Applied Intelligence (2023) 53:17292–17309
https://doi.org/10.1007/s10489-022-04399-y

DAS-GNN: Denoising autoencoder integrated with self-supervised learning in graph neural network-based recommendations

Jiuqian Dai[1] · Weihua Yuan[1] · Chen Bao[1] · Zhijun Zhang[1]

Accepted: 11 December 2022 / Published online: 28 December 2022

Abstract
To enhance the recommendation performance, session-based recommendations typically model based on graph neural networks (GNN). These models use the most recently clicked item as the user's short-term interest, as well as the query vector in the attention mechanism. Based on it, the attention score is calculated with the remaining items to obtain the user's long-term interest. However, the obtained representation of long-term interest is one-sided. Furthermore, unlike other recommendation technology, such as collaborative filtering that includes the user's entire history information, the session-based recommendation is more vulnerable to data sparsity. Existing models primarily make predictions based on observable user-item interactions and ignore items not interacted with by users. To address the aforementioned issues, we propose the denoising autoencoder integrated with self-supervised learning (SSL) in graph neural networks (DAS-GNN). In DAS-GNN, the query extraction module based on denoising autoencoder can mine multiple user interests and assist long-term interest to express user needs more comprehensively. We propose an effective way of dividing positive and negative samples in the SSL module and use adaptive thresholds to mine negative hard samples, thereby improving training efficiency and alleviating data sparsity. Extensive experiments demonstrate that the proposed DAS-GNN outperforms state-of-the-art models on four benchmarks. The source code is available at: https://github.com/daijiuqian/DAS-GNN.

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

1 Introduction

The problem of information overload makes it increasingly difficult for users to mine useful information in the age of information explosion, and recommender systems can assist users in discovering more needed information. Traditional recommendation models [1–3] typically use the user's historical interactions to learn his long-term static preferences, with the potential assumption that these interactions represent his current interest [4]. This assumption is usually inconsistent with the actual situation because the user's next interaction is primarily determined by his short-term interest, which accounts for only a small proportion of all his historical interactions [4]. Session-based recommendation, an emerging direction in recommender systems, focuses on modeling the user's preference based on short-term interactions, compensating for the flaw of traditional models that treat users' historical interactions equally.

Session-based recommendation predicts the user's subsequent interactions based on a session, which is a set of items in a sequence. With the popularity of GNN, researchers [5–9] convert a single session into a graph structure, using GNN to extract user preferences and make recommendations. Wu et al. [6] pointed out that in the session, GNN can learn more complex transformation relations. Xia et al. [10] proposed that by combining GNN and session, higher-order features can be learned while weakening the influence of strict-order relationships. According to Chen et al. [11], GNNs are becoming increasingly popular in session-based recommendation and have achieved state-of-the-art performance. Although GNN-based models have significant advances in comparison to traditional models, currently, they use the last interacted item in the session as the user's short-term interest and as the query vector in the attention mechanism. Furthermore, the attention score is calculated using the query vector and the remaining items, and the user's long-term interest is determined by

Jiuqian Dai and Weihua Yuan these authors contributed to the work equally and should be regarded as co-first authors

✉ Zhijun Zhang
zzjsdcn@163.com

[1] School of Computer Science and Technology, Shandong Jianzhu University, Jinan 250001, Shandong, China

fusing the item representation based on the attention score [6, 11, 12]. However, in real scenarios, this method has limitations, as demonstrated in detail in Fig. 1.

Figure 1 illustrates an example of a lady shopping. As shown in Fig. 1, in the time row the user's interaction moments are [t_1, t_2, t_3, t_4, t_5, *last click*]. The items and their representations at the corresponding moment are expressed by the item and embedding rows, respectively. In the embedding row, we use color to represent the similarity between item representations. The last item the user interacts with is the green dress at the last click moment, and its representation is used to denote her short-term interest. The label to the right of the time row represents the item to be predicted. The number next to the black arrow represents the difference in attention scores between short-term interest and item representations. According to the session data, the user is undecided between purchasing shoes and dresses, indicating that she has two different interests. The model's label is red high-heeled shoes. Suppose the user's short-term interest (v_6) is used as the query vector for extracting the long-term interest. Because the representation of her short-term interest is not similar to representations of shoes, their computed attention score will be very low. Therefore, the expression of the user's long-term interest focuses primarily on features of dresses rather than features of shoes, as shown in Fig. 1. Hence, the long-term interest representation is less similar to the label representation and cannot make accurate recommendations, which will reduce the recommender system's performance.

In summary, a user's multiple interests may be distributed in different positions of the session rather than a single interest in a session. The representation of long-term interest obtained using short-term interest as the query vector is one-sided [13] because it is considerably similar to the short-term interest. Thus, the model tends to make recommendations based on a single interest. However, the user's next interaction may be influenced by his multiple interests instead of a single interest in the current session.

Compared with recommendation methods, such as collaborative filtering [14–17], the session-based recommendation cannot obtain the complete history of user interactions. Moreover, the model suffers from data sparsity due to the relatively short length of the session sequence [18]. Current research [5, 6] focuses primarily on user-item interactions observed during sessions and does not make full use of the information that was not interacted with. Self-supervised learning (SSL) [19–21] uses negative samples to exploit unobserved interactions, which can mitigate the effect of data sparsity on recommendations. However, for session-based recommendations, it is difficult to construct reasonable and effective positive and negative samples due to the lack of item attributes and user information. As a result, SSL cannot exert its advantages in session-based recommendations.

The following is a summary of the two challenges:

1) How to choose a suitable query vector to mine multiple user interests that may be overlooked and how to obtain a more comprehensive representation of long-term interest?
2) How can a session-based recommendation be combined with SSL to alleviate the issue of data sparsity?

To address the first challenge, we propose a denoising autoencoder to extract the query vector. It takes the item representations from the session as input and returns the effectively hidden layer vector as the query. This query contains the user interest's multiple features, allowing for a more comprehensive long-term interest. To address the second challenge, we propose an SSL framework, which includes a new method for dividing positive and negative samples and an adaptive threshold for mining negative hard samples to improve training efficiency. The following are the main contributions of this paper:

- We present a query extraction module based on a denoising autoencoder that can aid in the discovery of a

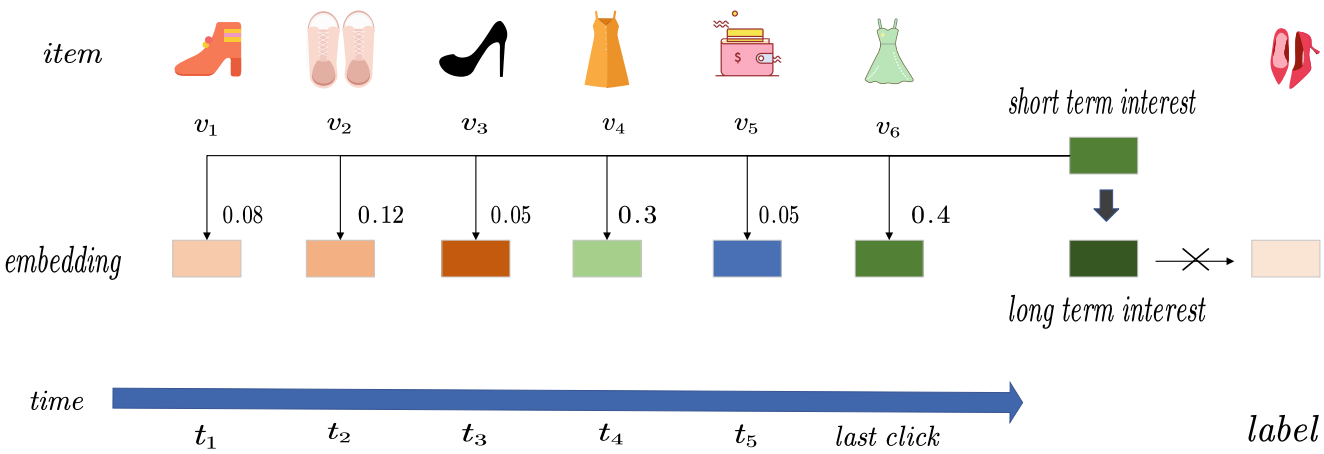


Fig. 1 Limitation of short-term interest as the query to obtain long-term interest

user's multiple interests. To the best of our knowledge, this is the first time a denoising autoencoder has been combined with a GNN and separate short-term interest and query to handle in the model for session-based recommendations.
- We propose a preference generation module to generate the user's final preference. It combines query and attention mechanisms to obtain a more comprehensive long-term interest. Unlike previous work, in addition to adding long and short-term interests, we add information from the query to obtain the user's final preference and explain the reasons.
- We propose an SSL framework to mitigate the impact of data sparsity on the recommendation, which includes a more reasonable method of dividing positive and negative samples, and mining negative hard samples greatly improves training efficiency compared with the state-of-the-art representative models.

We ran extensive experiments on four datasets, and the results show that our proposed DAS-GNN outperforms the state-of-the-art models, validating the effectiveness of the proposed model.

The following are the remaining chapters of this article. In Section 2, we list the work on the session-based recommendation. Section3 lists the main symbols and problem definition. Section 4 provides a thorough introduction to the DAS-GNN. Section 5 poses five questions and uses experiments to answer them. Finally, Section 6 discusses additional application scenarios and the model's drawbacks, then concludes this study and discusses future research directions.

2 Related works

The session-based recommendation has the advantages of privacy and applicability, and it has found widespread use in e-commerce and other fields. With the rise of GNN, some studies make GNN-based recommendations for users. In Section 2.1, we first introduce the application of GNN in the recommendation. Sections 2.2 and 2.3 describe how autoencoders and SSL are used in recommendations, respectively.

2.1 Graph neural network for recommendations

GNN has recently become popular in recommender systems. SR-GNN [6] suggests converting the session into a graph structure, using a gated graph neural network to optimize node representation, and using the user's most recent interaction in the session as a query vector to determine his long-term interest. FGNN [5] proposed considering the sequence order in the session and its potential order relationship, but its long-term interest is still extracted based on the SR-GNN method. When a session is encoded as a graph structure, LESSR [11] improved the encoding method to solve the problem of information leakage, that is, the many-to-one problem between the session and the session graph. However, when it comes to the extraction of long-term interest, the attention mechanism still selects short-term interest as the query. HA-GNN [22] considers the high-order relationship of nonadjacent items in the session, and the short-term interest is used as a query to extract long-term interest. The aforementioned studies all focus on improving the model framework and GNN structure while ignoring the limitations in extracting long-term interest.

2.2 Autoencoder for recommendations

The autoencoder, as a neural network model, can effectively mine features of the input data. AutoRec [23] is the first to introduce an autoencoder into collaborative filtering recommendation, taking the rows and columns of the user-item interaction matrix as input to obtain the representation of users and items, respectively. Li et al. [24] proposed using a marginalized denoising autoencoder to extract additional information about users and items while making full use of the side information during recommendation. CDAE [25] employs a denoising autoencoder for user top-N recommendation tasks. The addition of Gaussian noise improves the model's robustness, and an additional user node is added to the denoising autoencoder to enrich the model's semantic information. VAE-CF [26] improves the prediction accuracy using the variational autoencoder for collaborative filtering. It modifies the regular parameters of the standard variational autoencoder and replaces the Gaussian distribution with a multivariate distribution.

The autoencoder is usually used as a whole predicting model in previous research. In such cases, it is challenging for an autoencoder model to mine complex transition relationships between items because of its simple structure. Furthermore, some of the above autoencoder models are combined with classical collaborative filtering algorithms for recommendations. The aforementioned work did not perform well because of the autoencoder's inability to mine higher-order information like GNN-based models. The autoencoder is included as a building block of the model DAS-GNN, which alleviates the problem of multiple user interests when combined in a GNN-based model.

2.3 Self-supervised learning for recommendations

SSL is a type of unsupervised learning widely used in NLP [27] and CV [19, 20]. Positive samples of a picture can be created using conventional image data expansion methods, such as cropping and rotation, whereas negative samples are obtained by sampling the other original images randomly.

Optimization learning can be performed without labels by employing loss functions like infoNCE loss [28]. S3-rec [29] was the first to incorporate SSL into the recommendation for sequential recommendation pre-training. SGL [30] optimizes node representation by maximizing the similarity of the same node in different views using SSL for collaborative filtering. DHCN [10] introduces SSL in the session-based recommendation, which combines the features of hypergraph and line graph perspectives. The negative samples of DHCN, however, are obtained by shuffling, that is, by disrupting the internal order of samples. The highly random approach may result in several erroneous negative samples. Therefore, we propose a new SSL framework that can solve the problem of high randomness of negative samples and can alleviate the data sparsity for recommendations.

3 Preliminary

Problem statement Given a set of sessions $S = \{s^{(1)}, \ldots, s^{(t)}\}$, covering the set of items $I = \{v_1, \ldots, v_n\}$. A session $s \in S$ is a sequence of items $s = [v_1, v_2, \ldots, v_k]$, in which k represents the length of the session. As the length of a session sequences is not fixed, k is not constant. $v_j \in I$ represents the j-th interactive item in s. The goal of session-based recommendation is to predict items that the user will interact with next. Formally, given a session s as an input, the model will predict the user's next interactive item $v_k + 1$ by learning the item transition relationships in s.

The main symbols and their descriptions in this paper are shown in Table 1.

4 Proposed model

This chapter first covers converting from a session to a directed weighted graph and some initialization settings for node representation. Second, it shows how GNN modules encode item representations. Third, for the first challenge, we propose a query extraction module. For the second challenge, we propose a novel SSL framework with hard samples. Finally, we extract the user's final recommendation preference. Figure 2 depicts the overall framework of the DAS-GNN model.

4.1 Session to graph and embedding

Session to graph For a given session $s = \{v_1, v_2, \ldots v_k\}$, we create a directed weighted graph $G = (\nu, \varepsilon)$. In the session graph, we consider the item $v_i \in \nu$ as a node in G, and regard the behavior of a user clicking item $v_i - 1$ first and clicking the item v_i next as an edge $(v_{i-1}, v_i) \in \varepsilon$. If (v_{i-1}, v_i) appears more than once, the weight corresponding to this edge is increased by one each time it appears. Figure 3 depicts a toy example of the construction process in a session graph.

Embedding Each v_i in $I = \{v_1, v_2, \ldots, v_n\} \in \mathbb{R}^{n\times 1}$ is a scalar, and it is randomly initialized to $\mathbf{v}_i \in \mathbb{R}^d$ by embedding, as done by most GNN-based methods. Accordingly, the embedding dimension of item set is $\mathbb{R}^{n\times d}$. To facilitate the GNN module description, we also denote the representation of the item node i at layer 0 of GNN as $h_i^{(0)} = \mathbf{v}_i \in \mathbb{R}^d$.

4.2 Graph neural network module

We use GNN to optimize the node representations after obtaining their randomly initialized embeddings. Because RNN can only obtain useful information from the forward and reverse directions, the gradient vanishing problem emerges gradually as the sequence length increases [31]. As GNN can aggregate information from multiple neighbors to obtain higher-order features, we explain the GNN forward propagation from the standpoint of message passing.

The GNN message-passing process is divided into two stages, which are message aggregation and message update.

Table 1 Main symbols and descriptions in the paper

Symbol	Description	Symbol	Description
I	Set of all items	v_i	An item in I
s	Single session	S	Collection of all sessions
G	A directed weighted graph composed of a session	ν	Set of all nodes in G
ε	Set of all edges in G	W	Weight Matrix
‡	Query vector	η	User's final preference
n	Number of items in I	d	Item embedding dimension
m_{ij}	Edge weight between node i and j	λ	Number of negative samples
θ	User's long-term interest	$h_{last}^{(L)}$	User's short-term interest
$h_i^{(l)}$	Representation of node i at layer l of the GNN module	$\hat{y}$	Probability of predicted labels

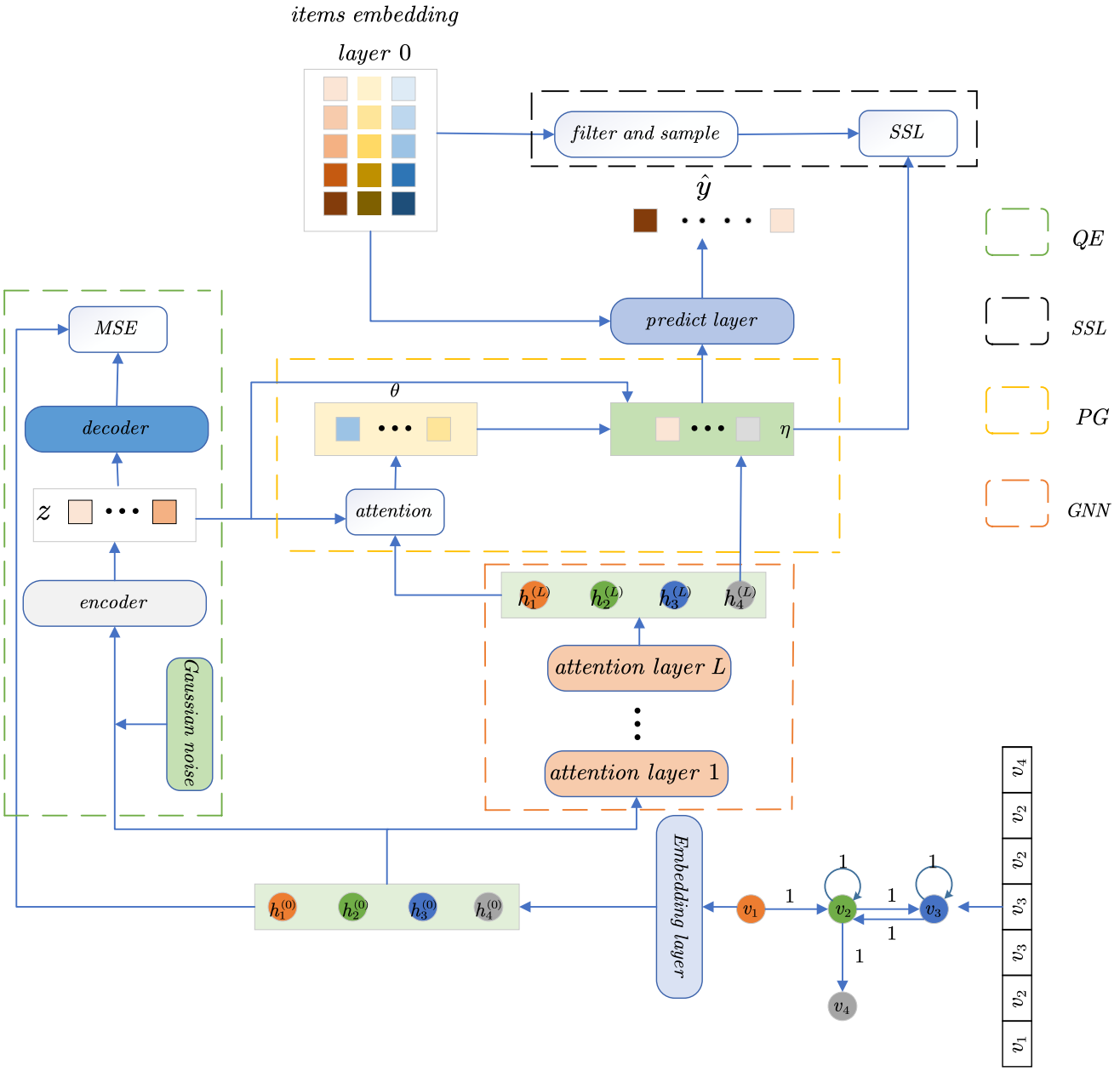


Fig. 2 The green dotted box represents the query extraction module (QE), which is used to extract the query vector in the DAS-GNN model. The SSL module (SSL), denoted by the black dashed box, helps alleviate the problem of data sparsity. The yellow dotted box represents the preference generation module (PG), which is used to generate the user's final preference. The graph neural network (GNN) module is indicated by the orange dotted box

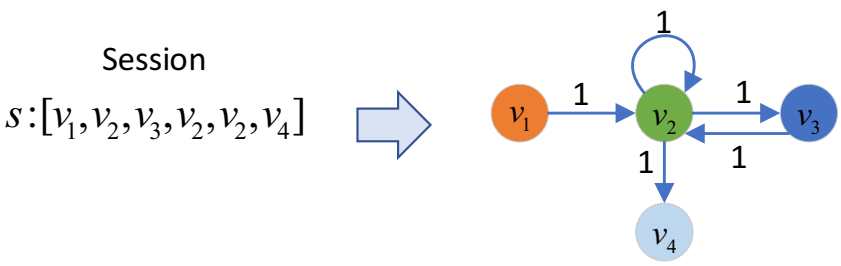


Fig. 3 A directed weighted graph composed of the session s, the weight of the edge is 1, and the items v_1, $v_2v_3v_4$ of session s constitute nodes in the graph

$$\beta_{ij}^{(l)} = \sum\left(K^{(l)}h_i^{(l)} \odot Q^{(l)}h_j^{(l)} + b^{(l)}\right)m_{ij}, j \in N(i) \quad (1)$$

$$\alpha_i^{(l)} = softmax\left(\beta_i^{(l)}\right) \quad (2)$$

$$h_i^{(l+1)} = \sum_{j \in N(i)} \alpha_{ij}^{(l)} V^{(l)} h_j^{(l)} \quad (3)$$

The message aggregation stage is represented by Formulas (1) and (2), which use an attention mechanism [32] to extract neighbor information. $h_i^{(l)}, h_j^{(l)} \in \mathbb{R}^{d\times 1}$ denote the representations of nodes i, j at the layer l in GNN, respectively. $K^{(l)}$, $Q^{(l)}$, $V^{(l)} \in \mathbb{R}^{d\times d}$ are learnable parameters, and the representations of nodes i, j are mapped to the same space via $Q^{(l)}, K^{(l)}$. The related score between the i, j, denoted as $\beta_{ij}^{(l)}$ is obtained by element-wise product and the bias $b^{(l)} \in \mathbb{R}^{d\times 1}$. Based on the attention mechanism, we add the edge weight of the session graph to enrich the target node's feature, where the scalar m_{ij} represents the edge weight between nodes i and j, $N(i)$ represents the set of nodes connected to the node i. Formula (2) normalizes the related scores to obtain the attention scores of the target node and its neighbors. The message update stage is depicted in Formula (3), where the adjacent nodes are multiplied by the corresponding attention scores and then superimposed to obtain the representation $h_i^{(l+1)} \in \mathbb{R}^{d\times 1}$ of the node i at the layer $l + 1$.

4.3 Query extraction module based on autoencoder

This section focuses on the first challenge mentioned in the introduction: how to choose a suitable query to mine various multiple user interests that may be overlooked. Sedhain et al. [23] proposed that an autoencoder-based recommendation model has the advantages of compactness and efficiency. We propose using a denoising autoencoder to extract the query vector and using its hidden layer vector as the query vector in the attention mechanism. First, the query obtained by the autoencoder-based approach covers a broader range of data through encoding the session compared with the use of short-term interest (the representation of the last interactive item in a session) as the query. Therefore, it allows the query to include a variety of characteristics of interest. Second, the user's interests indicate his goals or demands in the session. They often occur intermittently or continuously throughout the session and represent the majority of the features in the session. As a result, if the query vector can restore the session information as much as possible by the decoder, it contains the multiple user interests in a session. Meanwhile, we use Gaussian noise to improve the autoencoder's robustness, and the parameter sizes in the denoising autoencoder are obtained by optuna and grid search.

$$z = encoder\left(h_{flatten}^{(0)} + noise_G \middle| W_1^e, b_1^e, \ldots W_3^e, b_3^e\right) \quad (4)$$

$$\widehat{h} = decoder\left(z \middle| W_1^d, b_1^d, \ldots, W_3^d, b_3^d\right) \quad (5)$$

$noise_G$ denotes Gaussian noise. $z \in \mathbb{R}^{d\times 1}$ represents the hidden layer vector, that is, the query vector used to extract long-term interest.

$W_1^e \in \mathbb{R}^{128\times 480}$, $b_1^e \in \mathbb{R}^{128\times 1}$, $W_2^e \in \mathbb{R}^{64\times 128}$, $b_2^e \in \mathbb{R}^{64\times 1}$, $W_3^e \in \mathbb{R}^{32\times 64}$, $b_3^e \in \mathbb{R}^{32\times 1}$ are the learnable parameters of the encoder part. Because of the varying length of sessions, represented as m, we tried and compared the effect of the parameter, and finally, set it to $m = 15$. Zero padding is used for sessions with $m < 15$, and the last 15 items are kept if $m \geq 15$. $h_{flatten}^{(0)} \in \mathbb{R}^{(d*m)\times 1}$ is the vector that flattens the representation of the m items in the 0th layer as the encoder's input.

$W_1^d \in \mathbb{R}^{64\times 32}$, $b_1^d \in \mathbb{R}^{64\times 1}$, $W_2^d \in \mathbb{R}^{128\times 64}$, $b_2^d \in \mathbb{R}^{128\times 1}$, $W_3^d \in \mathbb{R}^{480\times 128}$, $b_3^d \in \mathbb{R}^{480\times 1}$ are the learnable parameters of the decoder part. Both the encoder and decoder in Formulas (4) and (5) represent the encoders and decoders, respectively, composed of fully connected networks. The encoder has a three-layered fully connected network, with 480, 128, and 64 neurons. While the decoder has a three-layered fully connected network, with 64, 128, and 480 neurons. $\widehat{h} \in \mathbb{R}^{(d*m)} \times 1$ represents the decoder output, which should be as close to $h_{flatten}^{(0)}$ as possible, resulting in the query vector ‡ containing the multiple user interests in the session. MSE is the loss function used by the denoising autoencoder (Fig. 4).

$$loss_2 = MSE\left(\widehat{h}, h_{flatten}^{(0)}\right) \quad (6)$$

4.4 Preference generation module

We use the attention mechanism to assign different importance to interests and fuse them to get the user's long-term interest θ, based on the query vector z, as shown in Formula (7). Then, combine the short-term interest $h_{last}^{(L)}$ with the query vector z and the user's long-term interest θ to obtain the user's final preference η, as shown in Formula (8).

$$\theta = \sum_{i\in\mathcal{V}} soft\max\left(q^T\left(W_1 h_i^{(L)} + W_2 z + b_1\right)\right)h_i^{(L)} \quad (7)$$

$W_1, W_2 \in \mathbb{R}^{d\times d}$, $b_1, b_2 \in \mathbb{R}^{d\times 1}$, and $q \in \mathbb{R}^{d\times 1}$ are the learnable parameters, $h_i^{(L)}$ is the representation of a node i in the layer L, and ν represents the set of items in the current session graph. In Formula (7), the attention scores are computed by z and $h_i^{(L)}$, and the user's long-term interest θ is obtained by multiplying the attention score with the representation of nodes in $\mathcal{V}$. In comparison to the previous methods, θ includes one or more user interests, which is primarily due to the query z

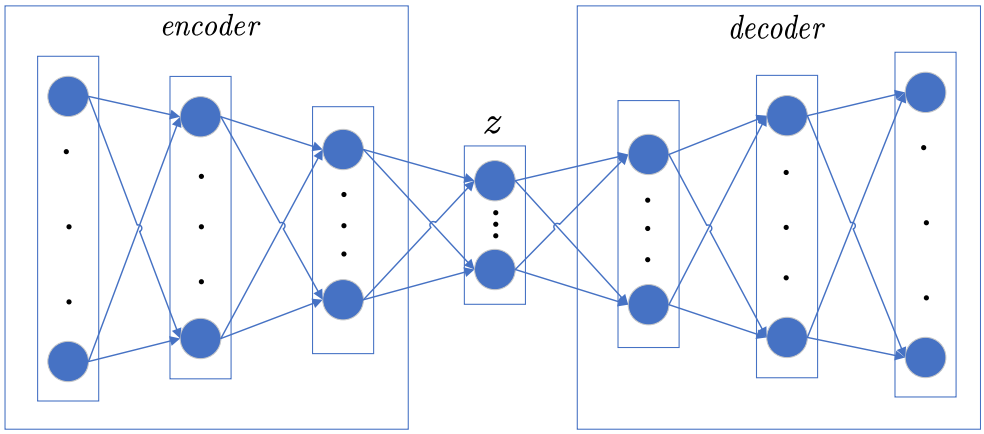


Fig. 4 The Overview of the autoencoder

obtained by the denoising autoencoder. z can assist in discovering different interests for it contains various interest features.

Since previous studies [6, 11] showed the importance of explicitly considering users' short-term interests, by the combination of the short-term interest $h_{last}^{(L)}$ with the query vector z and the user's long-term interest θ, we can obtain the user's final preference η, as shown in Formula (8).

$$\eta = W_3\left(z \| \theta \| h_{last}^{(L)}\right) + b_2 \tag{8}$$

$W_3 \in \mathbb{R}^{d \times 3d}$, $b_1, b_2 \in \mathbb{R}^{d \times 1}$, and $\|$ represents the concatenation operation. In Formula (8), the user's final preference η is obtained after the concatenation of z, θ, and $h_{last}^{(L)}$ fed into a fully connected network layer. Because the query vector z contains interest feature information, it is incorporated in Formula (8) to obtain an enhanced expression of the user's final preference. This method is similar to previous approaches for generating the user's final preference, except that previous works used one variable to represent both short-term preference and query vector [6].

4.5 Self-supervised learning module

4.5.1 Loss and method to create hard samples

We propose a new SSL framework in this section to address the second challenge: how can session-based recommendations be integrated with SSL to alleviate the problem of data sparsity? Yang et al. [18] proposed that negative sampling could be performed through unobserved data. Yao et al. [33] masked attribute features of the same and different items to obtain positive and negative samples, respectively. In contrast to Zhou et al. [29] and Yao et al. [34], it is difficult to define positive and negative samples in session-based recommendations because it lacks information about the user and item attributes.

To address the aforementioned issue, we propose a new SSL framework based on session-based recommendations. In SSL, we design the method of dividing positive and negative samples and the margin for mining negative hard samples as well as optimize SSL based on the loss function. We define $s_n = \{i|\ i \in I,\ i \notin s\}$ as the set of negative samples of the session s because items in different sessions have a high probability of being dissimilar. The number of s_n is enormous, however, after model training, most negative samples have been identified to be simple samples, and the gradients generated from them are of little benefit to the model [34, 35]. According to [34], only a small portion of negative sample representations are close to the user's final preference η, which is misjudged by the model and belongs to hard samples. The embeddings of hard samples are extremely similar to that of the user's final preferences. However, they are not labels. For example, learning the pictures of a cat and a tiger will result in very similar representations, but they do not belong to the same species. Providing more hard samples to the model allows it to learn more fine-grained information for the prediction or classification task. Gao et al. [17] suggested that hard samples contained more valuable information, which could help the model to obtain enhanced item representations. Based on the aforementioned description, we designed the method to create hard samples.

First, we sample from s_n to get the set s_{sample} and use $|s_{sample}| = \lambda$ to denote its size. λ is a manually tuned hyperparameter. Then, to speed up model training, we propose using a learnable parameter margin γ for filtering simple samples. After the filter of simple samples, the size of s_{sample} is $|s_{sample} - \gamma|$.

$$\gamma = \sigma(W_3\eta + b_3) \tag{9}$$

$$loss_3 = \frac{1}{|s_{sample} - \gamma|} \sum_{i \in s_{sample}} \sigma\left(\eta^T \mathbf{v}_i\right), \sigma\left(\eta^T \mathbf{v}_i\right) > \gamma \tag{10}$$

where $W_3 \in \mathbb{R}^{1 \times d}$, $b_3 \in \mathbb{R}^{1 \times 1}$ are learnable parameters, σ is the sigmoid function, and η is the user's final preference in session s. In Formula (9), the margin γ is calculated through a fully connected network with one layer and $\gamma \in [0, 1]$, which avoids the overly tedious manual tuning of parameters. In Formula (10), we calculate the relevant score between η and hard samples and take the average value as $loss_3$, which $\mathbf{v}_i$ represents the initial embedding of node i.

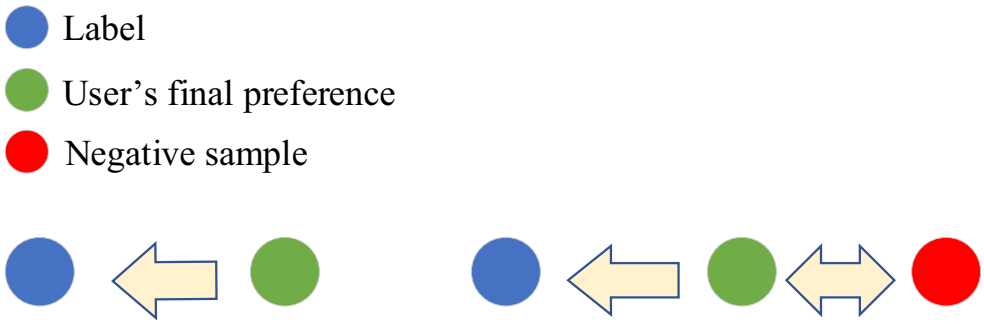


Fig. 5 Comparison of models with and without negative samples in the metric space

4.5.2 Rationale for using negative samples to alleviate data sparsity

Figure 5 illustrates why the inclusion of negative samples can alleviate the problem of data sparsity.

As shown in Fig. 5, the user's final preference, represented as the green circular, is the vector to be learned and optimized. The label, represented as the blue circular, provides a target for the user's final preference to be approximated, whereas the negative sample, represented as the red circular, provides a target that is far away from the final preference. In the left side of Fig. 5, when available data is sparse and there is no negative sample, the preference has only one gradient from the label. Fewer statistics on items and users result in fewer optimization gradients. On the right side of the figure, when the data is sparse, adding negative samples has two advantages to alleviate the issue of data sparsity. First, it can add additional gradient information for optimization. Second, it can help the model learn fine-grained information, especially when it belongs to hard samples. In a nutshell, the labels provide a target to approach, whereas the negative samples provide a target to depart from. More gradients help in better localization of preference than the case on the left side of Fig. 5. Therefore, our proposed SSL module can alleviate the problem of data sparsity.

4.5.3 Comparison with DHCN of the sampling method

In Fig. 6, we explain why our proposed sampling approach will not result in the problem of excessive randomness in DHCN.

Figure 6 illustrates the comparison of different sampling methods between DHCN and our proposed DAS-GNN. Each batch in DHCN comprises multiple sessions as shown at the top of Fig. 6, and the vector in each row represents the user's final preference for each session. Each row in the negative sample matrix is a negative sample of the user's final

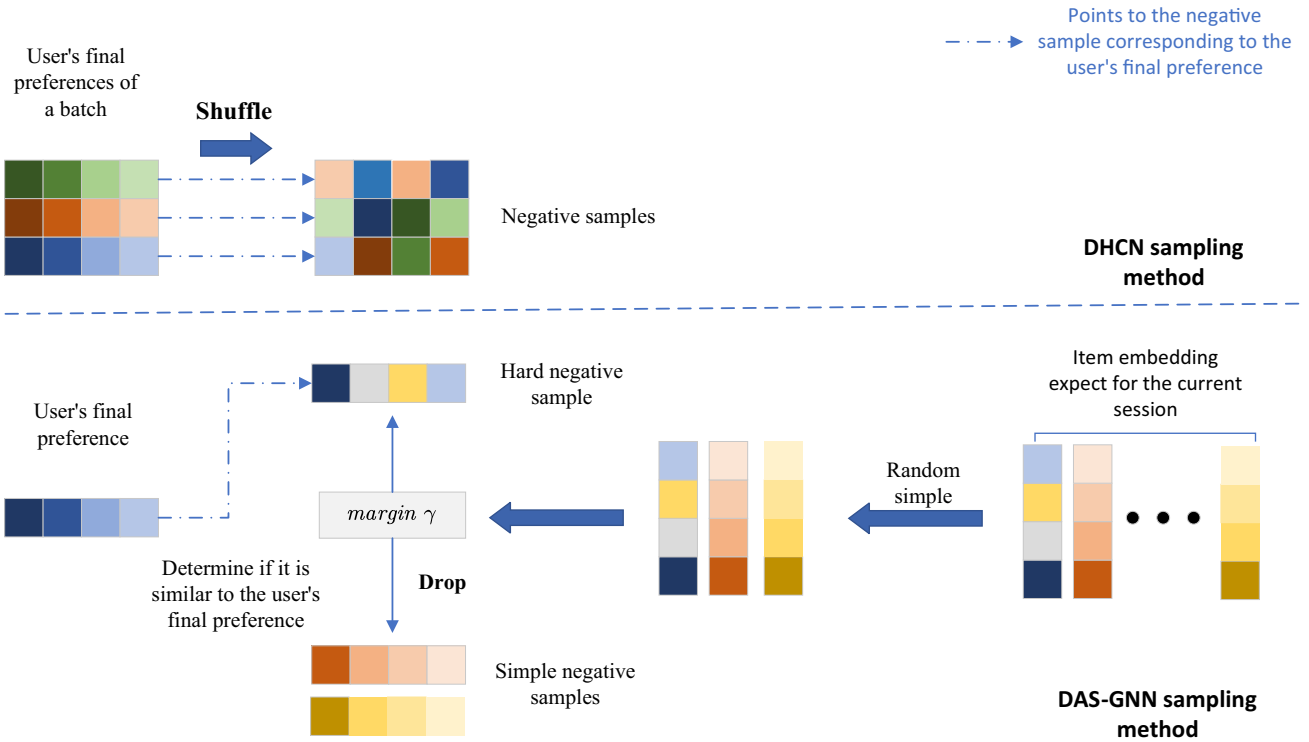


Fig. 6 Comparison of the DHCN sampling approach and our proposed sampling approach

Table 2 Dataset statistics

Dataset	Diginetica	Gowalla	RetailRocket	Nowplaying
Sessions	777,029	830,893	448,775	915,128
Items	42,596	29,510	36,968	60,417
Average lengths	4.8	3.85	5.43	7.42

preference for the corresponding row in the batch. The color in each row indicates the similarity between the vectors. DHCN shuffles the matrix to get negative samples. However, owing to the high randomness of the shuffling, most negative samples obtained will be simple samples. From the sample matrix in Fig. 6, barring the vector in the third row, others belong to simple negative samples which contribute little to learn the user's final preference.

The bottom part of Fig. 6 shows how our proposed SSL framework works. The negative samples are obtained by sampling all remaining items outside the current session. The similarity between the user's final preference and the negative samples is calculated, and if the similarity is lower than the margin (the threshold γ), it will be discarded. Thus, we can find $|s_{sample} - \gamma|$ hard negative samples to obtain fine-grained information. The new SSL framework has two advantages over the DHCN sampling approach: 1) The samples are obtained from the original data rather than constructed and 2) the use of margin γ can help the model learn more fine-grained information when finding hard negative samples.

4.6 Prediction layer

Based on the preceding steps, the prediction score for the candidate item $i \in I$ is calculated using the inner product between η and v_i, and a top-N recommendation list is produced for the target users.

$$\widehat{y} = soft\max\left(\eta^T \mathbf{v}_i\right) \tag{11}$$

We use cross-entropy as the loss function to optimize the model.

$$loss_1 = \sum_{i=1}^{N} y_i \log\left(\widehat{y}_i\right) + (1-y_i)\log\left(1-\widehat{y}_i\right) \tag{12}$$

The final loss of the model is obtained by adding $loss_1$ in Formula (12), $loss_2$ in Formula (6) and $loss_3$ in Formula (10).

$$loss = loss_1 + \alpha_1 loss_2 + \alpha_2 loss_3 \tag{13}$$

α_1, α_2 are hyperparameters used to adjust the proportion of the two auxiliary losses. The adam optimizer is used to optimize the total loss, and the parameters are tuned by calculating the gradients of the *loss*.

5 Experiments

We conducted extensive experiments on four datasets to validate DAS-GNN effectiveness and attempt to answer the following questions:

- **RQ1** How does the DAS-GNN model perform in comparison with other cutting-edge models?
- **RQ2** How do the hyperparameters α_1, α_2, λ, and L affect the model's performance?
- **RQ3** How does each module affect the model's performance?
- **RQ4** What effect does our proposed new SSL framework have on item representation?
- **RQ5** What effect does the query extraction module have when multiple interests exist in a session?

5.1 Datasets and metrics

The proposed DAS-GNN was tested on four datasets, that is Diginetica,[1] Gowalla,[2] Nowplaying,[3] and RetailRocket.[4] These are widely used in session-based recommendation tasks. We deleted the sessions of length one from the dataset as done in [6, 36–38] and filtered out items with frequency < 5. The data from the most recent week is divided into the test set, and the rest is used as the training set. Moreover, we used the same data augmentation techniques described in [10, 11, 37]. Table 2 shows different statistics for the four datasets.

Following Wang et al. [8, 39], we use HR@20 and MRR@20 as evaluation metrics to measure the model's performance. HR measures whether an item of interest to the user is on the recommended list and focuses on precision. Like the MAP metric in some recommendation models [40, 41], MRR emphasizes the relative order of items, that is, whether the item the user requires is ranked at or near the top.

[1] http://cikm2016.cs.iupui.edu/cikm-cup
[2] https://snap.stanford.edu/data/loc-gowalla.html
[3] http://dbis-nowplaying.uibk.ac.at/#nowplaying
[4] https://www.kaggle.com/retailrocket/ecommerce-dataset

5.2 Performance comparison (RQ1)

5.2.1 Baselines

We compare the DAS-GNN model with the following representative models:

- Item-KNN [42]: It recommends items that are similar to items in the session, and the similarity between items is calculated using cosine similarity.
- FPMC [43]: It employs Markov chains for sequence recommendation. Following [10], we do not introduce the user's latent vectors when calculating the recommendation scores.
- GRU4REC [44]: It models the session sequence using RNN to capture the user's preferences, and optimizes the model using a ranking-based loss function.
- NARM [45]: It includes an RNN-based attention mechanism to capture the user's main purpose and sequential behavior.
- SR-GNN [6]: It uses a gated graph convolution to obtain item representations and combines attention to determine the user's long-term and short-term interests. Finally, the interests are combined to get a graph representation vector.
- FGNN [5]: It transforms a session into a directed weighted graph and uses GAT to learn item representations.
- LESSR [11]: It focuses on the problem of information leakage when converting a session into a graph structure, and proposes ways to improve the transition of the session to address the issue.
- SHARE [46]: It makes recommendations for users by introducing global session information and modeling it as a hypergraph.
- DHCN [10]: It constructs two types of hypergraphs to learn intra-session and inter-session information, and uses SSL to enhance their connection.

For the aforementioned models, we used grid search to find the optimal hyperparameters and used 20% of the training set as the validation set to adjust the hyperparameters [12, 47], and used the adam optimizer to optimize the parameters. Table 3 shows the performance of the baselines with optimal hyperparameters.

5.2.2 Comparison with baselines

To demonstrate the effectiveness of DAS-GNN, we compare it with the aforementioned state-of-the-art models, and Table 3 shows all the comparison results.

Table 3 shows that the traditional machine learning models like Item-KNN and FPMC perform the worst because they make recommendations based on item similarity or item transition relationships. Sequence-based models GRU4REC and NARM, which take sequence information into account, perform better than Item-KNN and FPMC. NARM outperforms GRU4REC in terms of performance because it employs an RNN-based attention mechanism to distinguish the importance of different items, demonstrating the importance of attention mechanisms in the recommendation.

The performance of GNN-based models, such as SR-GNN and FGNN, outperforms all RNN-based models, which demonstrates the superiority of GNN when making recommendations. It can learn higher-order relationships by converting the session sequence into a graph structure, while the sequence-based model only extracts neighbor features in two directions. Compared to SR-GNN, the effect of FGNN is still slightly worse although it uses additional edge weights, demonstrating

Table 3 Results of DAS-GNN with state-of-the-art models. All figures in the table are percentage numbers with "%" omitted. The underlined figures are the second-best results. The bold figures indicate the best results

Method	Diginetica		Gowalla		Nowplaying		RetailRoket	
	HR@20	MRR@20	HR@20	MRR@20	HR@20	MRR@20	HR@20	MRR@20
Item-KNN	39.51	11.22	38.60	16.66	15.94	4.91	–	–
FPMC	28.50	7.67	29.91	11.45	7.36	2.82	32.37	13.82
GRU4REC	29.45	8.33	–	–	7.92	4.48	44.01	23.67
NARM	49.80	16.57	50.07	23.92	18.59	6.93	50.22	24.59
SR-GNN	50.81	17.31	50.32	24.25	18.87	7.47	50.32	26.57
FGNN	50.03	17.01	50.06	24.12	18.78	7.15	–	–
LESSR	51.70	18.15	51.34	25.49	<u>20.35</u>	8.18	53.41	<u>28.03</u>
SHARE	51.48	17.35	<u>51.8</u>	<u>25.79</u>	19.7	<u>8.28</u>	<u>53.43</u>	26.9
DHCN	<u>53.18</u>	<u>18.44</u>	–	–	20.03	8.18	52.66	27.30
DAS-GNN	**53.51**$_{\pm 0.08}$	**18.86**$_{\pm 0.03}$	**52.73**$_{\pm 0.04}$	**25.96**$_{\pm 0.01}$	**20.56**$_{\pm 0.03}$	**8.42**$_{\pm 0.10}$	**53.47**$_{\pm 0.07}$	**28.56**$_{\pm 0.04}$

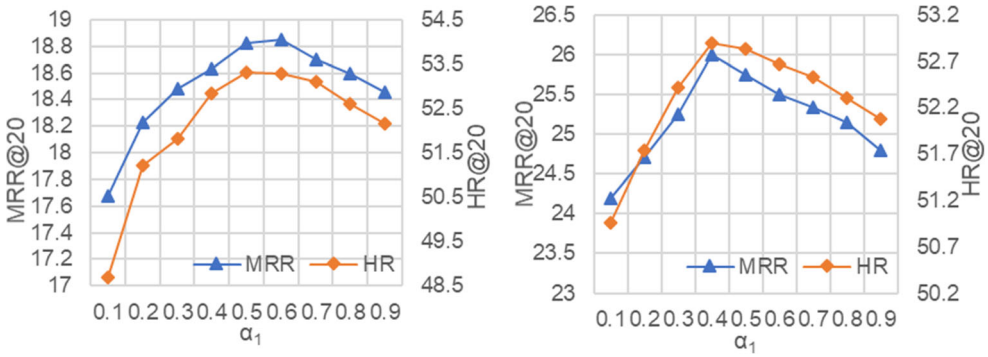


Fig. 7 Influence of hyperparameter α_1 on the performance of DAS-GNN on the Diginetica (left) and Gowalla (right) datasets

the importance of the graph convolution method [11]. LESSR considers the problem of information leakage when converting a session into a graph, which further improves the recommendation's performance. SHARE views all sessions as a hypergraph and incorporates global data, which alleviates the problem of data sparsity. DHCN, based on introducing global session information, mines feature from the perspectives of item relationship and session relationship and employs SSL to integrate representations from the two perspectives, achieving good performance.

Our proposed DAS-GNN outperforms all the baselines on four datasets. First, DAS-GNN can capture a user's long-term interest more accurately and comprehensively than other GNN-based models because of the design and fusion of the query extraction module into the GNN-based model. Second, it employs SSL to address the data sparsity problem in the session-based recommendations, which introduces a threshold to filter simple samples and significantly improves the model's efficiency. Although DHCN incorporates SSL to integrate the representation from both perspectives, it performs worse than DAS-GNN because of its shuffling method for negative samples. The majority of the samples obtained by the shuffling are simple samples that do not provide much valid information for the prediction.

5.3 Hyperparameter study (RQ2)

In this section, we study the effect of the hyperparameters α_1, α_2, λ , L on the model. Figures 7, 8, 9, and 10 show the results of metrics MRR@20 and HR@20 on datasets Diginetica and Gowalla, respectively.

As shown in Fig. 7, the MRR@20 increases with the increase of α_1 and reaches its maximum when α_1 takes values of 0.6 and 0.4 on Diginetica and Gowalla, respectively, and it begins to decline with its value continuing to increase. Similarly, HR@20 reaches its best values with α_1 equals 0.5 and 0.4 on the corresponding datasets. Figure 7 demonstrates that DAS-GNN is sensitive to the MSE loss of the denoising autoencoder. The MSE loss, as part of the total loss in Formula (13), has a significant impact on model performance, demonstrating the importance of the query extraction module in the model. The performance of DAS-GNN can be further enhanced by setting the value of α_1 reasonably.

Figure 8 shows the effect of α_2 on DAS-GNN. The value of MRR@20 reaches its peak when α_2 equals 0.5 and 0.4 on Diginetica and Gowalla, respectively. With the continuous increase of α_2, the MRR@20 begins to fluctuate slightly and decline. The trends in HR@20 are similar to MRR@20. The maximum values are obtained in the corresponding dataset when α_2 takes the values of 0.5 and 0.4, respectively. The trend on MRR@20 and HR@20 demonstrates the effectiveness of the SSL framework and it can alleviate the issue of data sparsity to a certain extent through the efficient sampling of negative samples.

On the left side of Fig. 9, as the value of negative samples λ increases, MRR@20 and HR@20 show an upward trend and finally tend to be flat. This demonstrates that it is the most cost-effective for the model when λ equals 4 or 5 on the Diginetica dataset, and too many negative samples will not result in a further improvement. On the right side of Fig. 9, when λ is larger than 4, the values of MRR@20 and HR@20 fluctuate, which decrease and then gradually increase. This

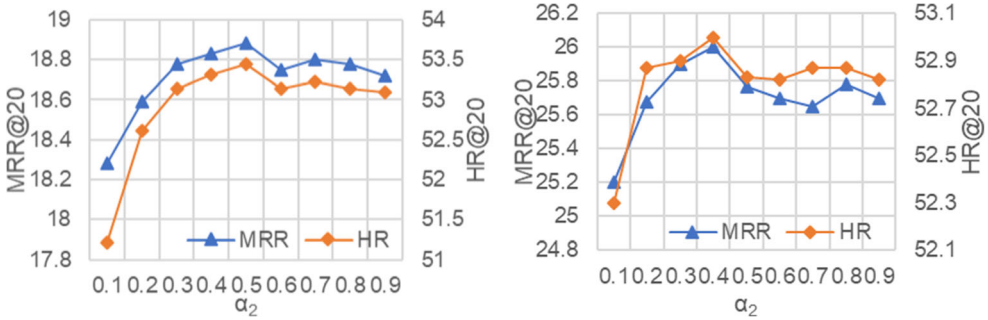


Fig. 8 Influence of hyperparameter α_2 on DAS-GNN on Diginetica (left) and Gowalla (right) datasets, respectively

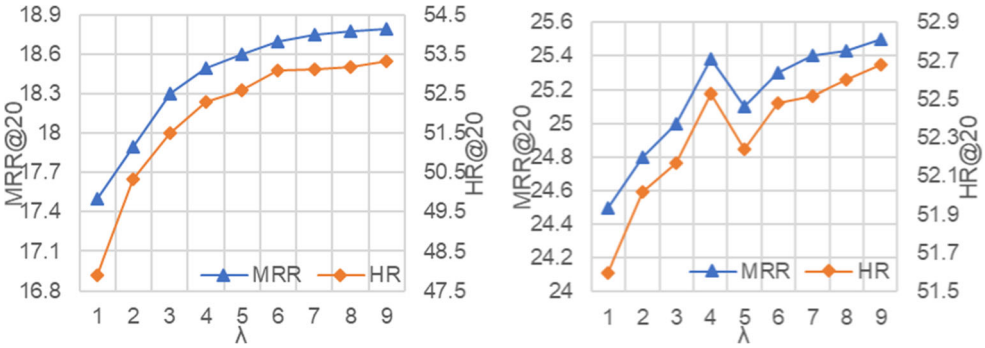


Fig. 9 On Diginetica (left) and Gowalla (right) datasets, the effect of hyperparameter λ on DAS-GNN

phenomenon may be caused by the instability of DAS-GNN training.

Figure 10 shows the influence of layer L on DAS-GNN on Diginetica and Gowalla datasets, respectively. The figure shows that DAS-GNN performs best with L equal four in both datasets. The model shows weak expression ability and poor performance with a one-layered GNN due to underfitting. As the number of layers increases, its MRR@20 and HR@20 gradually improve and the model performs best when $L = 4$. With the value of L larger than 4, the model's performance gradually declines, since a GNN with more layers will result in the problems of overfitting and over-smoothing [48]. In short, Fig. 10 demonstrates that the performance of DAS-GNN is sensitive to parameter L, that is, the number of layers of the GNN module.

5.4 Ablation study (RQ3)

To explore the impact of each module in DAS-GNN, we designed various variants to verify their performance on Diginetica and Gowalla datasets.

1) DAS-GNN-N: It uses short-term interest as the query to extract the users' long-term interest without the query extraction and SSL modules.
2) DAS-GNN-S: It represents a variant that removes the query extraction module and employs only the SSL module.
3) DAS-GNN-E: It employs the query extraction module and the SSL module is removed.

Table 4 shows that DAS-GNN-S and DAS-GNN-E outperform DAS-GNN-N on both datasets which indicates the effectiveness of both the SSL and the query extraction modules. As for the SSL module, it provides a method of dividing positive and negative samples which effectively makes use of unobserved data information and alleviates the data sparsity issue. While for the query extraction module, it depends on an autoencoder to mine a user's multiple interests and improve the recommendation accuracy. DAS-GNN, which employs both the SSL and query extraction modules, performs the best than its variants, indicating the effectiveness of combining the two modules, and demonstrating the DAS-GNN model's superiority.

5.5 Effect of self-supervised learning (RQ4)

5.5.1 Visualization and analysis of item representations

In this section, we will verify the effect of self-supervised learning by further comparison and analysis of DAS-GNN with DAS-GNN-E, as mentioned above, in which the SSL module is removed. The experiments are conducted on the Diginetica dataset. We perform dimensionality reduction via t-SNE and visualize the trained item representations using KDE and scatterplots. The results are demonstrated in Figs. 11, 12, and 13, respectively.

Following the dimensionality reduction via t-SNE, the abscissa and ordinate values in Figs. 11, 12, and 13 are the first and second dimensions of the item representation, respectively. Figure 11 depicts the distribution of item representations

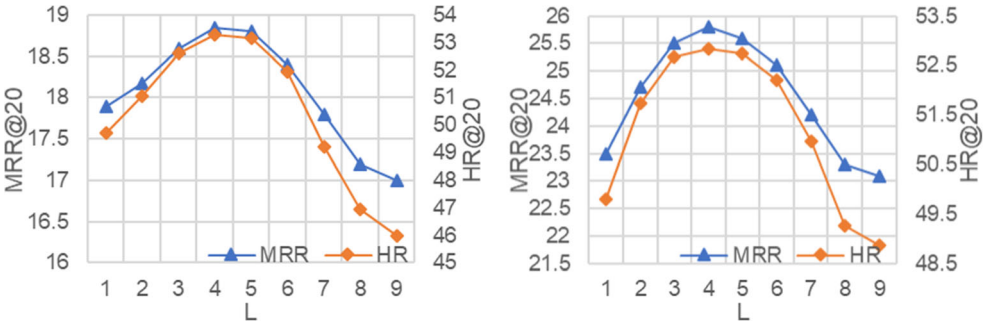


Fig. 10 Influence of layer L on DAS-GNN on Diginetica (left) and Gowalla (right) datasets, respectively

Table 4 Influence of various model variants on recommendation performance. The bold numbers show the best results

Variants	Diginetica		Gowalla	
	MRR@20	HR@20	MRR@20	HR@20
DAS-GNN-N	18.1	50.85	24.49	50.34
DAS-GNN-S	18.46	51.76	24.96	50.79
DAS-GNN-E	18.62	52.63	25.42	51.93
DAS-GNN	**18.87**	**53.52**	**25.96**	**52.72**

using KDE, and we can see that the distribution of item representations of the DAS-GNN-E is more concentrated, whereas the distribution of item representations of the DAS-GNN is wider. It indicates that the ability to distinguish different items in DAS-GNN-E is weaker than that of DAS-GNN. Simultaneously, Fig. 12 shows that the majority of the item representations of the DAS-GNN-E are mixed. The boundaries between clusters are more blurred. In Fig. 13, the clustering effect of DAS-GNN item representations is relatively good. Clusters of items are more compact and relatively well-bounded. This phenomenon is primarily due to the new SSL framework, which can effectively use the information of unobserved items. The new positive and negative sample division method improves the quality of the obtained positive and negative samples, and the mining of hard samples helps the model learn fine-grained information.

5.5.2 Evaluation of item representations "centrality"

To evaluate the "centrality" of the item representations of the different sampling methods, we designed three variants and used Silhouette Coefficient [49] as a metric. The Silhouette Coefficient is a commonly used metric to measure the effect of clustering. Its values range from −1 to 1, with larger values indicating better clustering. Besides DAS-GNN-E, we designed DAS-GNN-D, which employs a variant of DHCN's sampling as its sample method. DAS-GNN means that our proposed SSL sampling approach is used in the model.

Table 5 shows the comparison of variants with different sampling methods on Diginetica and RetailRocket datasets using the Silhouette Coefficient metric. From the table, it can be seen that DAS-GNN-E has the worst performance because it ignores negative sampling when optimizing the model. DAS-GNN-D has improved performance compared with DAS-GNN-E because it uses both the label and a variant of the DHCN's sampling, which shows the effect of negative sampling on the performance. DAS-GNN achieves the best performance since it improves its sampling approach by sampling from other sessions and setting a margin to filter simple samples, which indicates the superiority of the proposed SSL framework.

5.6 Effect of the query extraction module (RQ5)

Explainable AI methods (xAI) [50, 51] study how to understand AI, mentioning that heatmaps can help us better understand AI models. In this section, we use heatmaps to explain the effect of the query extraction module.

We selected a representative session from Diginetica's test set to represent multi-interest scenarios. It has the following item sequence [3687, 135,335, 3687, 13,776, 13,776, 3687, 8513, 3687, 13,271, 3687, 82,380, 3687, 8488, 8518 (label)], with the corresponding categories [291, 291, 291, 186, 186, 291, 291, 291, 186, 291, 186, 291, 291, 186 (label)]. We

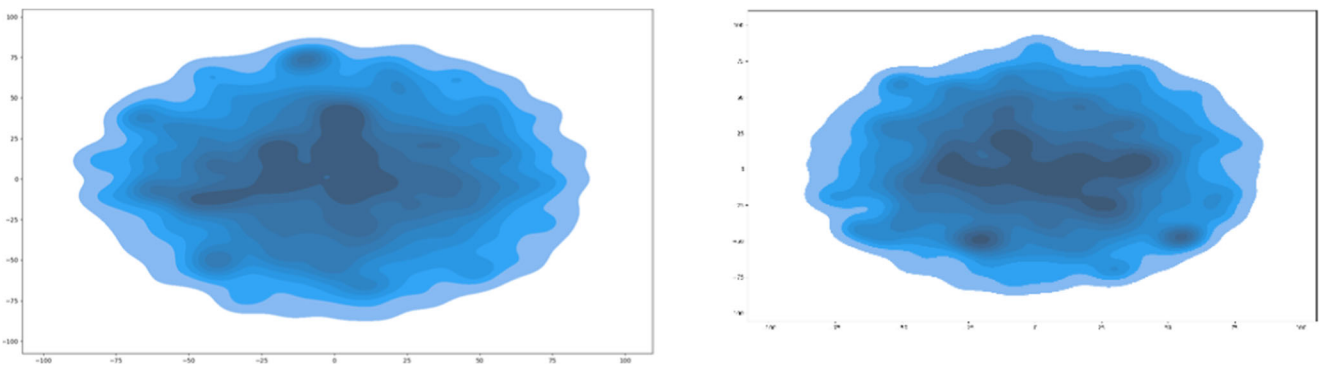


Fig. 11 Visualization of the distribution of item representations in the Diginetica dataset using KDE, with DAS-GNN-E on the left and DAS-GNN on the right

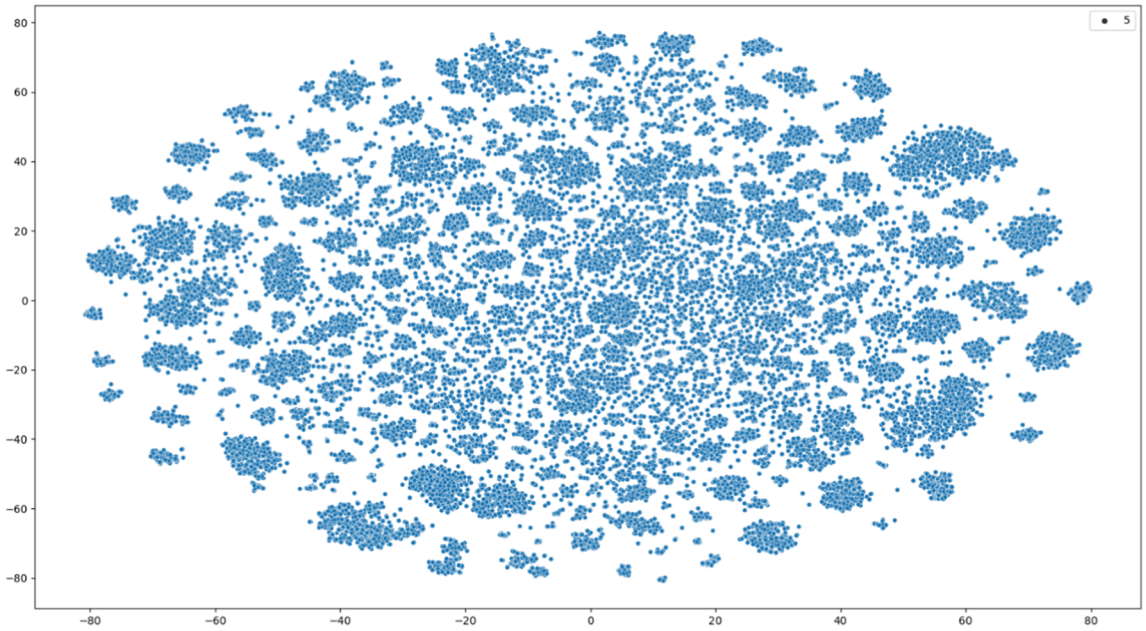


Fig. 12 Visualization of item representations for DAS-GNN-E in the Diginetica dataset using scatterplots

visualize the attention scores between the query vector and the item representations in the session, using the last item in the session and the hidden layer vector of the autoencoder as the query, respectively. The left and right sub-graphs of Fig. 14 show their difference. As shown in Fig. 14, it can be seen that the left sub-graph tends to recommend the last three items of category 291 due to the direct use of the last item vector (category 291) as the query to obtain the attention score. In comparison, the right sub-graph tends to recommend the 11th item (category 291) and the 8th item (category 186) because the hidden layer vector of the autoencoder contains the information of category 291 and category 186. Finally, the query extraction module makes it possible to recommend the 8518 item (category 186) in the end. This example shows that the

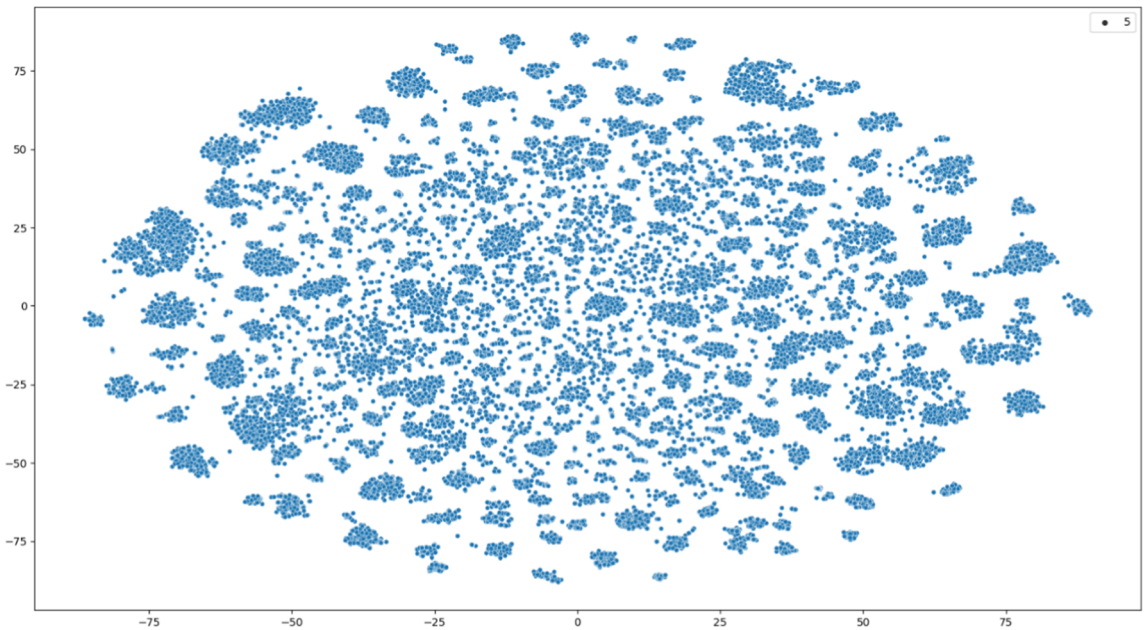


Fig. 13 Visualization of item representations for DAS-GNN in the Diginetica dataset using scatterplots

Table 5 Comparison of variants with different sampling methods under Silhouette Coefficient metric on Diginetica and RetailRocket datasets

Dataset Variants	Silhouette Coefficient (DIGINETICA)	Silhouette Coefficient (RetailRocket)
DAS-GNN-E	0.06999437	0.022977062
DAS-GNN-D	0.07528592	0.03357826
DAS-GNN	0.0915474	0.04807572

query extraction module can effectively solve the multi-interest problem.

6 Discussions, conclusions, and future works

6.1 Discussions

In conclusion, our study achieved good results in the scenario of multi-interests and improved localization of item vectors by negative samples. Furthermore, we believe that our research is also effective in other fields. Since the query from the query extraction module contains information of multiple interests, the recommendations based on it can better meet the needs of users. Therefore, it can be applied in the field of multi-interest recommendations. Furthermore, our proposed SSL framework can also be integrated with other recommendation fields, such as collaborative filtering. Our model also has some drawbacks. The addition of a denoising autoencoder and a new SSL framework to DAS-GNN brings more hyperparameters, which increase the difficulty of parameter adjustment for the model.

6.2 Conclusions and future works

In the paper, we propose the DAS-GNN model to address two problems in session-based recommendations, which are the multiple interests and data sparsity problems. To address the first problem, we propose a query extraction module that can aid in the discovery of the user's multiple interests. For the problem of data sparsity, we propose a new SSL framework that alleviates data sparsity by creating hard samples to help the model learn fine-grain information.

We hope that in the future, the proposed model will be applied not only in the field of session recommendations but also in other fields, such as short video and web page recommendations. As the autoencoder can restore the session information to a great extent, the query can discover a user's multiple interests. Therefore, we make multi-interest recommendations as one of our future works. Additionally, not only considering sequential relationships, we intend to introduce the time factor into our model to further distinguish the importance of a user's multiple interests.

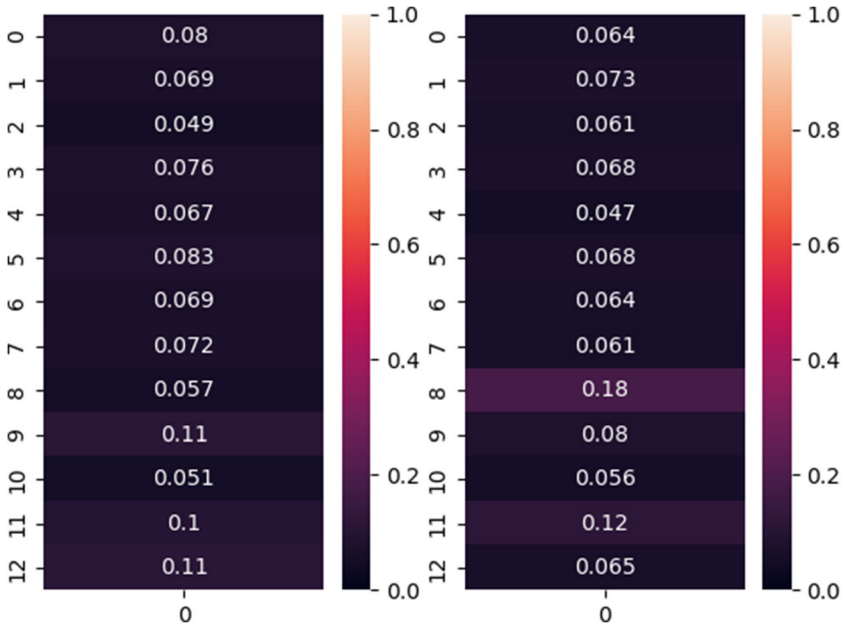


Fig. 14 The left sub-graph indicates attention scores computed when the short-term interest is the query. The right sub-graph shows attention scores when the query is from the query extraction module

Acknowledgments This paper is partially supported by the National Natural Science Foundation of China (61902221, 62177031), the Natural Science Foundation of Shandong Province (ZR2021MF099, ZR2022MF334), and Undergraduate Education Reform Project of Shandong Province (M2021130).

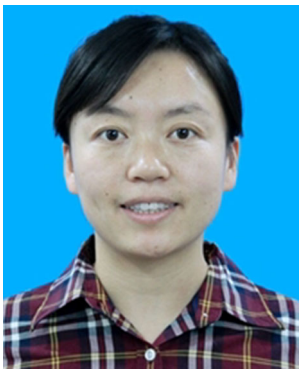
References

1. Lops P, De Gemmis M, Semeraro G (2011) Content-based recommender systems: State of the art and trends. In: Recommender systems handbook. Springer, pp 73–105
2. Schafer JB, Frankowski D, Herlocker J et al (2007) Collaborative filtering recommender systems [M]. The adaptive web. Springer. 291–324
3. Burke R (2002) Hybrid recommender systems: Survey and experiments [J]. User Model User-Adap Inter 12(4):331–370
4. Wang S, Cao L, Wang Y et al (2021) A survey on session-based recommender systems [J]. ACM Comput Surv 54(7):1–38
5. Qiu R, Li J, Huang Z et al (2019) Rethinking the item order in session-based recommendation with graph neural networks; proceedings of the Proceedings of the 28th ACM International Conference on Information and Knowledge Management, F, [C]
6. Wu S, Tang Y, Zhu Y et al (2019) Session-based recommendation with graph neural networks; proceedings of the Proceedings of the AAAI Conference on Artificial Intelligence, F, [C]
7. Huang C, Chen J, Xia L et al (2021) Graph-enhanced multi-task learning of multi-level transition dynamics for session-based recommendation; proceedings of the Proceedings of the AAAI Conference on Artificial Intelligence, F, [C]
8. Wang Z, Wei W, Cong G, Li X-L, Mao X-L, Qiu M, Feng S (2020d) Exploring global information for session based recommendation. arXiv preprint arXiv:2011.10173
9. Xia X, Yin H, Yu J et al (2021) Self-Supervised Graph Co-Training for Session-based Recommendation; proceedings of the Proceedings of the 30th ACM International Conference on Information & Knowledge Management, F, [C]
10. Xia X, Yin H, Yu J et al (2021) Self-supervised hypergraph convolutional networks for session-based recommendation; proceedings of the Proceedings of the AAAI Conference on Artificial Intelligence, F, [C]
11. Chen T, Wong RC-W (2020) Handling information loss of graph neural networks for session-based recommendation; proceedings of the Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, F, [C]
12. Liu Q, Zeng Y, Mokhosi R et al (2018) STAMP: short-term attention/memory priority model for session-based recommendation; proceedings of the Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, F, [C]
13. Yuan J, Song Z, Sun M, et al. Dual Sparse Attention Network For Session-based Recommendation; proceedings of the Proceedings of the AAAI Conference on Artificial Intelligence, F, 2021 [C]
14. He X, Deng K, Wang X et al (2020) Lightgcn: Simplifying and powering graph convolution network for recommendation; proceedings of the Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, F, [C]
15. Mao K, Zhu J, Xiao X et al (2021) UltraGCN: Ultra Simplification of Graph Convolutional Networks for Recommendation; proceedings of the Proceedings of the 30th ACM International Conference on Information & Knowledge Management, F, [C]
16. Mao K, Zhu J, Wang J et al (2021) SimpleX: A Simple and Strong Baseline for Collaborative Filtering; proceedings of the Proceedings of the 30th ACM International Conference on Information & Knowledge Management, F, [C]
17. Gao Z, Cheng Z, Pérez F et al (2022) MCL: Mixed-centric loss for collaborative filtering. In: Proceedings of the ACM Web Conference 2022, pp 2339–2347
18. Yang Z, Ding M, Zou X et al (2022) Region or global A principle for negative sampling in graph-based recommendation. IEEE Transactions on Knowledge and Data Engineering
19. He K, Fan H, Wu Y et al (2020) Momentum contrast for unsupervised visual representation learning; proceedings of the Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, F, [C]
20. He K, Chen X, Xie S et al (2022) Masked auto encoders are scalable vision learners. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 16000–16009
21. Chen T, Kornblith S, Norouzi M et al (2020) A simple framework for contrastive learning of visual representations; proceedings of the International conference on machine learning, F, [C]. PMLR
22. Sang S, Liu N, Li W et al (2022) High-order attentive graph neural network for session-based recommendation [J]. Appl Intell 52: 16975–16989
23. Sedhain S, Menon A K, Sanner S et al (2015) Autorec: Autoencoders meet collaborative filtering; proceedings of the Proceedings of the 24th international conference on World Wide Web, F, [C]
24. Li S, Kawale J, Fu Y (2015) Deep collaborative filtering via marginalized denoising auto-encoder; proceedings of the Proceedings of the 24th ACM international on conference on information and knowledge management, F, [C]
25. Wu Y, Dubois C, Zheng AX et al (2016) Collaborative denoising auto-encoders for top-n recommender systems; proceedings of the Proceedings of the ninth ACM international conference on web search and data mining, F, [C]
26. Liang D, Krishnan RG, Hoffman MD et al (2018) Variational autoencoders for collaborative filtering; proceedings of the Proceedings of the 2018 world wide web conference, F, [C]
27. Zhou W, Lee D-H, Selvam RK, Lee S, Lin BY, Ren X (2021) Pre-training text-to-text transformers for concept-centric common sense. International Conference for Learning Representation
28. Oord A, Li Y, Vinyals O (2018) Representation learning with contrastive predictive coding. arXiv preprint arXiv:1807.03748
29. Zhou K, Wang H, Zhao WX et al (2020) S3-rec: Self-supervised learning for sequential recommendation with mutual information maximization; proceedings of the Proceedings of the 29th ACM International Conference on Information & Knowledge Management, F, [C]
30. Wu J, Wang X, Feng F et al (2021) Self-supervised graph learning for recommendation; proceedings of the Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, F, [C]
31. Luo J, Zhang X (2022) Convolutional neural network based on attention mechanism and Bi-LSTM for bearing remaining life prediction [J]. Appl Intell 52(1):1076–1091
32. Vaswani A, Shazeer N, Parmar N et al (2017) Attention is all you need; proceedings of the Advances in neural information processing systems, F, [C]
33. Yao T, Yi X, Cheng DZ et al (2021) Self-supervised Learning for Large-scale Item Recommendations; proceedings of the Proceedings of the 30th ACM International Conference on Information & Knowledge Management, F, [C]
34. Zhuo J, Zhu Q, Yue Y et al (2022) Learning explicit user interest boundary for recommendation. In: Proceedings of the ACM Web Conference 2022, pp 193–202
35. Wang X, He X, Wang M et al (2019) Neural graph collaborative filtering; proceedings of the Proceedings of the 42nd international ACM SIGIR conference on Research and development in Information Retrieval, F, [C]

36. Ren P, Chen Z, Li J et al (2019) Repeatnet: A repeat aware neural recommendation machine for session-based recommendation; proceedings of the Proceedings of the AAAI Conference on Artificial Intelligence, F, [C]
37. Tan YK, Xu X, Liu Y (2016) Improved recurrent neural networks for session-based recommendations; proceedings of the Proceedings of the 1st workshop on deep learning for recommender systems, F, [C]
38. Yu F, Zhu Y, Liu Q et al (2020) TAGNN: Target attentive graph neural networks for session-based recommendation; proceedings of the Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, F, [C]
39. Wang Z, Wei W, Cong G et al. (2020) Global context enhanced graph neural networks for session-based recommendation; proceedings of the Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, F, [C]
40. Vo DV, Tran TT, Shirai K et al (2022) Deep generative networks coupled with evidential reasoning for dynamic user preferences using short texts. IEEE Transactions on Knowledge and Data Engineering
41. Liang S, Ren Z, Zhao Y et al (2017) Inferring dynamic user interests in streams of short texts for user clustering [J]. ACM Trans Inf Syst 36(1):1–37
42. Sarwar B, Karypis G, Konstan J et al (2001) Item-based collaborative filtering recommendation algorithms; proceedings of the Proceedings of the 10th international conference on World Wide Web, F, [C]
43. Rendle S, Freudenthaler C, Schmidt-Thieme L (2010) Factorizing personalized markov chains for next-basket recommendation; proceedings of the Proceedings of the 19th international conference on World wide web, F, [C]
44. Hidasi B, Karatzoglou A, Baltrunas L, Tikk D (2016) Session-based recommendations with recurrent neural networks. ICLR
45. Li J, Ren P, Chen Z et al (2017) Neural attentive session-based recommendation; proceedings of the Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, F, [C]
46. Wang J, Ding K, Zhu Z et al (2021) Session-based recommendation with hypergraph attention networks; proceedings of the Proceedings of the 2021 SIAM international conference on data mining (SDM), F, [C]. SIAM
47. Tang J, Wang K (2018) Personalized top-n sequential recommendation via convolutional sequence embedding; proceedings of the Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, F, [C]
48. Chen F, Wang YC, Wang B et al (2020) Graph representation learning: a survey. APSIPA Transactions on Signal and Information Processing:9
49. Rousseeuw PJ (1987) Silhouettes: a graphical aid to the interpretation and validation of cluster analysis [J]. J Comput Appl Math 20: 53–65
50. Holzinger A, Saranti A, Molnar C et al (2022) Explainable AI methods-a brief overview; proceedings of the International Workshop on Extending Explainable AI Beyond Deep Models and Classifiers, F, [C]. Springer
51. Holzinger A, Malle B, Saranti A et al (2021) Towards multi-modal causability with graph neural networks enabling information fusion for explainable AI [J]. Inf Fusion 71:28–37

Jiuqian Dai He was born in Shandong, China, in 1997. He received the B.S. degree in computer science and technology from Shandong University of Science and Technology in 2019. Currently, he is a M.S. candidate in School of Computer Science and Technology, Shandong Jianzhu University. His recent research interests include personalized recommendation, machine learning and data mining.

Chen Bao He was born in Anhui, China, in 1997. He received the B.S. degree in college of energy and mechanical engineering from Jiangxi University of Science and Technology in 2020. Currently, he is a M.S. candidate in School of Computer Science and Technology, Shandong Jianzhu University. He recent research interests include personalized recommendation, machine learning and data mining.



Weihua Yuan She was born in Shandong, China, in 1977. She received her Ph.D. degree in School of Information Science and Engineering, Shandong Normal University. She is currently an associate professor of School of Computer Science and Technology, Shandong Jianzhu University. Her recent research interests include personalized recommendation and machine learning.

Zhijun Zhang He was born in Shandong, China, in 1973. He received his Ph.D. degree in School of Information Science and Engineering, Shandong Normal University. He is currently a professor of School of Computer Science and Technology, Shandong Jianzhu University. His recent research interests include personalized recommendation, machine learning and data mining.