Self-Supervised Dual Graph and Intention Association for Session-based Recommendation

Junnan Zhuo[1], Bohan Li[1,2,3]✉, Sujie Yu[1], Shuai Xu[1], Xinzhe Zhao[1], Zekun Xu[1], and Guan Yuan[4]

[1] College of Artiffcial Intelligence & Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, China
[2] Ministry of Industry and Information Technology, Collaborative Innovation Center of Novel Software Technology and Industrialization, Nanjing, China
[3] National Engineering Laboratory for Integrated Aero-Space-Ground Ocean Big Data Application Technology, Xi'an, China
[4] School of Computer Science and Technology, China University of Mining and Technology, Xuzhou, China
{bhli,junnanzhuo}@nuaa.edu.cn

Abstract. Session-based recommendation systems aim to predict user clicks using anonymous session data. Current models struggle with understanding complex item transition patterns and are affected by noise in the data, thus reducing accuracy. To address these problems, we introduce a self-supervised dual graph and intention association technique for session-based recommendations, named SDGIA (**S**elf-supervised **D**ual **G**raph and **I**ntention **A**ssociation). SDGIA constructs a global undirected graph and session-directed graphs, enhancing information representation to capture transition patterns within and across sessions. A self-supervised learning mechanism improves feature extraction and generalization, meanwhile an intention association module filters out noise for more precise item representations. Experiments on three datasets demonstrate that SDGIA significantly outperforms existing models.

Keywords: Session-based recommendation · Graph neural networks · Self-supervised learning

1 Introduction

In the rapidly evolving internet era, the sheer volume of information has led to a significant and growing issue of information overload, making it increasingly difficult for users to find relevant content. Recommendation systems have emerged as a crucial tool to alleviate this burden. However, traditional recommendation techniques depend heavily on either users actively sharing personal information or on predictions based on users' historical interaction patterns. With the rise of privacy technologies, if users don't log in, it is impossible to access their historical behavior, rendering traditional recommendation techniques inapplicable [1,4,31]. Consequently, session-based recommendation systems [26] (SBR) have arisen, intending to forecast the next item a user will click leveraging anonymous session sequences grounded in temporal relationships [12].

Most SBRs can be broadly categorized into three types: traditional SBR, latent representation SBR, and deep learning-based SBR [17,28,30]. Traditional models generally rely on item co-occurrence within session sequences, such as Markov chain-based methods [35], which lack sequential information. Latent representation SBR typically require users' personal information. Deep learning-based recommendation systems can achieve good recommendation performance, such as RNN like GRU4REC [5] and NARM [10], which can address the issue of sequential transitions between consecutive items to some extent. However, they struggle to capture complex transitions between items. While attention mechanisms help mitigate long-term dependency challenges by enabling models to focus on relevant items in session graphs, current methods, such as GCE-GNN [21], often fall short in fully utilizing the sequential dependency within sessions.

Deep learning-based SBRs face several problems: (1) They overlook the complex transition relationships between items, and research shows that constructing sequences as session graphs can better retain pairwise transition relationships. (2) They lack the utilization of inter-session information. (3) They contain noisy interfering items, making the final recommendation results inaccurate. As shown in Fig. 1, for item u_1, the relevant items are u_2, u_3, and u_4, while the noisy items are u_5 and u_6. If the current session is session 3, item transitions from other sessions can be used to predict the representation of session 3, such as $[u_1, u_3]$. However, using session 2 to predict session 3 introduces noise because $[u_5, u_6]$ in session 2 are irrelevant and cannot distinguish the noisy items.

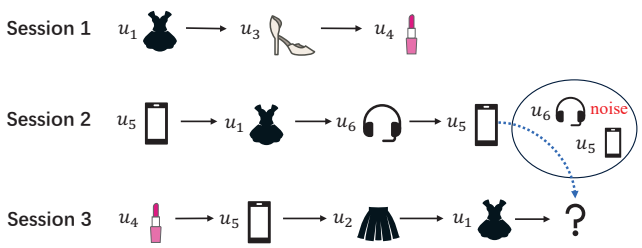


Fig. 1. Illustration of noise interference between sessions.

Consequently, in order to address the aforementioned challenges, we introduce an innovative framework SDGIA. Within this framework, we represent session sequences as a global undirected graph and session-directed graphs, and acquire global and local embeddings through global aggregator and session aggregator. The global aggregator extracts extensive contextual information from the global graph, while the session aggregator focuses on capturing fine-grained transition patterns within each session. Additionally, we implement a self-supervised learning mechanism to improve feature learning by optimizing the information representation between the global and session graphs. Furthermore, SDGIA preserves temporal information using learnable reverse position encoding. It en-

hances the quality of current session representations by combining a soft attention mechanism with a module for similar intent associations. This approach effectively captures complex item transition patterns and filters out noise interference in session data, thus significantly improving recommendation performance.

This study offers several key contributions:

- We tackle several major issues in existing session-based recommendation systems, including the neglect of complex item transition relationships, insufficient utilization of inter-session information, and inaccuracies in recommendation results due to noise interference.
- We propose a novel model called SDGIA, which constructs a global undirected graph and session-directed graphs to effectively capture transition patterns within and across sessions. The self-supervised learning mechanism enhances feature extraction capabilities, while the intention association module filters out noise to yield more precise item representations.
- Experiments on three datasets demonstrate that SDGIA significantly outperforms existing models, validating the effectiveness of the proposed approach.

2 Related Work

2.1 RNN-based SBR

RNNs are effective for sequential data and user behavior patterns. Hidasi et al.'s GRU4Rec [5] pioneered using RNNs for SBR, with GRUs modeling user preferences. Tan et al. [18] enhanced this approach with GRU4Rec+, which enhances recommendation effectiveness by capturing temporal dynamics. Li et al.'s NARM [10] integrates attention mechanisms to further enhance RNN performance for short and long durations. STAMP [11] captures users' immediate interests with a short-term memory network. ISLF [14] combines Variational Autoencoders (VAE) and RNNs to address changes in user interests. However, these methods mainly rely on sequential relationships, struggling to capture dependencies between distant items [2].

2.2 GNN-based SBR

In the past few years, GNNs have been applied to SBR owing to their robust capability to learn from graph-structured data [6]. Wu et al.'s SR-GNN [22] leverages GGNN to derive item representations from session sequences. GC-SAN [27] builds upon SR-GNN by incorporating a self-attention mechanism. Qiu et al.'s FGNN [13] employs multi-head attention to integrate information from adjacent nodes. Wang et al.'s MBGCN [8] employs a multi-behavior graph convolution network to consider user interaction behaviors within sessions. DHCN [23] introduces hypergraphs, focusing on hyperedge connectivity. Although GNNs effectively capture item relationships, they struggle with long-range dependencies and can suffer from overfitting and noise propagation when stacking multiple layers [3].

2.3 Self-supervised Learning

Self-supervised learning (SSL) has gained significant attention for its ability to generate supervision signals from unlabeled data, with wide applications in GNNs and recommendation systems. Contrastive learning [32], a core SSL technique, enhances model representation by maximizing mutual information between different data views [20]. In sequential recommendation, researchers address data sparsity through data augmentation and contrastive learning, as seen in the methods by Xie et al. [25] and Su et al.'s [16] graph-based polar contrastive learning. Additionally, RESTC [19] and COCO-SBRS [15] combine temporal information and causal inference to effectively tackle sparsity and temporal relationship issues. KMCLR [29] incorporates contrastive learning to optimize user and item representations, effectively capturing the relationships between multiple behaviors and alleviating the data sparsity problem. Ultimately, it demonstrates the great potential of self-supervised learning in enhancing recommendation system performance.

3 Preliminary

3.1 Research Objective

Consider $V = \{u_1, u_2, \ldots, u_m\}$, representing a collection of items, where m indicates the total quantity of possible items across all sessions. Each session s_t is an ordered sequence of interactions, denoted as $s = \{u_1^s, u_2^s, \ldots, u_l^s\}$, where u_i^s indicates the i-th item selected in session s, with l denoting the session duration. The dataset consists of multiple sessions $S = \{s_1, s_2, \ldots, s_n\}$, with n representing the total count of sessions.

Given session s, the objective is to recommend the top-N items from V that the user is expected to choose next, i.e., predict the next item u_{l+1}^s based on observed interactions within the session.

3.2 Graph Construction

Session Graph Construction: The session graph model captures session item embeddings by modeling sequential relationships within the current session. Given a session $s = \{u_1^s, u_2^s, \ldots, u_l^s\}$, we convert it to a directed graph $G_s = (V_s, E_s)$, with $V_s \subseteq V$ being the items clicked in session s and E_s representing the transitions. Each edge e_{ij}^s signifies a transition between item u_i^s and u_j^s within session s.

To accurately represent item relationships, we apply a normalized weight to each edge, determined by its frequency divided by the out-degree of the starting node, highlighting key transitions. Additionally, self-loops are included to capture self-transitions. From the directed graph G_s, two adjacency matrices $\mathbf{A}_{out} \in \mathbb{R}^{m \times m}$ and $\mathbf{A}_{in} \in \mathbb{R}^{m \times m}$ are constructed to capture complex session transitions.

Global Graph Construction: To utilize item transition information across all sessions, we introduce a global graph model that captures global-level item transitions. For any item $u_i^{S_p}$ in session S_p, its δ-neighbor set is defined as:

$$M_\delta(u_i^{S_p}) = \{u_k^{S_r} \mid u_i^{S_p} = u_m^{S_r} \in S_p \cap S_r; k \in [m-\delta, m+\delta]; S_p \neq S_r\}, \quad (1)$$

where m is the position of item $u_i^{S_p}$ in session S_r, and δ controls the transition scope between $u_i^{S_p}$ and items in session S_r. The neighbors of item u (i.e., $\mathcal{N}_g(u)$) are defined similarly to $\mathcal{M}_\delta(u)$.

Using the δ-neighbor set, we construct a global graph $G_g = (V_g, E_g)$, where V_g includes all items from all sessions, and E_g consists of edges representing global item transitions. Each item in the global graph is encoded into a unified d-dimensional embedding space. The initial embedding $\mathbf{z}_{u_i} \in \mathbb{R}^{|d|}$ is obtained using one-hot encoding and serves as the input to the global layer.

4 Approach

We introduce a novel SBR system named SDGIA. As shown in Fig. 2, we will elaborate on the implementation details of each component.

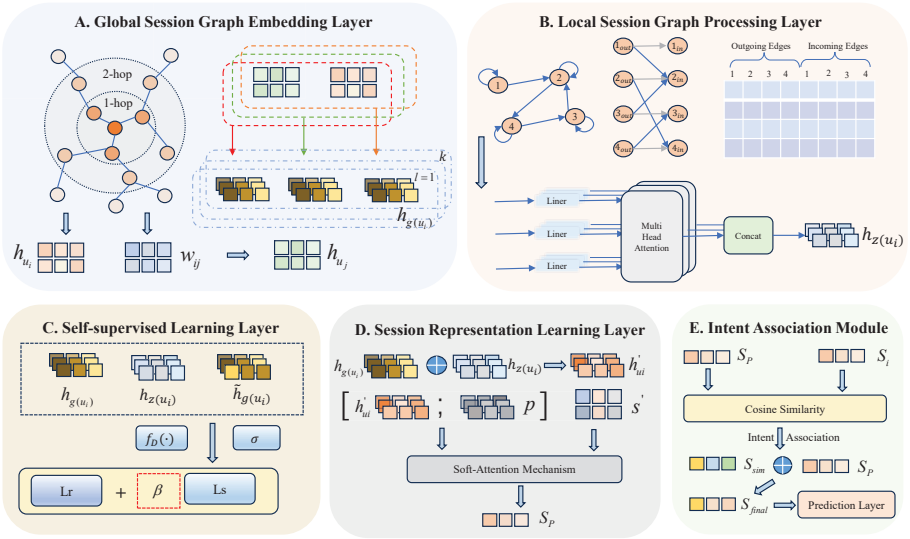


Fig. 2. The model architecture of SDGIA framework.

4.1 Global Session Graph Embedding Layer

The approach builds upon the architectures of Graph Convolutional Networks (GCN) and Graph Attention Networks (GAT).

Items can appear in multiple sessions, accumulating valuable transition data that enhances prediction accuracy. To extract the first-order neighboring features of an item, we employ a session-aware attention mechanism to evaluate the significance of each neighbor within the δ-neighbor set. This is important because not every item in the δ-neighbor set holds equal relevance to the user's preference during the current session. The features of the neighbors are linearly combined according to their attention scores.

$$\mathbf{h}_{\mathcal{N}_g(u_i)} = \sum_{u_j \in \mathcal{N}_g(u_i)} \beta(u_i, u_j)\mathbf{h}_{u_j}, \tag{2}$$

Here, $\beta(u_i, u_j)$ represents the normalized weight between nodes u_i and u_j, calculated based on the frequency of transitions between these nodes and the out-degree of u_i.

The session feature $\mathbf{s}$ is calculated by averaging the item embeddings in the current session:

$$\beta(u_i, u_j) = \mathbf{q}_1^T \text{LeakyReLU}\left(\mathbf{W}_1\left((\mathbf{s} \odot \mathbf{h}_{u_j}) \parallel w_{ij}\right)\right), \quad \mathbf{s} = \frac{1}{|S|}\sum_{u_i \in S} \mathbf{h}_{u_i}, \tag{3}$$

In Eq. (3), $\odot$ denotes element-wise multiplication, while $\parallel$ represents concatenation. The term w_{ij} specifies the edge weight between items u_i and u_j within the global network, signifying their connection strength. Additionally, the matrices $\mathbf{W}_1 \in \mathbb{R}^{d+1\times d+1}$ and $\mathbf{q}_1 \in \mathbb{R}^{d+1}$ serve as tunable parameters, allowing the model to learn optimal transformations for each step in the calculation.

Unlike mean pooling, our method utilizes session-aware attention to emphasize items most relevant to the ongoing session. The attention scores are adjusted using the softmax function.

After obtaining the weighted features of an item's neighboring nodes, we combine the item's representation $\mathbf{h}_{u_i}$ with its neighborhood representation $\mathbf{h}_{\mathcal{N}_g(u_i)}$ using the following aggregation function:

$$\mathbf{h}'_{g(u_i)} = \text{ReLU}\left(\mathbf{W}_2\left[\mathbf{h}_{u_i} \parallel \mathbf{h}_{\mathcal{N}_g(u_i)}\right]\right), \tag{4}$$

ReLU is the activation function used here, and $\mathbf{W}_2 \in \mathbb{R}^{d\times 2d}$ is the transformation matrix. This single-layer aggregation allows the item's representation to depend on itself and its immediate neighbors. To capture higher-order connectivity, we extend the aggregation process to multiple layers. The embedding of an item at the k-th layer is defined as follows:

$$\mathbf{h}_k(u_i) = \text{agg}(\mathbf{h}_{(k-1)}(u_i), \mathbf{h}_{(k-1)}(\mathcal{N}_g(u_i))), \tag{5}$$

The item representation from the previous layer $\mathbf{h}_{(k-1)}(u_i)$, is initialized as $\mathbf{h}_{u_i}$ in the first layer. This iterative process includes information from neighbors within k hops, enhancing the depiction of the current session. To avoid overfitting, dropout is applied to the aggregated result.

4.2 Local Session Graph Processing Layer

To capture local preferences within a session, we utilize GGNNs with GRUs to manage information flow. The update rules for node embeddings are as follows:

$$\mathbf{p}_i^s = \mathbf{A}_s[i,:] \cdot [\mathbf{z}_{u_1^s}, \mathbf{z}_{u_2^s}, \cdots, \mathbf{z}_{u_l^s}]^T \mathbf{W}_3 + \mathbf{b}_1, \tag{6}$$

$$\mathbf{z}_i^s = \sigma(\mathbf{W}_z \mathbf{p}_i^s + \mathbf{U}_z \mathbf{z}_{u_i^s}), \tag{7}$$

$$\mathbf{r}_i^s = \sigma(\mathbf{W}_r \mathbf{p}_i^s + \mathbf{U}_r \mathbf{z}_{u_i^s}), \tag{8}$$

$$\tilde{\mathbf{z}}_i^s = \tanh(\mathbf{W}_o \mathbf{p}_i^s + \mathbf{U}_o(\mathbf{r}_i^s \odot \mathbf{z}_{u_i^s})), \tag{9}$$

$$\mathbf{z}_{u_i}^{out} = (1 - \mathbf{z}_i^s) \odot \mathbf{z}_{u_i^s} + \mathbf{z}_i^s \odot \tilde{\mathbf{z}}_i^s, \tag{10}$$

where $\mathbf{W}_3$ is the weight matrix, $\mathbf{z}_i^s$ and $\mathbf{r}_i^s$ act as the update and reset gates, respectively. The sigmoid function is denoted by σ, element-wise multiplication by $\odot$, $\mathbf{z}_{u_i^s}$ represents the latent vector for node u_i^s, and $\mathbf{A}_s$ is the adjacency matrix.

To comprehensively represent user preferences and gather information from various perspectives, we utilize a multi-head attention mechanism. The embeddings of local items are updated as follows:

$$\mathbf{attn}_i = \text{softmax}\left(\frac{\mathbf{Q}_i \mathbf{z}_{u_i}^{out} \cdot (\mathbf{K}_i \mathbf{z}_{u_j}^{out})^T}{\sqrt{d_k}}\right) \mathbf{V}_i, \tag{11}$$

$$\mathbf{h}_{z(u_i)} = \text{concat}(\mathbf{attn}_1, \mathbf{attn}_2, \cdots, \mathbf{attn}_h)\mathbf{W}_{attn}, \tag{12}$$

where $\mathbf{Q}_i$, $\mathbf{K}_i$ and $\mathbf{V}_i$ are the query, key, and value matrices for the i-th head, $\sqrt{d_k}$ is a scaling factor, and $\mathbf{W}_{attn}$ combines the multi-head outputs. To prevent overfitting, dropout is applied to the final representation of local preferences.

4.3 Self-supervised Learning Layer

We generate two sets of session embeddings from the global session layer and the local session layer, respectively. We treat the corresponding session embedding pairs as positive samples and the randomly combined session embedding pairs as negative samples. By employing contrastive learning, we can enhance the mutual information between these two sets of embeddings, thereby boosting model performance.

We employ InfoNCE as the loss function, which is achieved by computing the binary cross-entropy for positive and negative samples. The detailed formula is given below:

$$L_s = -\log \sigma(f_D(\mathbf{h}_{g(u_i)}, \mathbf{h}_{z(u_i)})) - \log \sigma(1 - f_D(\tilde{\mathbf{h}}_{g(u_i)}, \mathbf{h}_{z(u_i)})), \tag{13}$$

where $\mathbf{h}_{g(u_i)}$ and $\mathbf{h}_{z(u_i)}$ represent session embeddings generated by the global session layer and local session layer after dropout, respectively. $\tilde{\mathbf{h}}_{g(u_i)}$ is a negative sample generated by randomly shuffling the rows and columns of $\mathbf{h}_{g(u_i)}$.

Ultimately, we integrate the recommendation and self-supervised tasks into a unified learning framework. The primary focus is on the recommendation task, with the self-supervised task serving as a supplementary component.

$$f_D(\mathbf{h}_{g(u_i)}, \mathbf{h}_{z(u_i)}) = \mathbf{h}_{g(u_i)}^T \mathbf{h}_{z(u_i)}, \quad L = L_r + \beta L_s, \tag{14}$$

where $f_D(\cdot)$ evaluates the alignment between two vectors via their dot product. L_r denotes the recommendation task loss, L_s represents the self-supervised task loss, and β adjusts the impact of the self-supervised task on the overall loss.

4.4 Session Representation Learning Layer

For each item, we compute the combined representation by summing its global embedding $\mathbf{h}_{g(u_i)}$ and session embedding $\mathbf{h}_{z(u_i)}$. To enhance prediction accuracy, we incorporate reverse position information. Items selected towards the end of a session usually have greater importance for forecasting the next item. A trainable position embedding matrix $\mathbf{P}$ is employed to compute the enhanced item embeddings as follows:

$$\mathbf{h}'_{u_i} = \mathbf{h}_{g(u_i)} + \mathbf{h}_{z(u_i)}, \quad \mathbf{e}_i = \tanh\left(\mathbf{W}_4(\mathbf{h}'_{u_i} \parallel \mathbf{p}_{l-i+1}) + \mathbf{b}_2\right), \tag{15}$$

where $\mathbf{p}_{l-i+1}$ is the reverse position embedding for position i in a session of length l, and $\mathbf{W}_4 \in \mathbb{R}^{d \times 2d}$ and $\mathbf{b}_2 \in \mathbb{R}^d$ are parameters that can be trained.

After deriving the enhanced item embeddings $\mathbf{e}_i$, an attention mechanism with soft weights is applied to derive the session vector. The attention scores are derived by considering both the enhanced item embeddings and the average session representation $\mathbf{s}'$. The final session embedding $\mathbf{S}_p$ is subsequently calculated by taking a weighted sum of the item embeddings:

$$\mathbf{s}' = \frac{1}{l}\sum_{i=1}^{l} \mathbf{h}'_{u_i}, \quad \alpha_i = \mathbf{q}_2^T \sigma\left(\mathbf{W}_5 \mathbf{e}_i + \mathbf{W}_6 s' + \mathbf{b}_3\right), \quad \mathbf{S}_p = \sum_{i=1}^{l} \alpha_i \mathbf{h}'_{u_i}, \tag{16}$$

where $\mathbf{W}_5, \mathbf{W}_6 \in \mathbb{R}^{d \times d}$ and $\mathbf{q}_2, \mathbf{b}_3 \in \mathbb{R}^d$ are parameters that can be trained, σ represents the sigmoid function, and α_i represents the corresponding weights learned through a soft-attention mechanism.

4.5 Intent Association Module

Existing methods often overlook the similarity of intents between sessions, leading to noise interference in the information. Therefore, we propose a similar-intent neighbor module primarily aimed at filtering out irrelevant sessions. Initially, sessions analogous to the current one are chosen by computing similarity scores between session embeddings. Specifically, the similarity score is computed using the cosine similarity formula:

$$Sim(\mathbf{S}_p, \mathbf{S}_i) = \frac{(\mathbf{S}_p^T \mathbf{S}_i + \mathbf{b}_4)}{\sqrt{\|\mathbf{S}_p\|^2 + \epsilon} \cdot \sqrt{\|\mathbf{S}_i\|^2 + \epsilon}}, \tag{17}$$

where $\mathbf{S}_p$ denotes the current session's embedding, $\mathbf{S}_i$ indicates the embedding of a candidate session, $\mathbf{b}_4$ is a bias term, and ϵ is a minor constant to avoid division by zero. We rank the candidate sessions by their similarity scores and choose the top F sessions as the similar sessions.

We carry out a weighted summation of the chosen similar sessions to derive the final similar session representation. The weights are normalized similarity scores obtained through the softmax function:

$$\beta_i = \frac{\exp(\gamma \cdot Sim(\mathbf{S}_p, \mathbf{S}_i) + \mathbf{b}_5)}{\sum_{j=1}^{F} \exp(\gamma \cdot Sim(\mathbf{S}_p, \mathbf{S}_j) + \mathbf{b}_6)}, \quad \mathbf{S}_{sim} = \sum_{i=1}^{F} \beta_i \mathbf{S}_i, \tag{18}$$

where γ is a parameter that controls the influence of similarity, $\mathbf{b}_5$ and $\mathbf{b}_6$ are bias terms, and F represents the count of sessions with analogous intents. $\mathbf{S}_{sim}$ is the final similar session representation.

After obtaining the final session embedding $\mathbf{S}_{final} = \mathbf{S}_p + \mathbf{S}_{sim}$, we proceed to the prediction layer. The score for each candidate item is determined by taking the dot product of its embedding $\mathbf{h}_{u_i}$ and the ultimate session vector $\mathbf{S}_{final}$. A softmax function is applied to the scores to derive the probability distribution across all potential items:

$$\hat{\mathbf{z}}_i = \sigma\left(\mathbf{W}_7^T(\mathbf{S}_{final} \parallel \mathbf{h}_{u_i}) + \mathbf{b}_7\right), \quad \hat{\mathbf{y}}_i = \text{softmax}(\hat{\mathbf{z}}_i), \tag{19}$$

where $\hat{\mathbf{z}}_i \in \mathbb{R}^m$ denotes the recommendation scores for the candidate items, and $\hat{\mathbf{y}}_i \in \mathbb{R}^m$ indicates the likelihood of each item being selected as the next click within the session.

To train the model, the loss function is formulated as the cross-entropy between the predicted probabilities and the actual outcomes, with an added regularization term:

$$L_r = -\sum_{i=1}^{m} \mathbf{y}_i \log(\hat{\mathbf{y}}_i) + (1 - \mathbf{y}_i) \log(1 - \hat{\mathbf{y}}_i), \tag{20}$$

where $\mathbf{y}_i$ represents the one-hot encoded vector of the true item, $\hat{\mathbf{y}}_i$ denotes the predicted probabilities.

5 EXPERIMENTS

Based on the following five research questions, we performed comprehensive experiments to assess the efficacy of our proposed SDGIA method: **(RQ1):** Does SDGIA outperform the state-of-the-art session-based recommendation baseline models on three real-world datasets? **(RQ2):** Do the proposed modules enhance the recommendation performance? **(RQ3):** Can domain information enhance the performance of SDGIA? **(RQ4):** How does SDGIA perform under different

aggregation operations? **(RQ5):** How do different hyperparameter settings affect the accuracy of SDGIA?

Table 1. Statistics of the datasets.

Dataset	#clicks	#train	#test	#items	avg.len.
Diginetica	982,961	719,470	60,858	43,097	5.12
Tmall	818,479	351,268	25,898	40,728	6.69
Nowplaying	1,367,963	825,304	89,824	60,417	7.42

5.1 Experimental Settings

Datasets. We adopt three benchmark datasets [21,22,30], namely **Diginetica**[1], **Tmall**[2], and **Nowplaying**[3]. These datasets are widely used in session-based recommendation tasks. The Diginetica dataset comes from the CIKM Cup 2016 and contains typical transaction data. The Tmall dataset is sourced from the IJCAI-15 competition and includes shopping records of anonymous users on the Tmall online shopping platform. The Nowplaying dataset records users' music listening behaviors and is derived from a Kaggle competition. Table 1 summarizes the statistics of the preprocessed datasets.

Evaluation Metrics. We use P@N and MRR@N as evaluation metrics [11,22]. P@N measures the proportion of correctly predicted items in the top N recommendations. MRR@N calculates the average of the reciprocal ranks of the correctly predicted items. In our experiments, we set N to 10 and 20.

Experimental Parameter Setting. We adopt the basis of previous work, fixing the dimension of the latent vector to 100 and setting the batch size to 100. In the local session layer, the number of GGNN layers is set to 1. For fair comparison, we maintain the same hyperparameter settings as previous work [21,22], including the learning rate and its decay. Our model parameters are initialized using a Gaussian distribution with a mean of 0 and a standard deviation of 0.1. The learning rate is 0.001, decaying by 0.1 every three training epochs. The L2 regularization coefficient is set to 10^{-5}. We use 10% of the training set as the validation set. The Adam algorithm is employed for optimization. Additionally, we introduce a neighbor set parameter, setting the number of neighbors to 12 and the maximum distance of adjacent items to 3 [7].

Comparison Models. To assess the effectiveness of the SDGIA model, we benchmark it against several other recommendation models:

[1] http://cikm2016.cs.iupui.edu/cikm-cup
[2] https://tianchi.aliyun.com/dataset/dataDetail?dataId=42
[3] http://dbis-nowplaying.uibk.ac.at/#nowplaying

- **GRU4Rec** [5]: Gated Recurrent Units for Recommendation, using GRU-based RNN for session-based recommendation.
- **NARM** [10]: Neural Attentive Recommendation Machine, utilizing RNNs and attention mechanism for session-based recommendation.
- **SR-GNN** [22]: Session-based Graph Neural Network recommendation, leveraging GNN to model intricate item transitions within sessions.
- **GCE-GNN** [21]: Graph Neural Network with Global Context Enhancement, incorporating global context into GNN recommendation.
- **S^2-DHCN** [23]: Session-based Dual Hypergraph Convolutional Network, constructing hypergraphs and line graphs, and combining SSL for session-based recommendation.
- **COTREC** [24]: Enhances SBR by using SSL with co-training to preserve complete session information and achieve state-of-the-art performance.
- **DIDN** [34]: Enhances SBR by modeling dynamic user intents and filtering out noisy clicks, achieving notable performance improvements in experiments.
- **Disen-GNN** [9]: Enhances SBR by capturing user intent at the factor level, outperforming existing methods in experiments.
- **Atten-Mixer** [33]: Improves SBR by removing redundant GNN propagations and leveraging multi-level attention readouts, demonstrating significant effectiveness and efficiency in experiments and real-world applications.

5.2 Model Performance Comparison (RQ1)

To illustrate the overall performance of the proposed SDGIA model, we benchmarked it against existing recommendation techniques. From Table 2, the following observations can be made:

Table 2. Comparison of methods on different datasets.

Dataset	Diginetica				Tmall				Nowplaying			
Method	P@10	MRR@10	P@20	MRR@20	P@10	MRR@10	P@20	MRR@20	P@10	MRR@10	P@20	MRR@20
GRU4Rec(ICLR'16)	17.93	7.73	30.79	8.22	9.47	5.78	10.93	5.89	6.74	4.40	7.92	4.48
NARM(CIKM'17)	35.44	15.13	48.32	16.00	19.17	10.42	23.30	10.70	13.60	6.62	18.59	6.93
SR-GNN(AAAI'19)	38.42	16.89	51.26	17.78	23.41	13.45	27.57	13.72	14.17	7.15	18.87	7.47
GCE-GNN(SIGIR'20)	41.16	18.15	54.22	19.04	28.01	15.08	33.42	15.42	16.94	8.03	22.37	8.40
S^2-DHCN(AAAI'21)	40.21	17.59	53.66	18.51	26.22	14.60	31.42	15.05	17.35	7.87	23.50	8.18
COTREC(CIKM'21)	41.88	18.16	54.18	19.07	30.62	17.65	36.35	18.04	18.54	8.68	24.34	9.36
DIDN(IPM'22)	42.99	19.08	56.23	20.00	29.09	16.61	34.01	17.16	17.65	8.30	23.25	8.68
Disen-GNN(TKDE'23)	40.61	18.08	53.79	18.99	24.22	14.63	28.97	14.93	15.93	8.04	22.22	8.22
Atten-Mixer(WSDM'23)	41.47	17.94	55.12	18.90	29.27	16.55	34.98	16.72	16.57	8.12	22.20	8.49
SDGIA	**60.53**	**22.17**	**60.55**	**22.18**	**41.35**	**20.16**	**41.35**	**20.17**	**28.76**	**9.62**	**28.76**	**9.62**
Improv.	40.80%	16.19%	7.68%	10.90%	35.04%	14.22%	13.76%	11.81%	55.12%	10.83%	18.16%	2.78%

Compared to traditional RNN models, GNN-based models exhibit stronger capabilities in capturing complex graph-structured data. SR-GNN performs exceptionally well on the Tmall and Nowplaying datasets, while GCE-GNN excels on the Diginetica dataset, indicating the greater potential of graph neural networks in modeling item transitions in session data. Although COTREC performs well on multiple datasets, its SSL capabilities are limited when handling larger datasets. Nevertheless, it performs exceptionally on the Tmall dataset, validating the importance of capturing complex structural patterns across sessions.

Our proposed model, SDGIA achieves state-of-the-art performance across all metrics. This success can be attributed to its simultaneous consideration of global and local information, enhanced feature capture and generalization capabilities through SSL mechanisms, and effective noise filtering through its intention association module.

5.3 Impact of Proposed Modules (RQ2)

To investigate the importance of each SDGIA component and its influence on performance, we created five variants and performed ablation studies on three datasets.

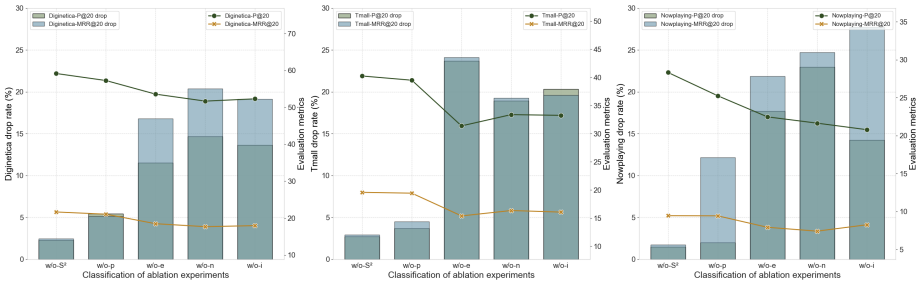


Fig. 3. Performance of different variants on three datasets.

- **w/o-S^2**: We exclude the self-supervised learning tasks.
- **w/o-p**: We remove the position embedding, the model dosen't take into account the positional information within the session.
- **w/o-e**: We use the Euclidean distance to compute the similarity matrix, thereby replacing the cosine similarity.
- **w/o-n**: We do not compute any similarity or distance matrix, instead we randomly select a fixed number of neighbors.
- **w/o-i**: We remove the intention association module.

As shown in Fig. 3, in the Nowplaying dataset, the effect of the w/o-S^2 is not evident, primarily due to the complex user behavior patterns and rich information in this dataset, which makes self-supervised learning less effective in enhancing model performance. The other four variants significantly decreased performance, highlighting their importance in supplementing contextual information and enhancing model expressiveness.

5.4 Domain Information Effectiveness (RQ3)

In the SDGIA model, higher-order interaction information is captured through a global feature encoder and integrated with session-level features to produce more precise recommendation outcomes. To validate its efficacy, we developed the following variants:

- **w/o-g**: Exclude the global feature encoder.
- **1-hop**: Limit the global feature encoder to one hop.
- **2-hop**: Limit the global feature encoder to two hops.

Table 3. Performance of different variants on three datasets.

Datasets	Diginetica		Tmall		Nowplaying	
Metrics	P@20	MRR@20	P@20	MRR@20	P@20	MRR@20
w/o-g	60.43	22.10	41.16	19.75	24.63	9.38
1-hop	60.15	22.06	40.63	19.50	28.76	**9.62**
2-hop	**60.55**	**22.18**	**41.35**	**20.17**	**29.13**	9.45

The data in Table 3 indicates that the global feature encoder has a significant impact on the performance of the SDGIA model. w/o-g leads to decreased performance across all datasets, especially on the Nowplaying dataset, highlighting its importance for datasets with complex user behavior patterns. 1-hop reduces performance, but in terms of MRR@20 on the Nowplaying dataset, the 1-hop result (9.62) outperforms the 2-hop result (9.45). This phenomenon may be related to specific user behavior patterns within that dataset, indicating that in certain cases, a simplified global perspective can better capture user intentions. 2-hop results in performance close to the full model, demonstrating that multiple hops help fully exploit relational information in the graph.

5.5 Aggregation Operations Evaluation (RQ4)

To assess the efficacy of various aggregation techniques, we devised and evaluated four operations: sum, gating mechanism, average pooling, and concatenation. The results of these experiments are presented in Table 4.

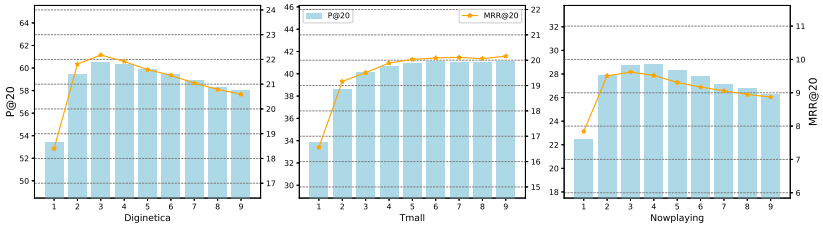
Table 4. Performance of different aggregation operations.

Datasets	Diginetica		Tmall		Nowplaying	
Metrics	P@20	MRR@20	P@20	MRR@20	P@20	MRR@20
Sum	**60.55**	**22.18**	**41.35**	**20.17**	28.76	9.62
Gating	60.13	22.08	40.89	19.97	29.07	9.61
Mean	60.38	22.18	41.14	20.12	**28.97**	**9.69**
Concat	59.08	21.46	36.90	18.24	25.44	9.65

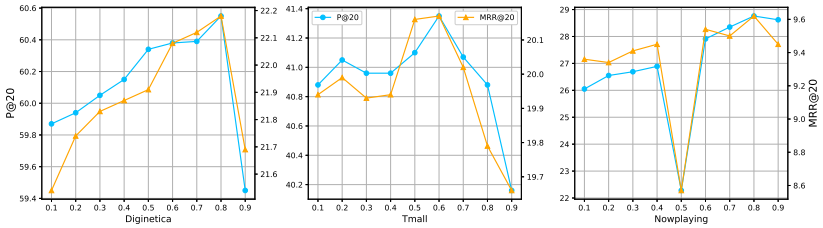
The sum operation performs best on the Diginetica and Tmall datasets, indicating that directly adding local and global features can effectively capture patterns in these datasets. The mean operation performs best on the Nowplaying dataset, demonstrating its stability in the presence of noisy data. The gating mechanism shows relatively balanced performance. The concatenation operation performs the worst across all datasets, likely due to introducing more noise or irrelevant information, which degrades the model's predictive capability.

5.6 Hyperparameter Sensitivity Analysis (RQ5)

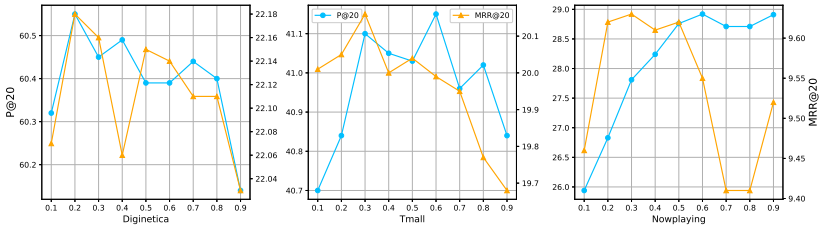
Impact of Similar Session Count. When user behavior in a dataset is more concentrated, a small number of similar sessions can capture the main information, while too many will introduce noise. However, for the Tmall dataset, where user behavior is more complex, more similar sessions are needed to fully capture user interests.



(a) Impact of similar session count.



(b) Impact of dropout intent configuration.



(c) Impact of dropout global configuration.

Fig. 4. Hyperparameter sensitivity analysis of the SDGIA model.

Analysis of Dropout Hyperparameters. Fig. 4(b) shows the dropout intent rate, which reduces noise by dropping certain user intention features. In contrast, Fig. 4(c) focuses on preventing overfitting by dropping a portion of input features. The dropout intent rate is stable across datasets, indicating good generalization. However the global dropout rate varies significantly, particularly in the Nowplaying dataset. When the dropout rate exceeds 0.5, MRR@20 decreases while P@20 increases. This may occur because global dropout makes the model focus on items closely related to user interests, enhancing P@20 but negatively impacting MRR@20 due to ranking changes. The complexity and noise in the Nowplaying dataset suggest that global dropout can improve accuracy but may lower the ranking of expected items, affecting MRR@20.

6 Conclusion

In this work, we introduce the SDGIA approach for session-based recommendation. The model captures the complex transition relationships between items by constructing dual graphs. We introduce a self-supervised learning mechanism to significantly enhance feature capture and generalization capabilities. Additionally, we design an intention association module to effectively filter noise interference, generating more accurate item representations. Empirical findings demonstrate that SDGIA markedly surpasses current leading models across three commonly utilized datasets. These results validate the superior performance of SDGIA in handling complex transition relationships and noise interference. In future work, we plan to further explore how to enhance the model's ability in semantically aware item transitions.

Acknowledgement. This work is supported in part by the "14th Five-Year Plan" Civil Aerospace Pre-Research Project of China under Grant No. D020101, the Natural Science Foundation of China No. 62302213, Innovation Funding of Key Laboratory of Intelligent Decision and Digital Operations No. NJ2023027, Ministry of Industrial and Information Technology Project of Hebei Key Laboratory of Software Engineering, No. 22567637H, the Natural Science Foundation of Jiangsu Province under Grant No. BK20210280.

References

1. Chen, W., Cai, F., Chen, H., Rijke, M.D.: Joint neural collaborative filtering for recommender systems. In: TOIS. **37**(4), 1–30 (2019)
2. Chen, X., Xu, H., Zhang, Y., Tang, J., et al.: Sequential recommendation with user memory networks. In: WSDM. pp. 108–116 (2018)
3. He, X., Deng, K., et al.: Lightgcn: Simplifying and powering graph convolution network for recommendation. In: SIGIR. pp. 639–648 (2020)
4. He, X., Liao, L., Zhang, H., Nie, L., Hu, X., Chua, T.S.: Neural collaborative filtering. In: WWW. pp. 173–182 (2017)
5. Hidasi, B., Karatzoglou, A., Baltrunas, L., Tikk, D.: Session-based recommendations with recurrent neural networks. arXiv preprint arXiv:1511.06939 (2015)
6. Huang, C., Chen, J., et al.: Graph-enhanced multi-task learning of multi-level transition dynamics for session-based recommendation. In: AAAI. vol. 35, pp. 4123–4130 (2021)
7. Huang, L., Li, R., et al.: Sequence-aware graph neural network incorporating neighborhood information for session-based recommendation. International Journal of Computational Intelligence Systems **17**(1), 32 (2024)
8. Jin, B., Gao, C., He, X., Jin, D., Li, Y.: Multi-behavior recommendation with graph convolutional networks. In: SIGIR. pp. 659–668 (2020)
9. Li, A., Cheng, Z., Liu, F., Gao, Z., Guan, W., Peng, Y.: Disentangled graph neural networks for session-based recommendation. In: TKDE. **35**(8), 7870–7882 (2022)
10. Li, J., Ren, P., Chen, Z., Ren, Z., Lian, T., Ma, J.: Neural attentive session-based recommendation. In: CIKM. pp. 1419–1428 (2017)

11. Liu, Q., Zeng, Y., Mokhosi, R., et al.: Stamp: short-term attention/memory priority model for session-based recommendation. In: SIGKDD. pp. 1831–1839 (2018)
12. Pan, Z., Cai, F., Chen, W., et al.: Star graph neural networks for session-based recommendation. In: CIKM. pp. 1195–1204 (2020)
13. Qiu, R., Li, J., Huang, Z., Yin, H.: Rethinking the item order in session-based recommendation with graph neural networks. In: CIKM. pp. 579–588 (2019)
14. Song, J., Shen, H., Ou, Z., et al.: Islf: Interest shift and latent factors combination model for session-based recommendation. In: IJCAI. pp. 5765–5771 (2019)
15. Song, W., Wang, S., et al.: A counterfactual collaborative session-based recommender system. In: WWW. pp. 971–982 (2023)
16. Su, T.T., Wang, C.D., et al.: Hierarchical alignment with polar contrastive learning for next-basket recommendation. In: TKDE. **36**(1), 199–210 (2024)
17. Sun, X., Cheng, H., et al.: All in one: Multi-task prompting for graph neural networks. In: SIGKDD. pp. 2120–2131 (2023)
18. Tan, Y.K., Xu, X., Liu, Y.: Improved recurrent neural networks for session-based recommendations. In: RecSys. pp. 17–22 (2016)
19. Wan, Z., Liu, X., Wang, B., et al.: Spatio-temporal contrastive learning-enhanced gnns for session-based recommendation. In: TOIS. (2024)
20. Wang, C., Ma, W., et al.: Sequential recommendation with multiple contrast signals. In: TOIS. **41**(1), 1–27 (2023)
21. Wang, Z., Wei, W., Cong, G., et al.: Global context enhanced graph neural networks for session-based recommendation. In: SIGIR. pp. 169–178 (2020)
22. Wu, S., Tang, Y., Zhu, Y., Wang, L., Xie, X., Tan, T.: Session-based recommendation with graph neural networks. In: AAAI. vol. 33, pp. 346–353 (2019)
23. Xia, X., Yin, H., Yu, J., et al.: Self-supervised hypergraph convolutional networks for session-based recommendation. In: AAAI. vol. 35, pp. 4503–4511 (2021)
24. Xia, X., Yin, H., Yu, J., Shao, Y., Cui, L.: Self-supervised graph co-training for session-based recommendation. In: CIKM. pp. 2180–2190 (2021)
25. Xie, X., Sun, F., et al.: Contrastive learning for sequential recommendation. In: ICDE. pp. 1259–1273. IEEE (2022)
26. Xin, X., Yang, L., Zhao, Z., et al.: On the effectiveness of unlearning in session-based recommendation. In: WSDM. pp. 855–863 (2024)
27. Xu, C., Zhao, P., Liu, Y., et al.: Graph contextualized self-attention network for session-based recommendation. In: IJCAI. vol. 19, pp. 3940–3946 (2019)
28. Xu, Z., Wu, W., Yin, Z., et al.: Bpgnn-sbr: Behavior progressive graph neural networks for session-based recommendation. In: APWeb. pp. 492–503 (2024)
29. Xuan, H., Liu, Y., Li, B., Yin, H.: Knowledge enhancement for contrastive multi-behavior recommendation. In: WSDM. pp. 195–203 (2023)
30. Yu, F., Zhu, Y., Liu, Q., et al.: Tagnn: Target attentive graph neural networks for session-based recommendation. In: SIGIR. pp. 1921–1924 (2020)
31. Yu, J., Gao, M., Li, J., et al.: Adaptive implicit friends identification over heterogeneous network for social recommendation. In: CIKM. pp. 357–366 (2018)
32. Yufang, L., Shaoqing, W., Keke, L., et al.: Channel-enhanced contrastive cross-domain sequential recommendation. In: DSE pp. 1–16 (2024)
33. Zhang, P., Guo, J., et al.: Efficiently leveraging multi-level user intent for session-based recommendation via atten-mixer network. In: WSDM. pp. 168–176 (2023)
34. Zhang, X., Lin, H., Xu, B., et al.: Dynamic intent-aware iterative denoising network for session-based recommendation. In: IPM. **59**(3), 102936 (2022)
35. Zhou, G., Mou, N., Fan, Y., et al.: Deep interest evolution network for click-through rate prediction. In: AAAI. vol. 33, pp. 5941–5948 (2019)