contributes on both datasets. The self-supervised co-training improves the base model the most, serving as the driving force of the performance improvement. When removing the self-supervised co-training, we can observe a remarkable performance drop on both the two metrics. Besides, the divergence constraint is effective to prevent mode collapse in co-training process. Without this module, the performance of COTREC is even worse than that of the base on Diginetica. Note that on Tmall, base-NP outperforms COTREC-base, proving that a strict temporal order of items may negatively influence the performance in some cases, which is in line with the our previous observation in $S^2$-DHCN. According to the results of base-NA, it is shown that learning different item importance across sessions is better than directly averaging representations of contained items for learning session representations in session-based recommendation.

| Method | Tmall | | Diginetica | |
|---|---|---|---|---|
| | P@20 | M@20 | P@20 | M@20 |
| COTREC-base | 27.71 | 13.36 | 52.66 | 18.19 |
| base-DHCN | 32.94 | 16.22 | 52.02 | 17.70 |
| base-MASK | 32.54 | 16.37 | 51.61 | 17.64 |
| COTREC | 36.35 | 18.04 | 54.18 | 19.07 |

**Table 3: Comparisons of Different SSL Methods.**

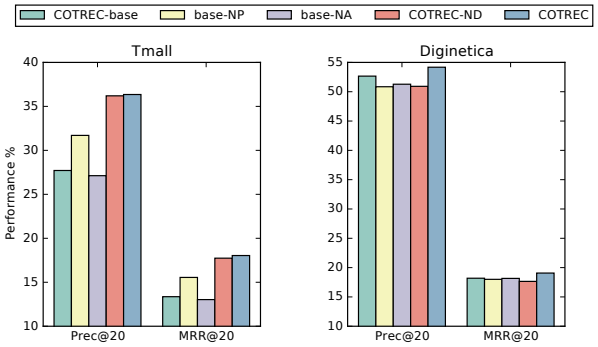


**Figure 2: Ablation Study.**

*4.2.3 Comparison with Different SSL Methods.* To further investigate the effectiveness of the proposed self-supervised graph co-training, we also compare it with other different SSL methods that are based on self-discrimination and random dropout to generate self-supervision signals on Tmall and Diginetica. The first is the method proposed in $S^2$-DHCN. DHCN proposes to capture item-level and session-level information and maximize mutual information between the session representations learned at the two levels. Positive examples are two types of session representation of the same session, whereas negative pairs are representations of different sessions. The second compared SSL method is based on the item mask, which is an often used strategy where some items are randomly dropped in each session. The generated new session can be the positive example and other sessions can be negative samples. For a fair comparison, we employ these SSL strategies on the base model of COTREC. So we name the three as **base-COTREC**, **base-DHCN**, **base-MASK**. Besides, these SSL methods are used to establish auxiliary tasks in the model optimization and we use a hyperparameter to control the magnitude of SSL. Finally, we use grid-search to adjust the parameter to ensure the best performances of them, and the best results are shown in Table 3.

From Table 3, we can see that, only the self-supervised graph co-training can boost the recommendation performance on both datasets and it also achieves the best performance, while the other two methods can only take effect on Tmall, demonstrating that self-supervised graph co-training is more effective than self-discrimination and dropout-based methods. We also find that the strategy of item mask is the least effective in most cases, proving that masked subsequences can only generate sub-optimal self-supervision signals in the scenario of session-based recommendation due to the very limited behaviors.

*4.2.4 Handling Different Session Lengths.* In real world situations, sessions with various lengths are common, so it is interesting to know how stable our COTREC as well as the baseline models are when dealing with them, and it is also a critical indicator for production environments. To evaluate this, we follow [20, 41] to split the sessions of *Tmall* and *Diginetica* into two groups with different lengths and name them as **Short** and **Long**. Short contains sessions whose lengths are less than or equal to 5, while Long contains sessions whose lengths are larger than 5. Then, we compare the performance of COTREC with some representative baselines, i.e., STAMP, SR-GNN, GCE-GNN and $S^2$-DHCN in terms of Prec@20

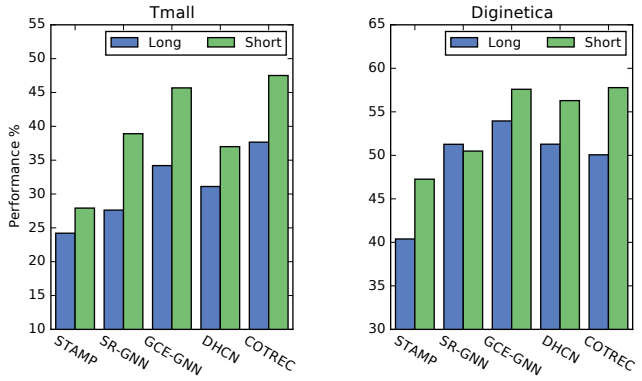


**Figure 3: P@20 results on Long and Short.**

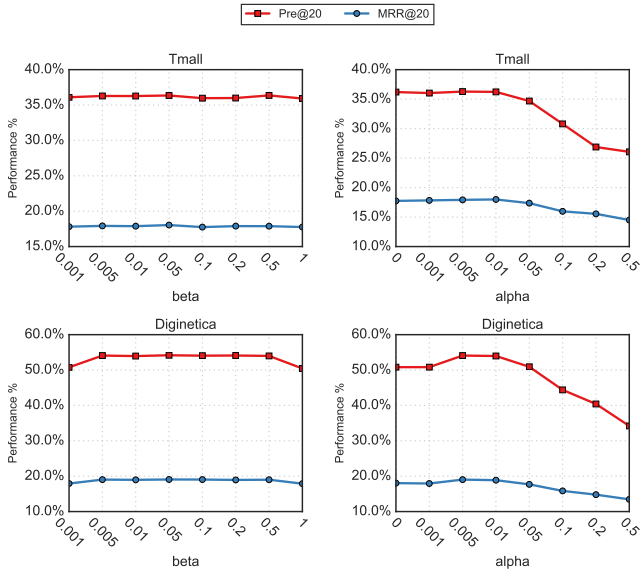


**Figure 4: Hyperparameter Analysis.**

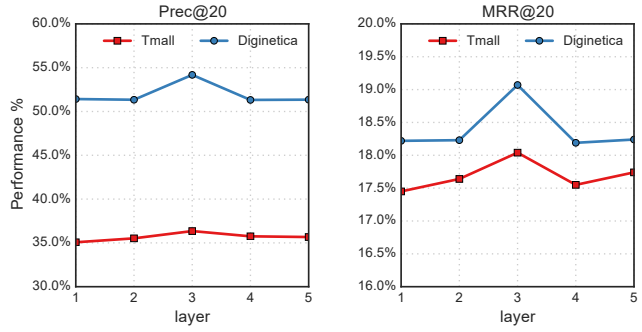


**Figure 5: The impacts of the number of layer.**

on Short and Long. Results in Figure 3 show that COTREC almost outperforms all the baseline models on both datasets with different session lengths. It demonstrates the adaptability of COTREC in real-world session-based recommendation. Besides, it is shown that the performance on the short sessions is better than that on the long sessions.

*4.2.5 The Impact of Hyperparameters.* In COTREC, we have two hyperparameters to control the magnitude of the SSL taks and the effect of the divergence constraint, i.e. $\beta$ and $\alpha$. To investigate the influence of them, we report the performance with a set of representative $\beta$ values in { 0.001, 0.005, 0.01, 0.05, 0.1, 0.2, 0.5, 1} and $\alpha$ values in {0, 0.001, 0.005, 0.01, 0.05, 0.1, 0.2, 0.5} on Tmall and Diginetica. We fix the other parameter as 0.005 when investigating $\beta$ or $\alpha$. According to the results in Figure 4, our model achieves the best performance when jointly trained with the SSL taks and the divergence constraint. On both Tmall and Diginetica, the best setting is $\beta = 0.05$ and $\alpha = 0.005$. When using large $\alpha$, a huge performance drop is observed, demonstrating that when there is large divergence between two encoder, it is hard for them to supervise each other.

*4.2.6 The impact of the number of layers.* To investigate the impact of the number of layers in graph convolution network, we range the number of layers in {1, 2, 3, 4, 5}. According to the results in Figure 5, we can see that for both *Tmall* and *Diginetica*, a three-layer setting achieves the best performance. When the number becomes larger, performance will drop due to the over-smoothing issue. Besides, an obvious performance fluctuation is observed on *Diginetica*.

## 5 CONCLUSION

Self-supervised learning is an emerging machine learning paradigm which exploits unlabeled data by generating ground-truth labels from the raw data itself and recently has been utilized in many fields to enhance deep learning models. Existing SSL-based recommendation methods usually adopt random dropout-based self-discrimination to generate self-supervision signals. However, we argue that it cannot adapt to session-based recommendation because it would create sparser data and cannot leverage informative self-supervision signals from other entities. In this paper, we design a self-supervised graph co-training framework to address this issue. In our framework, co-training can iteratively selects evolving pseudo-labels as informative self-supervision examples for each view to improve the session-based recommendation. Extensive experiments and empirical studies demonstrate the effectiveness of our framework and show its superiority over other recent baselines.

## 6 ACKNOWLEDGMENT

This work was supported by ARC Discovery Project (Grant No. DP190101985),ARC Future Fellowship (FT210100624), National Natural Science Foundation of China (No. U1936104), CCF- Baidu Open Fund, and The Fundamental Research Funds for the Central Universities 2020RC25.

## REFERENCES

[1] Steven Abney. 2002. Bootstrapping. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*. 360–367.
[2] Philip Bachman, R Devon Hjelm, and William Buchwalter. 2019. Learning representations by maximizing mutual information across views. In *Advances in Neural Information Processing Systems*. 15535–15545.

[3] Avrim Blum and Tom Mitchell. 1998. Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory*. 92–100.

[4] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*. PMLR, 1597–1607.

[5] Tianwen Chen and Raymond Chi-Wing Wong. 2020. 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*. 1172–1180.

[6] Arthur F Da Costa, Marcelo G Manzato, and Ricardo JGB Campello. 2018. CoRec: a co-training approach for recommender systems. In *Proceedings of the 33rd Annual ACM Symposium on Applied Computing*. 696–703.

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[8] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572* (2014).

[9] Tengda Han, Weidi Xie, and Andrew Zisserman. 2020. Self-supervised co-training for video representation learning. *arXiv preprint arXiv:2010.09709* (2020).

[10] Kaveh Hassani and Amir Hosein Khasahmadi. 2020. Contrastive Multi-View Representation Learning on Graphs. *arXiv preprint arXiv:2006.05582* (2020).

[11] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 9729–9738.

[12] Balázs Hidasi and Alexandros Karatzoglou. 2018. Recurrent neural networks with top-k gains for session-based recommendations. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. 843–852.

[13] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2015. Session-based recommendations with recurrent neural networks. *arXiv preprint arXiv:1511.06939* (2015).

[14] R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. 2018. Learning deep representations by mutual information estimation and maximization. *arXiv preprint arXiv:1808.06670* (2018).

[15] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[16] Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay Pande, and Jure Leskovec. 2019. Strategies for pre-training graph neural networks. *arXiv preprint arXiv:1905.12265* (2019).

[17] Wei Jin, Tyler Derr, Haochen Liu, Yiqi Wang, Suhang Wang, Zitao Liu, and Jiliang Tang. 2020. Self-supervised learning on graphs: Deep insights and new direction. *arXiv preprint arXiv:2006.10141* (2020).

[18] Jing Li, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tao Lian, and Jun Ma. 2017. Neural attentive session-based recommendation. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. 1419–1428.

[19] Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. 2016. Predicting the next location: A recurrent model with spatial and temporal contexts. In *Thirtieth AAAI conference on artificial intelligence*.

[20] Qiao Liu, Yifu Zeng, Refuoe Mokhosi, and Haibin Zhang. 2018. 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*. 1831–1839.

[21] Xiao Liu, Fanjin Zhang, Zhenyu Hou, Zhaoyu Wang, Li Mian, Jing Zhang, and Jie Tang. 2020. Self-supervised learning: Generative or contrastive. *arXiv preprint arXiv:2006.08218* 1, 2 (2020).

[22] Jianxin Ma, Chang Zhou, Hongxia Yang, Peng Cui, Xin Wang, and Wenwu Zhu. 2020. Disentangled Self-Supervision in Sequential Recommenders. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 483–491.

[23] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748* (2018).

[24] Zhiqiang Pan, Fei Cai, Wanyu Chen, Honghui Chen, and Maarten de Rijke. 2020. Star Graph Neural Networks for Session-based Recommendation. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 1195–1204.

[25] Siyuan Qiao, Wei Shen, Zhishuai Zhang, Bo Wang, and Alan Yuille. 2018. Deep co-training for semi-supervised image recognition. In *Proceedings of the european conference on computer vision (eccv)*. 135–152.

[26] Jiezhong Qiu, Qibin Chen, Yuxiao Dong, Jing Zhang, Hongxia Yang, Ming Ding, Kuansan Wang, and Jie Tang. 2020. GCC: Graph Contrastive Coding for Graph Neural Network Pre-Training. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1150–1160.

[27] Ruihong Qiu, Jingjing Li, Zi Huang, and Hongzhi Yin. 2019. 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*. 579–588.

[28] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing personalized markov chains for next-basket recommendation. In *Proceedings of the 19th international conference on World wide web*. 811–820.

[29] Guy Shani, David Heckerman, and Ronen I Brafman. 2005. An MDP-based recommender system. *Journal of Machine Learning Research* 6, Sep (2005), 1265–1295.

[30] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019. BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer. In *Proceedings of the 28th ACM international conference on information and knowledge management*. 1441–1450.

[31] Fan-Yun Sun, Jordan Hoffmann, Vikas Verma, and Jian Tang. 2019. Infograph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization. *arXiv preprint arXiv:1908.01000* (2019).

[32] Ke Sun, Tieyun Qian, Tong Chen, Yile Liang, Quoc Viet Hung Nguyen, and Hongzhi Yin. 2020. Where to go next: Modeling long-and short-term user preferences for point-of-interest recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 214–221.

[33] Yong Kiam Tan, Xinxing Xu, and Yong Liu. 2016. Improved recurrent neural networks for session-based recommendations. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*. 17–22.

[34] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is All You Need. In *Advances in neural information processing systems*. 5998–6008.

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

[36] Shoujin Wang, Longbing Cao, and Yan Wang. 2019. A survey on session-based recommender systems. *arXiv preprint arXiv:1902.04864* (2019).

[37] Wen Wang, Wei Zhang, Shukai Liu, Qi Liu, Bo Zhang, Leyu Lin, and Hongyuan Zha. 2020. Beyond clicks: Modeling multi-relational item graph for session-based target behavior prediction. In *Proceedings of The Web Conference 2020*. 3056–3062.

[38] Ziyang Wang, Wei Wei, Gao Cong, Xiao-Li Li, Xian-Ling Mao, and Minghui Qiu. 2020. 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*. 169–178.

[39] Felix Wu, Tianyi Zhang, Amauri Holanda de Souza Jr, Christopher Fifty, Tao Yu, and Kilian Q Weinberger. 2019. Simplifying graph convolutional networks. *arXiv preprint arXiv:1902.07153* (2019).

[40] Lirong Wu, Haitao Lin, Zhangyang Gao, Cheng Tan, Stan Li, et al. 2021. Self-supervised on Graphs: Contrastive, Generative, or Predictive. *arXiv preprint arXiv:2105.07342* (2021).

[41] Shu Wu, Yuyuan Tang, Yanqiao Zhu, Liang Wang, Xing Xie, and Tieniu Tan. 2019. Session-based recommendation with graph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 346–353.

[42] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. 2020. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems* (2020).

[43] Xin Xia, Hongzhi Yin, Junliang Yu, Qinyong Wang, Lizhen Cui, and Xiangliang Zhang. 2020. Self-Supervised Hypergraph Convolutional Networks for Session-based Recommendation. *arXiv preprint arXiv:2012.06852* (2020).

[44] Xu Xie, Fei Sun, Zhaoyang Liu, Jinyang Gao, Bolin Ding, and Bin Cui. 2020. Contrastive Pre-training for Sequential Recommendation. *arXiv preprint arXiv:2010.14395* (2020).

[45] Xin Xin, Alexandros Karatzoglou, Ioannis Arapakis, and Joemon M Jose. 2020. Self-Supervised Reinforcement Learning forRecommender Systems. *arXiv preprint arXiv:2006.05779* (2020).

[46] Chang Xu, Dacheng Tao, and Chao Xu. 2013. A survey on multi-view learning. *arXiv preprint arXiv:1304.5634* (2013).

[47] Chengfeng Xu, Pengpeng Zhao, Yanchi Liu, Victor S Sheng, Jiajie Xu, Fuzhen Zhuang, Junhua Fang, and Xiaofang Zhou. 2019. Graph contextualized self-attention network for session-based recommendation. In *Proc. 28th Int. Joint Conf. Artif. Intell.(IJCAI)*. 3940–3946.

[48] Tiansheng Yao, Xinyang Yi, Derek Zhiyuan Cheng, Felix Yu, Ting Chen, Aditya Menon, Lichan Hong, Ed H Chi, Steve Tjoa, Jieqi Kang, et al. 2020. Self-supervised learning for deep models in recommendations. *arXiv e-prints* (2020), arXiv–2007.

[49] Hongzhi Yin and Bin Cui. 2016. *Spatio-temporal recommendation in social media*. Springer.

[50] Junliang Yu, Min Gao, Jundong Li, Hongzhi Yin, and Huan Liu. 2018. Adaptive implicit friends identification over heterogeneous network for social recommendation. In *Proceedings of the 27th ACM international conference on information and knowledge management*. 357–366.

[51] Junliang Yu, Hongzhi Yin, Min Gao, Xin Xia, Xiangliang Zhang, and Nguyen Quoc Viet Hung. 2021. Socially-Aware Self-Supervised Tri-Training for Recommendation. *arXiv preprint arXiv:2106.03569* (2021).

[52] Junliang Yu, Hongzhi Yin, Jundong Li, Min Gao, Zi Huang, and Lizhen Cui. 2020. Enhance Social Recommendation with Adversarial Graph Convolutional Networks. *arXiv preprint arXiv:2004.02340* (2020).

[53] Junliang Yu, Hongzhi Yin, Jundong Li, Qinyong Wang, Nguyen Quoc Viet Hung, and Xiangliang Zhang. 2021. Self-Supervised Multi-Channel Hypergraph Convolutional Network for Social Recommendation. In *Proceedings of the Web Conference 2021*. 413–424.

[54] Mi Zhang, Jie Tang, Xuchen Zhang, and Xiangyang Xue. 2014. Addressing cold start in recommender systems: A semi-supervised co-training algorithm. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*. 73–82.

[55] Yuyu Zhang, Hanjun Dai, Chang Xu, Jun Feng, Taifeng Wang, Jiang Bian, Bin Wang, and Tie-Yan Liu. 2014. Sequential click prediction for sponsored search with recurrent neural networks. In *Twenty-Eighth AAAI Conference on Artificial Intelligence*.

[56] Yan Zhang, Hongzhi Yin, Zi Huang, Xingzhong Du, Guowu Yang, and Defu Lian. 2018. Discrete deep learning for fast content-aware recommendation. In *Proceedings of the eleventh ACM international conference on web search and data mining*. 717–726.

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

[58] Andrew Zimdars, David Maxwell Chickering, and Christopher Meek. 2013. Using temporal data for making recommendations. *arXiv preprint arXiv:1301.2320* (2013).