International Journal for Housing Science and Its Applications
Publish August 10, 2025. Volume 46, Issue 4 Pages 1716-1731

https://doi.org/10.70517/ijhsa464150

A Bidirectional Enhancement Method for Contrastive Learning Optimization of Language Model Recommendation Data

Zuming Ka[1,*], Peng Zhao[2] and Bo Zhang[3]
[1] College of Operational Support, Rocket Force University of Engineering, Xi'an, Shaanxi, 710025, China
Corresponding authors: (e-mail: 15691790282@163.com).

Abstract Recommender systems are created to help users screen and filter the huge amount of data generated in the Internet, so that users and data can be matched perfectly. This paper establishes the CGSNet session recommendation model from the perspective of session recommendation system on the basis of explicit language model recommendation. The CGSNet model is based on graph convolutional neural network, which extracts the neighborhood information of the user's session through the fusion of the attention mechanism, and combines the global session representation learning to construct the global graph of the item. The self-supervised comparison learning module is introduced to realize the effective exchange of neighborhood information and global information, and the joint learning objective is designed to optimize the session model recommendation performance. The results show that the P@15 and MRR@15 metrics of the CGSNet model on the Tmall dataset are improved by 15.89% and 19.11%, respectively, compared with the suboptimal model. The effectiveness of the model is verified through multiple types of simulation experiments, which also illustrates the feasibility of comparative learning for optimizing recommendation data in session recommendation.

Index Terms graph convolutional neural network, attention mechanism, neighborhood information, CGSNet model, contrast learning

I. Introduction

Big language models have extensive research value and great application potential in the field of personalized recommendation, which aims to provide users with customized recommended content based on their interests and preferences [1]. However, traditional personalized recommendation methods still have certain challenges in dealing with the cold-start problem, data sparsity and long-tail item recommendation [2], [3]. Previous research efforts have relied on user behavioral data and item feature information for recommendation, but these approaches have limitations in capturing user intent and semantics [4], [5]. With the emergence of large language models, techniques such as deep learning and supervised fine-tuning can be applied to enable powerful semantic understanding and generation capabilities, bringing new opportunities for personalized recommendation research [6]-[8].

The lack of resource auditing also affects the recommendation quality of recommender systems, as resource search platforms face the challenge of imperfect resource auditing mechanisms, a large number of low-quality resources are uploaded and become popular recommendations, which ultimately leads to a decline in the quality of recommendations [9]-[11]. Most recommender system research focuses on model improvement and enhancement, but ignores the data quality problem [12]. While data quality is crucial for the training and learning of recommendation models, improving the quality of resources can effectively improve the recommendation results, so the study of resource audit classification methods is also of great significance for the field of online education [13], [14]. Among them, comparative learning enhances the model's ability to understand data distribution, slows down the influence of popular recommended resources on cold recommended resources, and improves the generalization ability of language models [15]. Contrastive learning is used to achieve the contrast alignment between the interaction features and semantic features of the resources in order to make full use of the semantic information combined with the interaction features to achieve the personalized recommendation of the resources [16].

The article establishes a recommendation model for user conversational language using graph convolutional neural network and self-supervised contrastive learning. The CGSNet model mainly consists of neighborhood information representation learning, global conversation representation learning, self-supervised contrastive learning, and model optimization prediction layer. In the neighborhood information representation learning layer, the GGNN is mainly used to convey and aggregate the information of nodes in the session graph. In the global session representation learning layer, a session-aware attention mechanism is introduced to calculate the contribution value of neighboring nodes to the current node in the global graph and estimate the weights of different neighbors. In the self-supervised contrast learning module, the information exchange between the neighbor information and the global session is achieved by contrasting the learning of positive and negative samples, maximizing the consistency between the two, and minimizing the gap between the two for the loss of comparison calculation. In the model optimization and prediction layer, the idea of InfoNCE is used to improve the performance of the model. Finally, four datasets of different magnitudes are selected to verify the application performance of the CGSNet model with two metrics, P@K and MRR@K.

II. Methodology

The essence of language model recommendation is to realize the session recommendation between the machine and the user, and the language model relies on the user's historical session interaction information, so as to mine the user's interest preferences and then realize the user's demand recommendation. Based on this, this paper starts the research from the session recommendation model of language model recommendation, and explores the session recommendation model combining comparative learning and graph neural network, which provides support for mining users' interests and preferences and forming users' language models.

II. A.Session-based recommender systems

II. A. 1) Session Recommender System

An recommender system is an algorithmic tool that provides content recommendations to users and is mainly used to solve the information overload problem faced by users in the context of the information explosion on the Internet. It can be used to provide users with decision-making strategies to help them find content that matches their preferences, as well as to help them make choices without the appropriate knowledge and experience. The core concept of the recommender system model is to take the degree of recommendation of others for different items as input , the recommender system aggregates the input data, extracts the correlation information contained therein, and generates personalized recommendations for the user looking for recommendations based on the extracted correlation information for the user [17]. We can categorize its overall framework into three phases, i.e., data collection phase, learning phase, and prediction/recommendation phase.

In the data collection phase, the recommender system needs to collect user-item interaction data, and for user-related information, the traditional recommender system model cannot accurately model user-item correlation information without having enough user-related information to generate accuracy recommendation results.

In the learning phase, the recommender system attempts to model user-item related information from the input data to extract effective information for recommendation.

In the prediction/recommendation phase, the recommender system model models the user based on the user's existing interaction information and generates user-oriented recommendation results accordingly.

II. A. 2) Recommended problem description

In today's age of information explosion, recommender systems play a crucial role in helping users discover personalized content from massive amounts of data. Session recommendation, as an important branch of recommender systems, focuses on analyzing users' behavioral patterns over a short period of time to predict and recommend items that may be of interest to them. The main challenge for session recommender systems is to accurately capture a user's interests from their short interaction history. These systems provide real-time recommendations to users by analyzing their click sequences over a specific time period and predicting the next possible clicked item.

Session-based recommender systems aim to provide attractive recommendations to anonymous users. Therefore, it requires language models that accurately capture a user's interests by using an event or a specific time interval rather than a complete history of interactions.

Assuming that there are m items and n sessions, let $V=\{v_1,v_2,\dots,v_m\}$ and $S=\{s^1,s^2,\dots,s^n\}$ denote the set of items and sessions respectively, and s^i denotes the i th session. Each session s^i is a chronologically ordered sequence $[v_1^i,v_2^i,\dots,v_l^i]$, $v_j^i\in V$ denotes the item that was clicked on in the j th session, and 1 denotes the length of the session. The goal is to predict the next item v_{l+1}^i of any session s^i, generate a session representation based on the item representations in the session, and finally compute the probability based on the similarity between the session representation and all the item embeddings and perform a Top-k recommendation. The output of the session-based recommendation model is a sorted list $y=\{y_1,y_2,\dots,y_m\}$, where $y_i(1\le i\le m)$ is the predicted probability of the corresponding item i . Finally, the first k item $(1\le k\le m)$ with the highest probability is chosen as the prediction result.

II. B. Graph Convolutional Networks and Comparison Learning

II. B. 1) Graph Convolutional Neural Networks

The types of graph neural networks in session recommendation can be categorized into three types: gated graph neural networks, graph convolutional neural networks, and graph attentional neural networks. A session can be viewed as a sequence or collection, and also a graph structure can be formed by the relationship of items in the session. Complex transformations between items can be better captured by graph neural networks, leading to the learning of more accurate item vector representations. Using graph neural networks for session recommendation generally involves two steps:

(1) Composition. Given a dataset containing multiple sessions, construct a session graph for each session, which in turn constructs a global graph for the entire dataset. Essentially both session graphs and global graphs are optional, and as long as one of the graphs is constructed in the method and a graph neural network is used, it is considered to be a graph neural network based method.

(2) Use the graph neural network to propagate and aggregate information between graph nodes on the graph data structure constructed in step (1) to complete the learning of node vector representations.

For a given session $s=[v_1,v_2,v_3,v_4]$, according to the transformation relationship of the items in the session in the session, a directed graph can be formed, and the nodes of the graph are the items in the session. The number of graph nodes will generally be smaller than the session length due to the presence of duplicate items in the session [18]. A directed graph can be represented by an out-degree matrix and an in-degree matrix. Once the session graph is constructed, a graph neural network can be used to learn the dependencies between items and obtain a new item vector representation.SR-GNN uses gated graph neural network (GGNN) for information propagation and aggregation between nodes, i.e.:

$$a^t_{s,i} = A_{s,i} : [v^{s1}_1, \cdots, v^{s1}_n]^T H + b \tag{1}$$

$$z^t_{s,i} = \sigma(W_z a^t_{s,i} + U_z v^{t^1}_i) \tag{2}$$

$$r^t_{s,i} = \sigma(W_r a^t_{s,i} + U_r v^{s1}_i) \tag{3}$$

$$\tilde{v}^t_i = \tanh(W_o Q^t_{s,i} + U_o (r^t_{s,i} \square\ v^{t-1}_i)) \tag{4}$$

$$v^t_i = (1 - z^t_{s,j}) \square\ v^{t-1}_i + z^t_{s,i} \square\ \tilde{v}^t_i \tag{5}$$

SR-GNN session graphs can only learn the dependencies between items inside a session and cannot learn the dependencies between items inside a session and items outside a session. In addition, the way the graphs are constructed tends to result in different sessions constituting the same graph. In order to solve the above problems, LESSR preserves the original order information of items in a session when constructing its edge-order keeping graph, and uses the GRU module to aggregate neighboring items in order, i.e:

$$v^{(l)}_{v_i,k} = GRU^{(l)}(n^{(l)}_{v_i,k1}, x^{(l)}_{v_i,k}) \tag{6}$$

where $v^{(l)}_{v_i,k}$ denotes the k th neighboring item vector of item v_i in order, and $n^{(l)}_{v_i,k}$ denotes the vector representation of it after aggregating the k th neighboring items.

Graph neural networks are often used in session recommendation to learn the dependencies between items. Due to its robust representational properties, the ability to learn item dependencies is stronger than methods based on recurrent neural networks and attention mechanisms. Due to the different ways of information dissemination and aggregation of graph nodes, the way they are constructed will also differ.

II. B. 2) Self-supervised comparison learning

Self-supervised learning (SSL) is an emerging unsupervised learning paradigm that reduces the reliance on manual labeling and enables training on massive unlabeled data [19]. Self-supervised learning aims to learn valuable representational information to help downstream tasks by designing pre-tasks (i.e., self-supervised tasks) to extract informative and transferable knowledge from large amounts of unlabeled data, and generating supervised signals semi-automatically. Self-supervised learning is able to cope with the problem of insufficient data labeling prevalent in various scenarios, and has a very broad range of applications, including language model pre-training, audio representation learning, and visual representation learning.

The basic idea of contrast learning is to consider each instance (e.g., picture, utterance, sequence, etc.) as a class, and create views based on the original data through various transformation methods, and bring the views of the same instance closer and push the views of different instances apart in the embedding space [20]. In general, two views of the same instance are considered as a pair of positive views, while views of different instances are considered as negative samples of each other. Positive views should introduce non-essential changes into the instance without modifying the semantics. By maximizing the consistency between positive pairs of views while minimizing the consistency between negative pairs of views, the model obtains distinguishable and generalized representations of instances. The formula for the contrast learning model can be written as:

$$f_\theta^* = \arg\min_{f_\theta, g_{\phi_s}} L_{ssl}(g_{\phi_s}(f_\theta(\tilde{D}_1), f_\theta(\tilde{D}_2))) \tag{7}$$

where $\tilde{D}_{(1)} \sim T_1(D), \tilde{D}_{(2)} \sim T_2(D)$ are two views of D obtained by the data augmentation operations $T_1(\cdot)$ and $T_2(\cdot)$. The loss function L_{ssl} computes the mutual information between the views by means of the vector representations learned by the encoder f_θ. The projection head g_{ϕ_s} mostly uses simple structures such as bilinear networks.

The optimization objective of contrast learning is to maximize the mutual information between orthogonal views, and in practical computation, the lower bound of mutual information is usually maximized. InfoNCE is one of the most commonly used contrast learning loss functions today, which is denoted as:

$$MJ_{NCE} = -E\left[\log \frac{e^{f_D(h_i,h_j)}}{\sum_{n \in N_i^- \cup \{j\}} e^{f_D(h_i,h_n)}}\right] \tag{8}$$

where j is the set of positive samples of i and N_i^- is the set of negative samples of i, usually obtained by sampling within a batch. $f_p(\cdot)$ is the similarity calculation function, or cosine similarity if used:

$$f_D(h_i,h_j) = \frac{h_i^T h_j}{\| h_i \| \| h_j \|} / \tau \tag{9}$$

where τ is the temperature coefficient, the smaller the temperature coefficient, the greater the differentiation between positive and negative samples, the more obvious the effect of comparative learning, the model will pay special attention to difficult samples. But on the other hand, these difficult samples are likely to be mixed with potentially positive samples, too large a temperature coefficient will lead to a decline in the effect.

II. B. 3) Attention mechanisms

Attention mechanism refers to focusing attention on something at a certain moment and ignoring other unimportant things around. Humans quickly scan the global information, focus only on the target information and devote more resources to that target information, and obtain more details of the target information by ignoring other information around them, which is the brain information processing mechanism unique to human vision, i.e., the visual attention mechanism. The attention mechanism in deep learning is also used to improve the accuracy and performance of the model by weighting the important regions in a large amount of data [21].

(1) Soft Attention Mechanisms

The soft attention mechanism is usually applied in sequence processing tasks, by learning the weight distribution of each element in the sequence to indicate the degree of attention of the model to each element, helping the model can focus on the important information in the sequence in order to better understand the semantics of the input sequence. The workflow of the soft attention mechanism is as follows:

In the first stage, the attention scores, i.e., weight assignments, are computed for all input data. Assuming the input data $X = \{x_1, x_2, ..., x_n\}$ and the query vector Query, there are three ways to compute the formula for the attention distribution:

$$Similarity(Query, x_i) = Query \cdot x_i \tag{10}$$

$$Similarity(Query, x_i) = \frac{Query \cdot x_i}{\| Query \| \cdot \| x_i \|} \tag{11}$$

$$Similarity(Query, x_i) = MLP(Query \cdot x_i) \tag{12}$$

The range of scores calculated is related to the computational method chosen in the first stage, and the second stage introduces the SoftMax layer for normalization, and the output value is the attentional weight of each keyword, i.e:

$$\alpha_i = softmax(sim_i) = \frac{\exp(sim_i)}{\sum \exp(sim_j)} \tag{13}$$

The third stage calculates the attention weighted sum to get the output of the attention network as:

$$Attention(X, Query) = \sum_{i=1}^{n} \alpha_i x_i \tag{14}$$

(2) Self-attentive mechanism

The self-attention mechanism is characterized by its ability to capture long-distance dependencies in a sequence. Each element in the self-attention mechanism can establish associations with other elements in the sequence, rather than just relying on elements in neighboring positions. It adapts to capture long distance dependencies between elements by calculating the relative importance between elements.

In self-attentive networks, three new vectors, Q, K, and V, need to be computed to measure the importance of each position in the input sequence, to represent the information of each position in the input sequence, and to represent the actual value of each position in the input sequence, respectively. These three vectors are obtained from the input data by linear transformation.

The specific computational procedure for single-head attention is:

$$\begin{cases} Q = IW_Q \\ K = IW_K \\ V = IW_V \\ Attentin(Q, K, V) = softmax(\frac{QK}{\sqrt{d}})V \end{cases} \tag{15}$$

where $I \in R^{N \times d}$ is the original input to the self-attention network, N denotes the number of input messages, and d denotes the dimension of the input messages.

II. C.CGSNet-based session recommendation modeling

II. C. 1) Session Recommendation Model Structure

Figure 1 shows the modeling framework of the Contrastive Graph Self-Attention Network (CGSNet). First, CGSNet is designed with two different types of graph encoders (a global session representation learning graph encoder as well as a neighborhood information representation learning encoder) to adaptively capture all item transition patterns in a session. Since the two graph encoders learn item representations from a global perspective and different items in a session have different levels of importance in learning that session, CGSNet thus designs a fusion module based on an attentional mechanism to learn collaborative session representations that represent users' long-term preferences. In addition, since sessions usually consist of a limited number of short-term user interactions, session representation modeling based only on graph neural networks is susceptible to data sparsity, so in this paper, we also use a Transformer-based self-attention sub-network to learn local item representations and obtain local session representations representing users' short-term preferences by applying an average pooling operation to the local item representations. In order to achieve mutual learning between session representations at different levels and to maximize the use of interaction information between collaborative sessions and local session representations, the model incorporates a contrastive learning paradigm into the training process of CGSNet. Finally, CGSNet obtains the next recommended item for a given session by computing the predicted score for each candidate item.

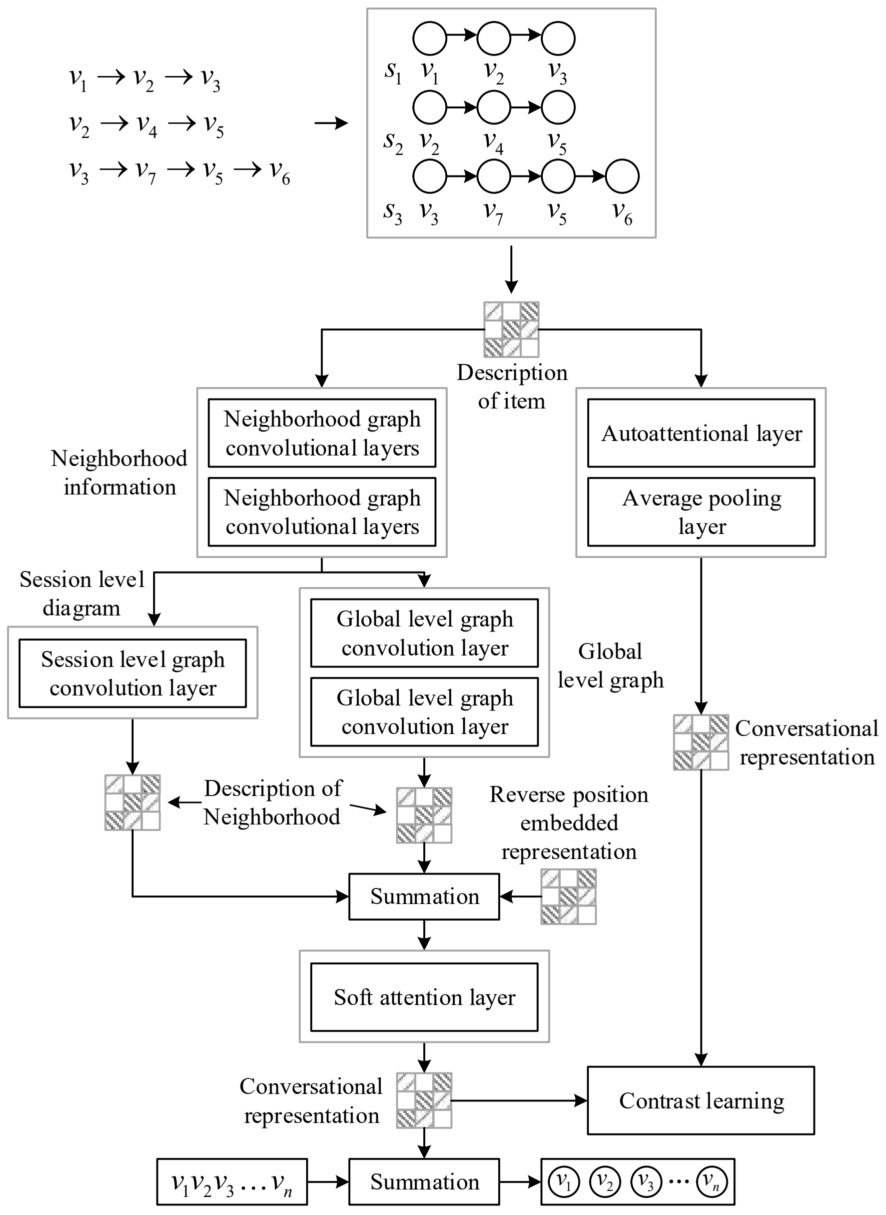

$v_1 \to v_2 \to v_3$
$v_2 \to v_4 \to v_5$
$v_3 \to v_7 \to v_5 \to v_6$
s_1 v_1 v_2 v_3
s_2 v_2 v_4 v_5
s_3 v_3 v_7 v_5 v_6
Description of item
Neighborhood information
Neighborhood graph convolutional layers
Neighborhood graph convolutional layers
Autoattentional layer
Average pooling layer
Session level diagram
Session level graph convolution layer
Global level graph convolution layer
Global level graph convolution layer
Global level graph
Conversational representation
Description of Neighborhood
Reverse position embedded representation
Summation
Soft attention layer
Conversational representation
Contrast learning
$v_1 v_2 v_3 \dots v_n$
Summation
v_1 v_2 v_3 $\cdots$ v_n

Figure 1: CGSNet model framework

II. C. 2) Neighborhood Information Representation Learning

In the neighborhood information learning layer, GGNN is utilized to convey and aggregate the information of nodes in the session graph.

First, each session sequence s is constructed as a directed graph $g_s = (\gamma_s, \varepsilon_s)$, called the session graph, where γ_s and ε_s distinguishably represent the set of nodes and edges. In the session graph, each node corresponds to an item $v_{s,i} \in \mathrm{V}$, and each edge $(v_{s,i-1}, v_{s,i}) \in \varepsilon_s$ denotes that a user in session s accesses the item $v_{s,i-1}$ after accessing item $v_{s,i}$. Since there may be recurring items in the session, a normalized weight is assigned to each edge, calculated by dividing the number of occurrences of the edge by the out-degree of the starting node. Each item $v \in V$ is embedded into a uniform embedding space, and the embedding vector $v_i \in R^d$ of the node represents the hidden vector of item i with dimension d.

The update process of the node vectors on a given session graph s is as follows:

$$a_{s,i}^t = A_{s,i;}^T [v_1^{t-1}, \dots, v_N^{t-1}]^T W^H + b \tag{16}$$

$$z_{s,i}^t = \sigma(W^z a_{s,i}^t + U^z v_i^{t-1}) \tag{17}$$

$$r_{s,i}^{t} = \sigma(W^{r} a_{s,i}^{t} + U^{r} v_{i}^{t-1}) \tag{18}$$

$$\tilde{c}_{l,i}^{t} = tanh(W^{o} a_{v}^{t} + U^{o}(r_{s,i}^{t} \odot v_{i}^{t-1})) \tag{19}$$

$$\hat{c}_{l,i}^{t} = (1 - z_{s,i}^{t}) \odot v_{i}^{t-1} + z_{s,i}^{t} \odot \tilde{v}_{i}^{t} \tag{20}$$

where $W^{H} \in R^{d\times 2d}$ and $b \in R^{d}$ control the weights of the nodes as well as the size of the bias term, and $z_{s,i}$ and $r_{s,i}$ are the updating and resetting gates, respectively. W^{z} and U^{z} are the learnable network parameters in the update gate, and W^{r} and U^{r} are the learnable network parameters in the reset gate. $[v_{1}^{t-1},...,v_{N}^{t-1}]$ is a list of node vectors in the s session. $\sigma(-)$ denotes the sigmoid function, and $\odot$ is the Hamada product, which denotes the dot product operation. The W^{o} and U^{o} denote the learnable network parameters in the output gate. The adjacency matrix $A_{s} \in R^{n\times 2n}$ determines the message passing mode of the nodes in the graph, and $A_{s,i} \in R^{1\times 2n}$ denotes the two-column block matrix of the node $v_{s,i}$ corresponding to the node A_{s}. A_{s} is a splice of $A(out)_{s}$ and $A(in)_{s}$, representing the weighted adjacency matrices of the outgoing and incoming edges, respectively.

For each session graph G_{s}, GGNN is able to compute all nodes simultaneously. Under the constraint of the adjacency matrix A_{s}, Eq. (16) is responsible for transferring information between different nodes, extracting potential embedding representations of neighboring nodes and feeding them into the graph neural network. Two gating mechanisms, the update gate in Eq. (17) and the reset gate in Eq. (18), determine which information is retained or discarded, respectively. Eq. (19) shows how the information from the previous state, the current state, and the reset gate can be utilized to construct candidate states. The final state in equation (20) is a combination of the previous state and the candidate state under the control of the update gate. The final node embedding representation is obtained after all nodes in the session graph are updated to convergence. In this paper, the time step is set to 1 in order to minimize the effects of excessive smoothing and gradient explosion.

Therefore, in this layer, the GGNN will mainly aggregate the direct information on the first-order neighbors while fusing some of the indirect information on the more distant distances. The final node vector C_{l} will continue to be passed into the global session representation learning layer used to learn the final session embedding vector.

II. C. 3) Global session representation learning

The global graph computes the contribution value information between neighboring nodes, and the information dissemination between nodes draws on a session-aware attention mechanism in the GCE-GNN model, which is used to compute the contribution value of neighboring nodes to the current node in the global graph, and to estimate the weights of different neighbors. That is, the closer an item is to the preference of the current session, the more important the item is for the next recommendation. The global graph information propagation formula is:

$$\pi(v_{i}, v_{j}) = q_{1}^{T} LeakyReLU(W_{3}[s \odot h_{v_{j}} \| w_{ij}]) \tag{21}$$

$$s = \frac{1}{|S|}\sum_{v_{i}\in S} h_{v_{i}} \tag{22}$$

where $\pi(v_{i}, v_{j})$ is the contribution value between node v_{i} and v_{j}, q_{1} is a learnable parameter, w_{3} is a learnable weight matrix, $h_{v_{j}}$ is a representation of the node, s computed by Eq. (22), considered as the current session feature, obtained by computing the average of the current session item representations, S represents the current session sequence, $|S|$ is the number of nodes issued by the session S, w_{ij} is the edge weights between the nodes v_{i} and v_{j}, and $\odot$ is the Hadamard product operation, $\|$ is the splicing operation, and $[\cdot]$ is used to specify the range of the splicing operation. The weights of all neighboring nodes connected to node v_{i} in the global graph are normalized using the SoftMax function to compute attention scores that are able to indicate which neighboring nodes should have a greater weight in the result. I.e:

$$\pi(v_{i}, v_{j}) = softmax(\pi(v_{i}, v_{j})) \tag{23}$$

Ultimately, the multi-order node relationships in the global graph are obtained by weighted neighborhood aggregation and superimposing multiple layers, i.e:

$$h_{N_{v_i}^g} = \sum_{v_j \in N_{v_i}^g} \pi(v_i, v_j) h_{v_j} \tag{24}$$

where $h_{N_i^g}$ is the neighborhood representation of v_i node. Information aggregation is one of the core operations in GNN, where each node updates its feature representation based on its neighboring nodes, which serves to aggregate the item representation and its neighboring representations of each node to obtain features. Information aggregation is mainly applied to capture graph structure, feature propagation and learning higher-order interrelationships, etc. The information aggregation formula is:

$$h_{v_i}^g = ReLU(W_4[h_{v_i} \| h_{N_{v_i}^g}]) \tag{25}$$

where $h_{v_i}^g$ is the global item representation, W_4 is the transformation weight and ReLU is chosen as the activation function, h_{v_i} is the feature representation of the current node, and $h_{N_i^g}$ is the feature representation of neighboring nodes, and the information aggregation is performed by splicing the obtained features and the original features are spliced to form the final representation. With the single-layer aggregator, the node representation depends not only on its own features but also on the representations of its neighbors. To explore higher-order connectivity information, this paper introduces multi-layer stacking of aggregators, which enables the representation of the target node to incorporate more relevant information. In the k th layer, the item is represented as:

$$h_{v_i}^{g,(k)} = agg(h_{v_i}^{g,(k-1)}, h_{N_{v_i}^g}^{g,(k-1)}) \tag{26}$$

where $h_{v_i}^{g(k)}$ is the representation of node v in the k th layer in the global graph, generated from previous information dissemination, $agg(\cdot)$ is the aggregation function, and $h_{V_i}^g(k-1)$ and $h_{N_i^g}^g(k-1)$ denote, respectively, node v's representation in the $k-1$ th layer of the global graph and the neighbor node representation, respectively. By this method, the node's representation is updated, and valid information can be fused into the current node representation, thus reflecting more comprehensively the structure and dynamics of the node in the graph.

II. C. 4) Self-supervised comparison learning

In order to realize the effective exchange of neighborhood information and global information, this paper introduces the self-supervised contrast learning mechanism, which realizes the exchange of information between each channel by contrasting the learning of positive and negative samples. For the item with the highest correlation with the predicted item, the consistency between the two can be maximized and the gap between them can be minimized to calculate the loss by comparison. Namely:

$$z_i^p = soft\max(I_p S_i^P) \tag{27}$$

$$z_z^p = soft\max(I_p S_h^p) \tag{28}$$

where I_p is the fused embedding of the items, z_i^p is the recommendation probability of each item, and the above is the de-learning of positive and negative samples in two different channels. Then:

$$A = sim(I_{iazt,s} + S_i^p, I_i + S_i^p) / \tau \tag{29}$$

$$B = sim(I_{iazt,s} + S_h^p, I_i + S_h^p) / \tau \tag{30}$$

$$L_a = -1og \frac{\sum_{i \in E_i^p} \exp(A)}{\sum_{i,j \in E_i^p, E_i^N} \exp(A)} \tag{31}$$

$$L_b = -1og \frac{\sum_{i \in E_i^p} \exp(B)}{\sum_{i,j \in E_i^p, E_i^N} \exp(B)} \tag{32}$$

$$L_c = L_a + L_b \tag{33}$$

where $sim(\cdot,\cdot)$ denotes the similarity between the two items, $I_{last.s}$ denotes the last item of the current session, τ is a temperature parameter used to adjust the scale of similarity, and $\varepsilon_i^p, \varepsilon_i^p$ is the sum of the two sets of positive samples derived from the channels, and finally the sum of the two losses L_c .

II. C. 5) Model Optimization and Prediction Layer

Referring to InfoNCE, it maximizes the amount of information shared between different levels of session representations to improve the performance of the model. The formula is as follows:

$$\mathrm{L}_s = -\log\sigma(f_D(s_{v_i}^{final}, s_{v_i}^h)) - \log\sigma(1 - f_D(\tilde{s}_{v_i}^{final}, s_{v_i}^h)) \tag{34}$$

$f_D(\cdot): R^d \times R^d \mapsto R$ is the discriminator function that compares the similarity between two vectors by feeding them. In the model of this paper, the discriminator is implemented as the dot product of the embedded representations of the two sessions. Since both the pairwise relation-based session representation S_{final} and the higher-order relation-based session representation S_h model the same session, they can be each other's truth values. It is corrupted by performing a row-by-row and column-by-column rearrangement of sv_i^{final} to create negative samples $\tilde{s}_{v_i}^{final}$. Finally, the recommendation task is unified with the comparison task and it is optimized by joint learning. Specifically, the joint learning objective is defined as:

$$\mathrm{L} = \mathrm{L}_r + \beta \mathrm{L}_s \tag{35}$$

where β is a hyperparameter used to control the magnitude of the self-supervised contrast learning task.

After completing the above three modules, a session representation S_{final} combining neighborhood and global features is obtained. In the prediction layer, for each potentially recommended item v_i , the prediction score of the node is obtained by applying the SoftMax function to the session-level embedded representation S_{final} and to the embedded representation of each node v_i that incorporates the global information x_i , to obtain the prediction score of the node $haty_i$. The formula is shown below:

$$\hat{y}_i = softmax({S_{final}}^T x_i) \tag{36}$$

where $\hat{y}_i$ denotes the probability that item v_i is clicked at moment $t+1$ in session s , vs:

$$\hat{y} = \{\hat{y}_1, \hat{y}_2, \cdots, \hat{y}_n\} \tag{37}$$

Sorting is performed, with larger values indicating that the item is more likely to be clicked on in the next moment, and therefore more likely to be recommended.

III. Experimentation

Recommender systems have dramatically improved user experience and business efficiency in various online platforms, such as e-commerce sites, social media, and online streaming services. However, traditional recommender systems such as content-based recommendation and collaborative filtering algorithms often ignore the dynamic evolution of user preferences, and thus have limitations in capturing users' latest interests. Unlike traditional recommender systems, session-based recommender systems focus on current session modeling, i.e., they only consider ongoing sessions and sequential sequences encapsulated in the sessions, and thus are more suitable for new users as well as timely and dynamic recommendations.

III. A. Data set and experimental setup

III. A. 1) Data set selection

This paper evaluates the proposed model using experiments with the publicly available Tmall, Dijinetica, Yoochoose, and Retail Rocket datasets.The Tmall dataset is obtained from the IJCAI-15 competition, and consists of user shopping records from Tmall.com.hk. The Dijinetica dataset is obtained from CIKMCup, and consists of user session records from a search engine.This paper only uses its transaction data. user session records, and only its transaction data is used in this paper.The Yoochoose dataset is derived from the RecSys Challenge, and contains a six-month data stream of user behaviors on an e-commerce website, and the types of behaviors include clicks and purchases.The Retail Rocket dataset is derived from a real e-commerce website, and includes sequential view and add-to-cart events. The Retail Rocket dataset is from a real e-commerce site and includes sequential view and add to cart events.

The four datasets are first preprocessed before the model is trained. Sessions of length 1 were filtered out; session records that were too short were not informative, and items with fewer than six occurrences were removed. The last week's data was set as the test set and the rest of the data as the training set. In addition, each session was segmented to generate sequences and corresponding labels. Where the last item clicked in the session is the label of the sequence.

III. A. 2) Parameter setting

In the simulation experiments of this paper, the dimensions of the latent vectors of the model are all set to 120. To avoid the overfitting problem, the Dropout strategy is applied to the CGSNet model, which is set to 0.15 on all the four datasets. The batch size is set to 200 during the training of the model. When fitting the model, the paper chooses Adam as the optimizer with an initial learning rate of 0.002. This experiments were all performed on a Linux workstation equipped with GeForce RTX 3090 (24G).

III. A. 3) Assessment of indicators

In this paper, the following metrics are used to evaluate the performance of the CGSNet model:

(1) The P@K score represents the proportion of test cases in which the correctly recommended items are located in the top K positions of the ranked list. Its expression is:

$$P@K=\frac{n_{hit}}{N} \tag{38}$$

where N is the number of test data in the session recommendation model, n_{hit} is the number of cases that have the desired item in the top K sorted list, and hit occurs when the item t occurs in the first K positions of the sorted list in the system.

(2) MRR@K indicates the average value of the inverse rank of the occurrence of the desired item t. If the rank is greater than K, the rank reversal is 0. It is defined as follows:

$$MRR@K=\frac{1}{N}\sum\frac{1}{Rank(t)} \tag{39}$$

MRR is a normalized score in the range [0,1], whose increasing value reflects the fact that most of the “hits” appear higher in the ranking order of the recommendation list, which indicates a better performance of the corresponding recommendation system.

III. B. Experimental results and analysis

III. B. 1) Performance comparison experiments

In order to validate the performance of the CGSNet model designed in this paper in conversational language model recommendation, according to the evolution of the field of conversational recommender systems from traditional methods to deep learning methods, a total of eight baseline models, one based on traditional methods, two based on RNNs, and five based on GNNs, are selected for comparison with the CGSNet model in this paper. Namely, FPMC, NARM, STAMP, SR-GNN, FGNN, GCE-GNN, DHCN, and Disen-GNN. Figures 2 and 3 show the comparison results of different models with P@15 and MRR@15 on different datasets, respectively.

Based on the data distribution in the figures, combined with the data of P@15 and MRR@15, the following conclusions are drawn:

(1) The deep learning-based models proposed in recent years (NARM, STAMP, SR-GNN, FGNN, GCE-GNN, DHCN, Disen-GNN, and the CGSNet model in this paper) significantly outperform the traditional model (FPMC). This is a good indication that deep learning plays a key role in the field of conversational language model recommendation.

(2) The success of RNN-based models (NARM, STAMP) shows that the order of items in a sequence is intuitively important for conversational language model recommendation. Because conversational language model recommendation may encode user behavior due to its user preferences and introduces an attention mechanism, the performance achieved a response improvement. This shows that assigning different attention weights to different items for conversational language coding is very effective. The graph neural network based models (SR-GNN, FGNN, GCE-GNN, DHCN, Disen-GNN) achieved further performance improvement. Among them, SR-GNN and FGNN proved that graph modeling is more effective than sequence modeling for session language model recommendation by modeling each session as a subgraph and applying it to gated graph neural networks to learn

embedded representations. GCE-GNN and Disen-GNN further enriched the graph modeling and attempted to explore the transformation relations of items in addition to the current session language, which made good progress. This suggests that item transformation information at the global level is also helpful for recommendation. DHCN, on the other hand, introduces a self-supervised task for the first time in session language model recommendation, but only achieves a small improvement due to the fact that it generates self-supervised signals through a rank-randomized transformation of sequential data.

(3) The CGSNet model proposed in this paper first acquires more complex higher-order information in addition to the low-order transitions in the current session through richer graph modeling, followed by variance-invariant representations through a self-supervised contrast learning mechanism, which allows different graph encoders to acquire information about each other and produce more informative self-supervised signals for effective data augmentation, which improves the model performance. Compared with the above baseline model in the four datasets, the CGSNet model has achieved certain improvements, and in the P@15 and MRR@15 indicators, it is 15.89% and 19.11% higher than the suboptimal model on the Tmall dataset, 4.73% and 4.56% on the Dijineca dataset, and 2.32% and 3.59% on the Yoochoose dataset, respectively, and Retail The P@15 indicators on the Rocket dataset increased by 7.04%, and the MRR@15 indicators achieved the suboptimal effect. The improvement is most obvious on the Tmall dataset, which possesses higher consistency than the other three datasets, considering that it is a real e-commerce environment for cell phones.

Combining self-supervised contrastive learning with graph neural networks can achieve optimal conversational language model recommendation, which mainly lies in the fact that self-supervised contrastive learning can augment the data with conversational language as a way to achieve better recommendation, which also suggests that consistency may be more important than strict sequential modeling.

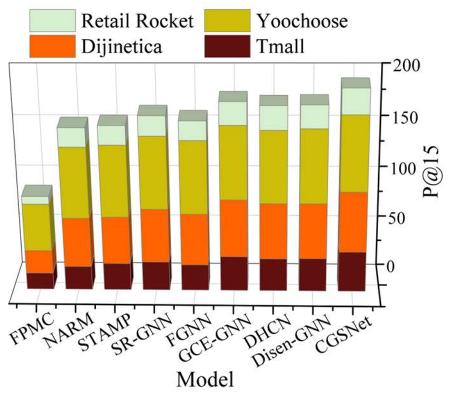


Figure 2: P@15 comparison results

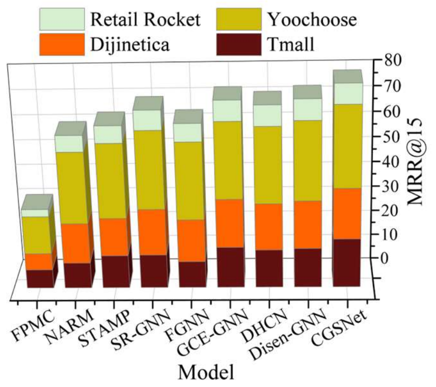


Figure 3: MRR@15 comparison results

III. B. 2) Model ablation experiments

The four main components in the CGSNet model established in this paper include neighborhood information representation learning (A), global session representation learning (B), self-supervised comparison learning (C), and model optimization (D), and we further investigate the effectiveness of each module in the CGSNet model by conducting ablation experiments. Specifically, several comparison variants of the CGSNet model are designed as follows:

(1) w/o A, the neighborhood information representation learning module is deleted and there is no gated graph neural network mechanism.

(2) w/o B, the global session representation learning module is deleted, and the session-aware attention mechanism does not exist.

(3) w/o C, the self-supervised comparison learning module is deleted and there is no neighborhood and global information exchange.

(4) w/o D, the model optimization module is deleted, and the self-supervised contrast learning task control parameters do not exist.

Table 1 shows the results of the ablation experiments of the CGSNet model.

From the table, we can observe that removing the above components consistently leads to performance degradation, which implies that all these components are important for the CGSNet model. Specifically, the performance of w/o A is lower than that of w/o C, indicating that the performance has been improved by directly incorporating temporal embedding in combination with spatial embedding for neighborhood information representation learning in the main conversational language model recommendation. Then, the decreasing trend is more pronounced for w/o B than w/o C. This phenomenon is consistent with our hypothesis that the spatial representation can be enhanced by obtaining implicit collaborative filtering information from a globally weighted session graph, which helps to compensate for the data sparsity problem of short-term sessions. In addition, by comparing with the standard CGSNet model and w/o C, it can be found that spatial-temporal contrast learning has significantly improved performance on both metrics. This suggests that regularized cross-view interactions in latent space by contrast learning can further enhance the session representation for the main prediction task.

Table 1: The results of the CGSNet model ablation experiment (%)

Model	Tmall		Dijinetica		Yoochoose		Retail Rocket	
	P@15	MRR@15	P@15	MRR@15	P@15	MRR@15	P@15	MRR@15
w/o A	36.12	16.04	54.23	16.11	65.31	27.15	20.51	4.63
w/o B	36.48	16.27	54.29	18.85	66.08	28.94	21.38	4.79
w/o C	37.65	17.32	54.68	19.32	68.42	29.26	22.84	5.91
w/o D	38.26	17.69	55.37	19.48	69.29	30.31	23.62	6.85
CGSNet	**39.51**	**19.51**	**58.94**	**20.16**	**73.54**	**32.29**	**24.79**	**7.94**

III. B. 3) Fusion Functions and Optimizers

In this paper, four different fusion functions are selected to analyze their impact on the recommendation performance of the CGSNet model.

(1) Summation function, which performs an element-by-element summation of two features to produce a new feature.

(2) Maximum pooling function, which performs an element-by-element comparison between two features and returns a new feature whose corresponding positions are all maxima.

(3) Splicing function, which splices two features by a specified dimension to obtain a new feature.

(4) Gated fusion function, which adaptively aggregates two features based on the relevant fusion function of the gated graph neural network.

Table 2 shows the comparison results of different fusion functions.

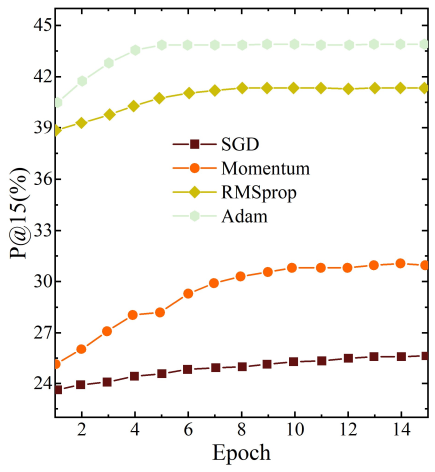
From the results in the table, it can be seen that on the Tmall dataset, although the splicing function has the best performance on the P@K metrics, the MRR@K metrics have the worst performance situation among the four fusion methods, and the summation function and the maximum pooling function generally do not perform as well as the gated fusion function on each metric. On the Dijinetica dataset, the summation function has the worst performance among the four fusion methods, and the gated fusion function has the next best performance. On the Yoochoose dataset, the gated fusion function has the best performance among the four fusion methods. In summary, the gated fusion function is better able to perform better on all three datasets, which may be due to the fact that the gated fusion function can be more flexible in adjusting the importance of the different inputs by deciding their contribution to the output based on the gating parameters of each input.

Table 2: Comparison results of different fusion functions (%)

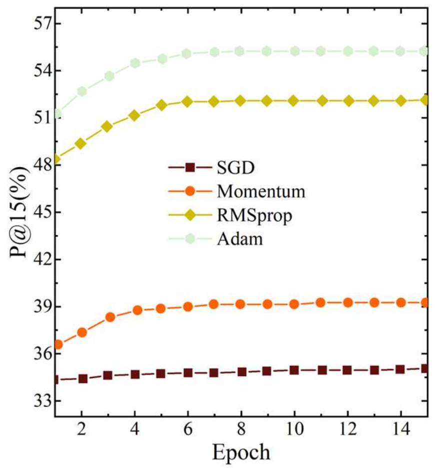
Methods	Tmall		Dijinetica		Yoochoose	
	P@15	MRR@15	P@15	MRR@15	P@15	MRR@15
Sum	41.83	19.82	37.26	19.42	58.14	31.94
Max	40.69	20.15	37.03	**19.78**	57.96	31.89
Concat	**42.57**	19.64	**37.49**	19.31	58.35	32.15
Ours	42.31	**20.46**	37.32	19.73	**59.03**	**33.28**

Optimizer is a key component in deep learning model training. In order to explore the effect of optimizers on the recommendation effect of conversational language model of CGSNet model, this paper selects SGD optimizer, Momentum optimizer, RMSprop optimizer, and Adam optimizer in turn to train this model. The Tmall and Dijinetica datasets are chosen to carry out the validation respectively, and Fig. 4 shows the model training results under different optimizers. Among them, Fig. 4 (a)~(b) shows the variation of P@K metrics on Tmall and Dijinetica datasets, respectively.

From the figure, it is easy to find that the SGD optimizer has the worst performance effect, which may be related to the fact that SGD only considers the current gradient in each update, which is easily affected by the data noise.The Momentum optimizer can be regarded as an improved version according to the SGD optimizer, which introduces a momentum term on top of the SGD and uses the historical gradient information to smooth the direction of the parameter update, which helps to reduce the shock during the gradient update. The RMSprop optimizer improves the recommendation accuracy substantially compared to the previous two optimizers, RMSprop is an adaptive learning rate optimization algorithm, which can update the model parameters more efficiently when training the model by dynamically adjusting the learning rate and the weighted average of the gradient squares. The Adam optimizer has the best recommendation effect, it combines the advantages between both RMSprop and Momentum, taking into account not only the momentum but also the variance of the gradient, which enables better adjustment of the learning rate to improve the accuracy of the model.



(a) Tmall



(b) Dijinetica

Figure 4: Model training results under different optimizers

III. B. 4) Model hyperparameter effects

(1) Effect of contrast learning parameters

The key role of contrast learning is to make the distribution of embedding vectors more uniform by discretizing the distance between items in the embedding space. In order to explore the influence of the temperature hyperparameter τ in contrast learning on the experimental results, this paper sets its value range to 0.01~1 and conducts experiments on two datasets, Tmall and Dijinetica. Figure 5 shows the performance variation brought about by different temperature coefficients on the two datasets. From the experimental results, it can be seen that the peak performance on the two datasets occurs when the values of τ are 0.35 and 0.65, respectively, and the P@15 index values are 44.69% and 35.11%, respectively. This is because in conversational language model recommendations, the similarity between different conversational languages is usually lower than that of commodities, and a relatively large temperature coefficient needs to be taken to make the conversational language distribution smoother. In e-commerce, the correlation between goods is usually large, and a relatively small temperature coefficient is required to make the project distribution more concentrated.

In addition, contrast learning can automatically extract the items related to the target items and enhance the model's understanding of the user's interests by training to obtain an effective denoising model. In the proposed CGSNet model, the hyperparameter β is utilized to control the degree of contrast learning. In order to investigate the effect of contrast learning on the overall performance of the CGSNet model. The range of hyperparameter is set to 0.001~0.01, experiments are conducted on Dijinetica and Tmall datasets respectively, and the performance of the model is evaluated under different magnitudes of contrast learning using P@15 and MRR@15, and the other parameters are taken as defaults and kept constant in the experiments. When co-optimized with the contrastive learning task, the performance of the CGSNet model for conversational language model recommendation is decently improved. When using smaller values for learning can promote both P@15 and MRR@15, but as the hyperparameter β becomes progressively larger, the performance of the CGSNet model decreases. This may be due to the fact that larger ones cause the model to overlearn, and useful information may be removed to form less accurate bootstrapping, making the model perform poorly.

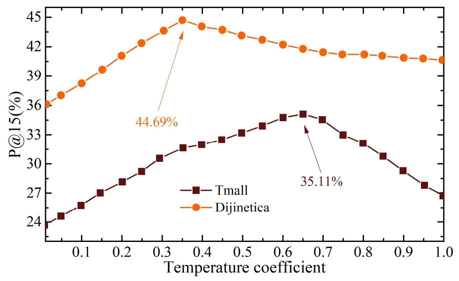
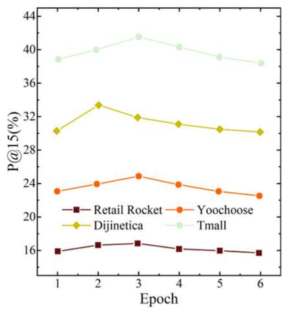


Figure 5: Performance variation of temperature coefficient

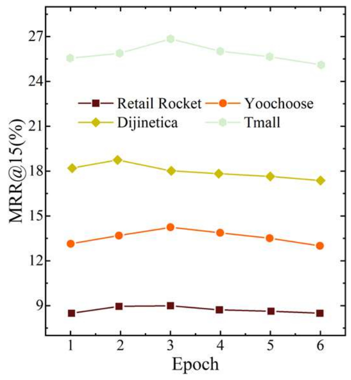
(2) Impact of the number of aggregation layers on modeling

When modeling global information, in order to obtain high-order session information, this paper aggregates three types of neighbor information with current session information in multiple layers. There are differences in the effect of different number of aggregation layers on the model. In order to verify that the aggregated multilayer module can enrich the current item representation and to study the effect of different number of training layers on the experimental performance, this paper sets the value of the number of aggregation layers k to take 1~6 respectively, and other parameters are kept uniform, and the experimental results on different datasets are shown in Fig. 6. Where Fig. 6 (a)~(b) shows the comparison results of P@15 and MRR@15, respectively.

The experimental results show that when k=3, which means aggregating the neighbor information of two hops with the current session, the best results are achieved on Tmall dataset, Dijinetica dataset and Yoochoose dataset. When k=2, the model has difficulty in capturing the complex item transition information. However, when k>3, too many stacked layers can lead to overfitting of the model, causing nodes to be difficult to distinguish, thus introducing too much noise making the model less effective.



(a) P@15



(b) MRR@15

Figure 6: The impact of the aggregation layer on the model

IV. Conclusions and outlook

IV. A. Conclusion

Using contrastive learning to optimize recommendation data followed by constructing language models, this paper focuses on the conversational perspective. Combining graph convolutional neural network and self-supervised contrastive learning, a CGSNet conversational language recommendation model is established. Four different types of datasets are selected to validate and analyze its recommendation performance.

The CGSNet model achieves certain improvement over the baseline model in all four datasets, and the P@15 and MRR@15 metrics on the Tmall dataset are improved by 15.89% and 19.11%, respectively, compared with the suboptimal model. And the roles of neighborhood information, global session and self-supervised contrastive learning in the model are crucial to improve the performance of conversational language models. Utilizing the self-supervised contrastive learning mechanism can fully enhance the user session data, which provides a guarantee for recommending items suitable for the user's interest preferences.

IV. B. Discussion

In this paper, specific solutions are proposed and implemented for the deficiencies in the recommendation algorithms of conversational language modeling, and certain results have been achieved. However, there are still places to be improved, and the outlook for future work is as follows:

(1) The contrastive graph self-attention networks used in this paper all start from the session perspective and construct positive and negative sample pairs based on the whole session interest. In the next step, we can consider constructing positive-negative sample pairs from a local perspective using local item representations to mine item information at a finer granularity and extract session interests.

(2) Further excavation can be done in negative sample sampling, difficult negative samples effectively promote the learning of data features by graph neural networks and improve the quality of data representation, the extraction of difficult negative samples is a difficult point, in the future, we can learn from the extraction methods of negative samples in other fields and migrate the use of negative samples, to explore more negative sample sampling methods, to further improve the performance of the model.

References

[1] Zhao, Z., Fan, W., Li, J., Liu, Y., Mei, X., Wang, Y., ... & Li, Q. (2024). Recommender systems in the era of large language models (llms). IEEE Transactions on Knowledge and Data Engineering.

[2] Chang, Y., Wang, X., Wang, J., Wu, Y., Yang, L., Zhu, K., ... & Xie, X. (2024). A survey on evaluation of large language models. ACM transactions on intelligent systems and technology, 15(3), 1-45.

[3] Bao, K., Zhang, J., Zhang, Y., Wenjie, W., Feng, F., & He, X. (2023, November). Large language models for recommendation: Progresses and future directions. In Proceedings of the Annual International ACM SIGIR Conference on Research and Development in Information Retrieval in the Asia Pacific Region (pp. 306-309).

[4] Chen, J., Liu, Z., Huang, X., Wu, C., Liu, Q., Jiang, G., ... & Chen, E. (2024). When large language models meet personalization: Perspectives of challenges and opportunities. World Wide Web, 27(4), 42.

[5] Lin, J., Dai, X., Shan, R., Chen, B., Tang, R., Yu, Y., & Zhang, W. (2025). Large language models make sample-efficient recommender systems. Frontiers of Computer Science, 19(4), 194328.

[6] Xu, L., Zhang, J., Li, B., Wang, J., Chen, S., Zhao, W. X., & Wen, J. R. (2024). Tapping the Potential of Large Language Models as Recommender Systems: A Comprehensive Framework and Empirical Analysis. ACM Transactions on Knowledge Discovery from Data.

[7] Wang, Z. (2024). Empowering few-shot recommender systems with large language models-enhanced representations. IEEE Access.

[8] Shang, F., Zhao, F., Zhang, M., Sun, J., & Shi, J. (2024). Personalized recommendation systems powered by large language models: Integrating semantic understanding and user preferences. International Journal of Innovative Research in Engineering and Management, 11(4), 39-49.

[9] Huang, S., Yang, K., Qi, S., & Wang, R. (2024). When large language model meets optimization. Swarm and Evolutionary Computation, 90, 101663.

[10] Wu, L., Zheng, Z., Qiu, Z., Wang, H., Gu, H., Shen, T., ... & Chen, E. (2024). A survey on large language models for recommendation. World Wide Web, 27(5), 60.

[11] Zheng, B., Hou, Y., Lu, H., Chen, Y., Zhao, W. X., Chen, M., & Wen, J. R. (2024, May). Adapting large language models by integrating collaborative semantics for recommendation. In 2024 IEEE 40th International Conference on Data Engineering (ICDE) (pp. 1435-1448). IEEE.

[12] Heinrich, B., Hopf, M., Lohninger, D., Schiller, A., & Szubartowicz, M. (2021). Data quality in recommender systems: the impact of completeness of item content data on prediction accuracy of recommender systems. Electronic Markets, 31, 389-409.

[13] Nayak, A., Božić, B., & Longo, L. (2023). Data quality assessment and recommendation of feature selection algorithms: An ontological approach. Journal of Web Engineering, 22(1), 175-196.

[14] Ko, H., Lee, S., Park, Y., & Choi, A. (2022). A survey of recommendation systems: recommendation models, techniques, and application fields. Electronics, 11(1), 141.

[15] Liang, F. Y., Xi, W. D., Xing, X. X., Wan, W., Wang, C. D., Chen, M., & Guizani, M. (2024, December). Contrastive Learning for Adapting Language Model to Sequential Recommendation. In 2024 IEEE International Conference on Data Mining (ICDM) (pp. 251-260). IEEE.

[16] Bai, Z., Zheng, Y., Yang, P., Liu, S., Zhang, Y., & Chang, Y. (2025). DCRLRec: Dual-domain contrastive reinforcement large language model for recommendation. Information Processing & Management, 62(4), 104140.

[17] Woo Seok Kim,Seongho Lim,Gun Woo Kim & Sang Min Choi. (2025). Extracting Implicit User Preferences in Conversational Recommender Systems Using Large Language Models. Mathematics,13(2),221-221.

[18] Aron Mohammadi,Jonathan Tsang,Xiao Huang & Richard Kearsey. (2025). Convolutional neural network based methodology for flexible phase prediction of high entropy alloys. Canadian Metallurgical Quarterly,64(2),431-443.

[19] Qi Kankan,Guo Yanchi & Zhou Jian. (2025). Self-supervised learning fabric defect segmentation using anomaly generation. International Journal of Clothing Science and Technology,37(2),340-354.

[20] Yang Rui,Jia Shi,Wang Houliang,Yan Yinglong,Zhang Pengfei & An Zenghui. (2025). Dilated dynamic supervised contrastive learning framework for fault diagnosis under imbalanced dataset conditions. Proceedings of the Institution of Mechanical Engineers,239(7),2626-2636.

[21] Houde Dai,Yiyang Huang,Liqi Zhu,Haijun Lin,Hui Yu,Yuan Lai & Yuxiang Yang. (2025). Battery state-of-health estimation based on random charge curve fitting and broad learning system with attention mechanism. Journal of Power Sources,636,236544-236544.