ITC 1/53

Information Technology and Control
Vol. 53 / No. 1 / 2024
pp.146-159
DOI 10.5755/j01.itc.53.1.33495

RWESA-GNNR: Fusing Random Walk Embedding and Sentiment Analysis for Graph Neural Network Recommendation

Received 2023/02/26 | Accepted after revision 2023/09/04

HOW TO CITE: Gu, J., Xu, Y., Liu, W. (2024). RWESA-GNNR: Fusing Random Walk Embedding and Sentiment Analysis for Graph Neural Network Recommendation. *Information Technology and Control,* 53(1), 146-159. https://doi.org/10.5755/j01.itc.53.1.33495

RWESA-GNNR: Fusing Random Walk Embedding and Sentiment Analysis for Graph Neural Network Recommendation

Junlin Gu, Yihan Xu, Weiwei Liu

College of Computer, Jiangsu Vocational College of Electronics and Information, Huai'an 223003, China

Corresponding author: junlin82@qq.com

A graph neural network-based recommendation system treats the relationship between user items as a graph, and achieves deep feature mining by modelling the graph nodes. However, the complexity of the features of graph neural network-based recommendation systems brings poor interpretability and suffers from data sparsity problems. To address the above problems, a graph convolutional neural network recommendation model (RWESA-GNNR) based on random walk embedding combined with sentiment analysis is proposed. Firstly, a random walk-based matrix factorization is designed as the initial embedding. Secondly, the user and item nodes are modelled using a convolutional neural network with an injected attention mechanism. Then, sentiment analysis is performed on the review text, and attention mechanism is introduced to fuse text sentiment features and semantic features. Finally, node features and text features are aggregated to generate recommendation results. The experimental results show that our proposed algorithm outperforms traditional recommendation algorithms and other graph neural network-based recommendation algorithms in terms of recommendation results, with an improvement of about 2.43%-5.75%.

KEYWORDS: recommendation system, graph neural network, random walk, sentiment analysis, data mining.

1. Introduction

With the advancement of productivity levels and the rapid development of Internet technology, mankind has crossed over into the information age. However, the rapid development of the information age has not only brought a wealth of information to the people, but also caused an explosion of information. The redundancy of useless information has led to a relative lack of information that users want to access and the phenomenon of information overload. Although people can filter some of the information and access the content through search engines, it is becoming increasingly difficult for people to choose their favourite content. As a result, the concept of Recommendation Systems (RS) has been proposed [3]. RS obtains hidden information through the user's historical behavior, interest preferences and other features, and then applies recommendation algorithms to generate a list of items that may be of interest to the user [6, 25].

In real applications of RS, most of the data inherently have a graph structure, and this inherent data feature makes it necessary to consider complex graph data relationships when making recommendations. Therefore, with the research and development of Graph Neural Networks (GNNs), more and more researchers are using GNNs for RS to extract node information about the associations between users and items. Jiang et al. [10] used GNNs as a basis to improve recommendation performance by capturing latent features using an end-to-end approach. Wu et al. [23] modelled user sequence behavior as graph-structured data and used GNNs to capture the complex transformational relationships between items. Song et al. [20] modeled contextually relevant social influences using graph attention networks and demonstrated the effectiveness of the model by testing on real datasets such as Facebook. A technique closely related to GNNs is network embedding, which expresses the hidden connections between items in an aggregated way to obtain an embedding representation. Commonly used network embedding methods are deep learning based graph embedding, random walk based graph embedding and so on. Ying et al. [26] combined random wandering and GNNs to generate node embeddings that contain graph structure as well as feature information of the nodes. Zhou et al. [30] proposed an algorithm of random walk combined with matrix decomposition to obtain the embedding vector of social relations. Liu et al. [12] proposed a random walk approach to build heterogeneous networks as a way to generate better recommendations. Canturk et al. [1] used random walk for location recommendation on model subgraphs. Zhang et al. [28] effectively captured the global and local structure of the network by an improved random walk strategy.

However, traditional GNNs methods often suffer from limitations in node representation learning, inability to make full use of user behavior and social network information, and data sparsity. At the same time, review text, which contains attribute information of users and items, can establish more objective evaluation and sentiment features for users and items, which can improve the interpretability of the RS while alleviating data sparsity. Therefore, how to design a node embedding method and combine node analysis with text analysis to further improve the accuracy and efficiency of GNNs recommendation algorithms is the focus of this paper.

In summary, we propose a GNN recommendation model for fusing review texts, called RWESA-GNNR, with the following main contributions:

1. We design an original framework which fuses interaction graph node features with review text features to achieve better recommendations.
2. We propose a random wander-based matrix factorization that enables nodes to feature the topology of a graph through the embedding of self-covariance similarity. At the same time, by combining with a specific aggregation function, the distribution of dissimilarity in the graph is captured and more potential connections between users and items are mined.
3. We propose to mine useful sentiment information in review texts by analyzing the sentiment polarity of the review texts, combining the attention mechanism, and combining sentiment weights with the semantics of the review texts to obtain personalized sentiment features of users and items.
4. We conduct experiments on the model on three publicly available Amazon datasets. The experimental results demonstrate that the proposed method is better than existing recommendation methods, with improved recommendation results.

2. Proposed Method

In this section, we describe the proposed RWESA-GNNR, the structure of which is shown in Figure 1. The RWESA-GNNR consists of four modules: (1) a user-item node feature extraction module for extracting features of nodes in the interaction graph; (2) a review text feature extraction module for extracting sentiment polarity features from the review text; (3) a sentiment feature fusion module for fusing the semantic features of the review text with the sentiment polarity features; (4) a prediction module, which performs recommendation prediction based on the final fused features.

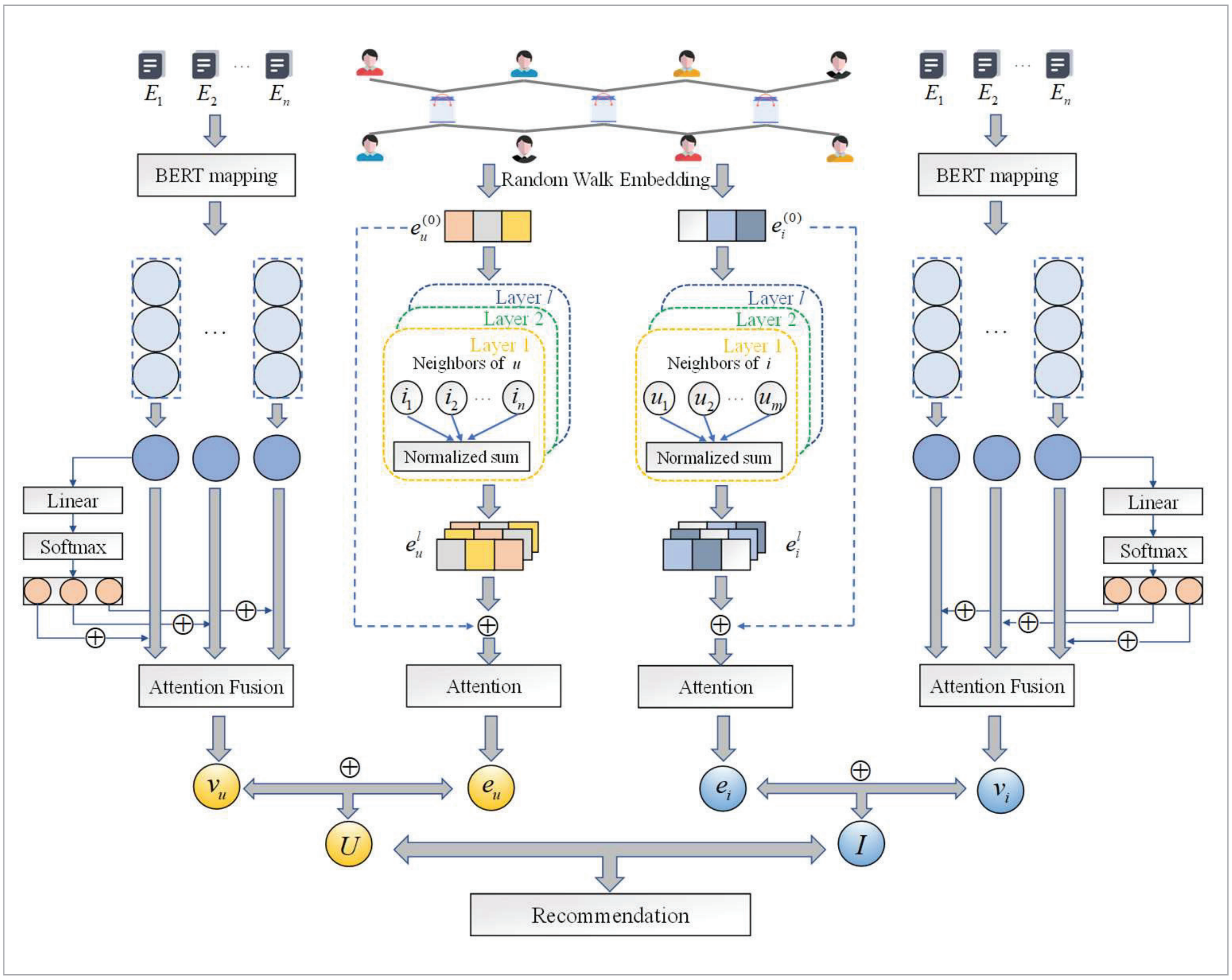
2.1. User-item Node Feature Extraction Module

The module consists of two layers, the first generating an initial embedding vector from the user-item interaction graph using a random walk-based matrix factorization, and the second aggregating the neighboring nodes of each node using a Graph Convolutional Neural Network (GCN) to obtain a nodal feature representation of the graph.

Considering that interaction graphs are interactions between users and items, we model the interactions as weighted undirected graphs $G = (V, E)$, where V denotes the set of nodes, including a total of n nodes

Figure 1

The overall framework of the proposed model



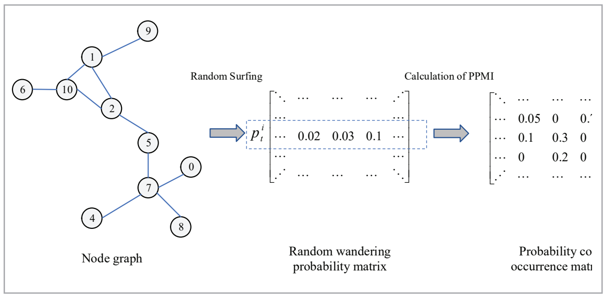
for users and items, denoting the number of users as a and the number of items as b; E denotes the set of edges, representing the m interactions between users and items, representing the interaction information of the graph through the symmetric adjacency matrix $A \in \mathbb{R}^{n \times n}$.

2.1.1. Node Initial Embedding Layer

This layer generates the initial embedding vector from the user-item interaction graph using a random wandering matrix factorization. For large undirected network graphs such as recommendation data, the interaction information is very unevenly distributed and usually follows a power-law distribution [11]. Random walks enable the graph to have a topological structure that can well overcome the power-law distribution.

We define node u and node v. The edge with weights connecting the two nodes is A_{uv}. If node u and node v interact, then $A_{uv} = 1$, otherwise $A_{uv} = 0$. The probabilistic covariance matrix of node pairs based on random wandering is shown in Figure 2.

Figure 2
Generating probability co-occurrence matrices based on random wandering



Since the random walk process is carried out randomly, the value of the next node is only related to the current node and has no relation to the previous node. We assume that the current point in time is t and the position is node u. The next random wandering result to the next node v is only related to the position of the current node u and not to the state (position) before time t. Therefore, the random wandering process is a class of Markov processes [4]. The sequence of nodes obtained after this process is a Markov chain over the set V of nodes. For a Markov chain, a standard random walk on a connected bipartite graph traverses the entire graph and has a unique smooth distribution $\pi \in \mathbb{R}^n$.

When using standard random wandering in an undirected weighted graph, the probabilities from node u and node v are only related to node u itself, so that the transfer probability obtained is proportional to the edge weight A_{uv}, as shown in Equation (1).

$$p(x(t+1) = v \mid x(t) = u) = \frac{A_{uv}}{\deg(u)}, \tag{1}$$

where $x(t) \in V$ is the node wandered to at time t; $\deg(\cdot)$ is a function of the degree of node seeking.

We denote the transfer probabilities between all pairs of nodes as the transfer matrix $M \in \mathbb{R}^{n \times n}$, as shown in Equation (2):

$$M = D^{-1} A. \tag{2}$$

For a connected bipartite graph, a random walk traverses the entire graph and has a unique smooth distribution $\pi \in \mathbb{R}^n$. For each node u, the distribution is shown in Equation (3):

$$\pi_u = \frac{\deg(u)}{\sum_v \deg(v)}. \tag{3}$$

The root of the similarity function based on random wandering is based on the probability of a wandering being co-visited for a node. The node similarity metric is essentially a function that maps the probability of co-visitation of a node pair to the concept of topological similarity. Two nodes are similar to each other if the function has a large positive value, and they are different if they have a large negative value. We use self-covariance as the similarity measure function, which has been shown to give better embedding results in recommendation tasks [9].

We define the self-covariance $R_{uv}(\tau)$ as the covariance between u and v over time t, as shown in Equation (4):

$$\begin{aligned} R_{uv}(\tau) &= \operatorname{cov}(X_u(t), X_v(t+\tau)) = \\ & E[(X_u(t) - E[X_u(t)])(X_v(t+\tau) - E[X_v(t+\tau)])] \end{aligned}, \tag{4}$$

where the value of the covariance $R_{uv}(\tau)$ is a linear measure of the joint variability of the probability of a random visit to nodes u and v at time t, and is a formal expression for the likelihood of two nodes being visited simultaneously.

The probability π of iterating and starting is shown in Equation (5).

$$R_{uv}(\tau)=\pi_u p(x(t+\tau)=v|x(t)=u)-\pi_u\pi_v \tag{5}$$

where $R_{uv}(\tau)\in[-\pi_u\pi_v,\pi_u(1-\pi_v)]$. We express it in the form of a matrix as shown in Equation (6):

$$R(\tau)=\prod M^{\tau}-\pi\pi^T. \tag{6}$$

The goal of the embedding algorithm is to generate a representation that maintains a given vector of similarity measures, subject to the conditions of Equation (7):

$$\begin{aligned} U^*&=arg\min\sum_{u,v}(u_u^T u_v-R_{uv})^2 \\ &=arg\min\left\|UU^T-R\right\|_F^2 \end{aligned}, \tag{7}$$

where $u_u^T u_v$ captures the similarity in the embedding space; R is the similarity matrix; and $\|\cdot\|_F^2$ is the Fibonacci parametrization.

Because R is a symmetric but extremely sparse matrix, its complexity will reach $o(n^3)$ if singular value decomposition is used, and it will drop to $o(nd^2+mdr)$ if it is decomposed by a scalable factorize method. Therefore, we use the ARPACK method [19] to capture the global information by deriving the eigenvalues and eigenvectors for the probability matrix to obtain the initial embedding $e^{(0)}\in\mathbb{R}^d$ for the final input GCN.

2.1.2. Node Feature Extraction Layer

This layer uses a GCN to extract features on the interacting nodes to obtain a node representation of the user and the item.

Firstly, we couple the initial embedding A obtained by random wandering with the dot product, which will have better performance in capturing the full graph heterogeneity distribution. The aggregation function is used to update the content of this node by aggregating information from neighboring nodes, using a non-linear activation function to model more non-linear relationships in the data. To better capture node features, we use the Bi-Interaction aggregator [22] to fully capture the feature information embedded in the initial vector of the transfer. In summary, we express the whole process as Equation (8):

$$\begin{aligned} e_u^l &= LeakyReLU(W_0(e_u^{l-1}+e_{N_{(u)}}^{l-1})) \\ &+LeakyReLU(W_1(e_u^{l-1}\odot e_{N_{(u)}}^{l-1})) \\ e_i^l &= LeakyReLU(W_0(e_i^{l-1}+e_{N_{(i)}}^{l-1})) \\ &+LeakyReLU(W_1(e_i^{l-1}\odot e_{N_{(i)}}^{l-1})) \end{aligned}, \tag{8}$$

where W_0 and W_1 are trainable weight matrices; l is the number of layers of the GCN; e_u is the embedding vector of user u and e_i is the embedding vector of item i; $N(U)$ is the set of neighboring nodes of user u and $N(i)$ is the set of neighboring nodes of item i; $\odot$ is the dot product operation; and $LeakyReLU(\cdot)$ is the activation function we use.

Secondly, considering that different layers of information have different weights of influence on the content of nodes, we introduce a self-attention mechanism to model the importance of each layer of embedding, and each layer of embedding is calculated as shown in Equation (9):

$$\alpha_k^* = W_a^T\tanh(W_2 e_j^k+b_0)+b_1, \tag{9}$$

where e_j^k is the layer k embedding of node j. W_a, W_2, b_0 and b_1 are learnable parameters, and α_k^* is the scoring function for layer k.

Third, we normalized the attention ratings to obtain the attention weights for each layer of the embedding, as shown in Equation (10):

$$\alpha_k = softmax(\alpha_k^*)=\frac{\exp(\alpha_k^*)}{\sum(\alpha_k^*)}. \tag{10}$$

Finally, we combine the embedding expressions of each layer weighted according to the weights to obtain a final representation of the association between nodes $e_u\in\mathbb{R}^d$ and $e_i\in\mathbb{R}^d$, as shown in Equation (11):

$$\begin{aligned} e_u &= \sum_{k=0}^{l}\alpha_k e_u^k \\ e_i &= \sum_{k=0}^{l}\alpha_k e_i^k \end{aligned}. \tag{11}$$

2.2. Review Text Feature Extraction Module

The module consists of two layers, the first for word embedding of the review text using the encoder and the second for sentiment polarity analysis of the review text.

2.2.1. Word Embedding Layer

This layer changes the review text from non-machine-recognizable words to an actionable set of embedding vectors via an encoder, facilitating machine learning for feature extraction.

Firstly, we use Bidirectional Encoder Representations from Transformers (BERT) for encoding, this is due to the fact that BERT is composed of multiple Transformer overlays, which can solve the problem of multiple meanings of words; at the same time, BERT can selectively use information from all levels, so that the multi-layered features of words can be exploited [24]. BERT is to input the whole review together, and then perform sentence division and padding, sentences that are less than the length will be filled, while sentences that are too long will be truncated. Specifically, we connect the results of the token, segment and position layers in BERT by element to obtain a representation of all embedding vectors about the input review text, as shown in Equation (12):

$$\begin{aligned} x^i &= x^i_{token} + x^i_{segment} + x^i_{position} \\ E &= (x^i_1, x^i_2, \ldots, x^i_n) \end{aligned}, \tag{12}$$

where x^i is the vector of the ith review text embedding representation, where x_1 is the initial label vector in BERT; and E is the initial embedding vector of the review text generated after the entire input of the ith review text.

It is worth mentioning that, unlike the often used text processing models such as one-hot encoding [17], word2vec [5] and ELMo [14], BERT uses a multi-headed attention mechanism to assist in capturing sequential information in the input content, and the Transformer model allows BERT to capture contextual features more comprehensively, regardless of the length of the input text. BERT uses a Word Piece [21] approach for word embedding, where a word is split into several parts according to the structure of the root affix, allowing different forms of the same word to be recognized. This is sufficient for BERT's own word list of over 30000 words.

Then, we input the initial representation vector A of the review text into the multilayer Transformer to extract the contextual relevance and enrich the feature vector representation of the review text. The final result of the text embedding process is thus obtained, as shown in Equation (13):

$$T_1 = Trm(E), T_2 = Trm(T_1), \ldots, T_l = Trm(T_{l-1}), \tag{13}$$

where $Trm(\cdot)$ is a Transformer encoding calculation; T_l is the output of the Transformer encoding block at layer l, and sometimes the embedded representation of all tokens for the next Transformer encoding module; $l \in [1, L]$, where L is the layer of Transformer encoders in BERT; and E is the final output representation of the BERT model, a sequence of token representations.

2.2.2 Sentiment Polarity Analysis Layer

The layer performs sentiment analysis on the review text features embedded by BERT through an activation function.

Since the BERT model is already a deep neural network with a stack of multi-layer Transformer encoder blocks, it already captures sufficient semantic information for the extraction of semantic information in sentences. Therefore, after training with multiple layers of stacked Transformer encoders, we can already obtain a certain degree of semantic information about the entire comment text. We input the embedding representation E of each review text into the activation function for sentiment polarity analysis, and predict the sentence-level sentiment polarity probability in the whole review, as shown in Equation (14):

$$p(E) = softmax(W_3 E + b_2), \tag{14}$$

where $p(E)$ is the probability of semantic sentiment polarity analysis for the whole review text; W_3 and b_2 are the trainable weight matrices; and D is the activation function we use.

2.3. Sentiment Feature Fusion Module

The module uses an attention mechanism to merge semantic features of review text with sentiment polarity features.

We dot product the node feature vectors of users and items and use them as queries in the attention mechanism, define the initial token in the text E_j of review j as c_j, and concatenate c_j with the sentiment polarity feature p_j of that review as the KEY-VALUE pair in the attention mechanism. We calculate a rating for each review based on the attention mechanism, as shown in Equation (15):

$$\begin{aligned}\beta_j^* &= W_b^T Relu(W_k(c_j \oplus p_j) \\ &+ W_q(e_u \odot e_i) + b_3) + b_4\end{aligned}, \quad (15)$$

where W_b, W_k, W_q, W_3 and W_4 are learnable parameters; β_j^* is the attention rating function for each review, and the attention value for a weighted combination of all reviews for the same user or item is calculated as shown in Equation (16):

$$\beta_j = Softmax(\beta_j^*) = \frac{\exp(\beta_j^*)}{\sum \exp(\beta_j^*)}. \quad (16)$$

Finally, the sentiment feature embedding of the user or item is calculated based on the obtained attention coefficient, as shown in Equation (17):

$$v = \beta_j(c_j \oplus p_j). \quad (17)$$

2.4. Prediction Module

The module uses connects interaction node features of user items with sentiment features for a final predicted match rating.

We connect the final interaction node features obtained from Equation (11) with the final sentiment features obtained from Equation (17) to obtain the final representation *U* of the user and the final representation *I* of the item, as shown in Equation (18):

$$\begin{aligned}U &= e_u \oplus v_u \\ I &= e_i \oplus v_i\end{aligned}. \quad (18)$$

Finally, we predict the matching ratings of users and items by calculating the inner product of their final representations, as shown in Equation (19):

$$\hat{y}_{ui} = U^T \cdot I. \quad (19)$$

2.4. Model Objective Function

To predict the interactions between users and items, we chose to train and optimize the model with the BPR loss function, a pairwise optimization method that assumes that observed interactions better reflect user preferences and therefore yield higher predictive values than unobserved interactions [7]. This objective function is defined as shown in Equation (20):

$$Loss = \sum_{(u,i,j)\in o} -\ln\sigma(\hat{y}_{ui} - \hat{y}_{uj}) + \lambda\|\theta\|_2^2, \quad (20)$$

where $o = \{(u,i,j) | (u,i) \in R^+, (u,j) \in R^-\}$ is the pairwise training data, R^+ is the observed interactions, and R^- is the unobserved interactions; $\sigma(\cdot)$ is the activation function, we choose the sigmoid function; $\theta = \{W_0, W_1, W_2, W_3, W_q, W_k, b_0, b_1, b_2, b_3, b_4\}$ is the set of all trainable parameters of the model; and λ controls the L2 regularization strength to prevent overfitting.

3. Experimentation and Analysis

In this section, we conduct experiments on the Amazon public dataset, which consist of parameter optimization experiments, performance analysis experiments and ablation experiments to confirm the effectiveness of RWESA-GNNR in a variety of ways.

3.1. Datasets

The Amazon dataset is one of the most widely used datasets in RS, with massive data support for our experiments [13]. Therefore, we selected three datasets with review texts from the Amazon dataset as our experimental datasets, namely Musical Instruments (MI), Beauty, and Amazon-CDs (CDs), whose number of users, items, interactions, and sparsity are shown in Table 1. To ensure feasibility and fairness, we randomly divided each dataset into training, testing, and validation sets in a 7:2:1 ratio. In the training set, we treated each user-item interaction as a positive example and then used a negative sampling strategy to match it with a negative item that the user had not previously interacted with. Next, we tuned the optimal parameter values on the validation set. Finally, we evaluated the model's performance on the testing set.

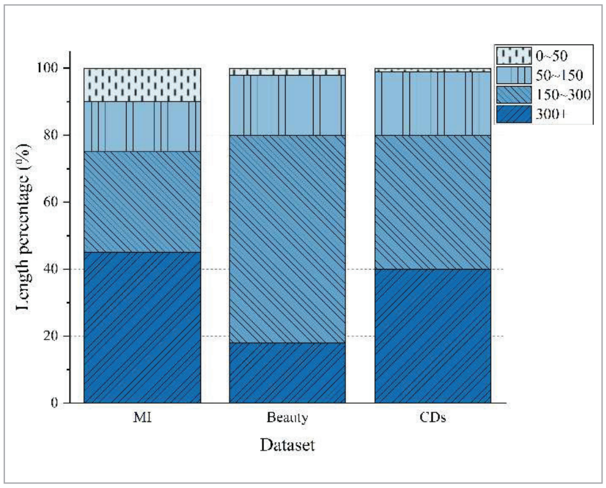
As can be seen from Table 1, although the data for each sample differed considerably, these datasets were sufficient to train and validate the proposed model due to the large enough data volume. In addition, the sparsity of each dataset is above 99%, which illustrates the significance of introducing item text features to alleviate sparsity. As our model is based on the idea of using fused review text data as an additional source of features to alleviate the data sparsity problem in rec-

Table 1
Datasets details

Dataset	Number of users	Number of items	Number of interactions	Data sparsity
MI	1429	900	10261	99.21%
Beauty	22335	12100	198502	99.93%
CDs	13573	8645	222042	99.81%

ommender systems, the length of the review text was counted to illustrate the information contained in the review text. The review lengths and their distribution for each dataset are shown in Figure 3.

Figure 3
Review length distribution



3.2. Experimental Setup

3.2.1. Evaluation Metrics

The evaluation strategies commonly used in RS to assess the effectiveness of recommendations include Accuracy, Recall, Precision and F1 values. In order to reflect the most realistic evaluation of effectiveness in RS, Recall is chosen in this paper to evaluate recommendation results. Recall is measured by calculating the number of correctly predicted positive cases as a proportion of all actual positive cases, which highlights the ability of the algorithmic model to detect interactions between users and unknown items [15]. At the same time, the prediction of recommendations is essentially a regression problem [27], and in order to better assess the effectiveness of the model implementation, we use Root Mean Square Error (RMSE) for evaluation. In order to evaluate the TOP-K effect of the recommendations in the experiments, we set the evaluation metric as Recall@K, RMSE and set K=20, i.e. to evaluate among the top 20 recommendation results, calculated as shown in Equations (21)-(22):

$$Recall=\frac{TP}{TP+FN} \tag{21}$$

$$RMSE=\sqrt{\frac{1}{|Z|}\sum_{u\in Z,i\in Z}(\hat{y}_{ui}-y_{ui})^2}\ . \tag{22}$$

Recall can be interpreted as the likelihood of the answer distribution in the confusion matrix, which is shown in Table 2, where *TP* is the number of predicted positive cases that are actually positive, *FN* is the number of predicted negative cases that are actually positive, *FP* is the number of predicted positive cases that are actually negative, *TN* is the number of predicted positive cases that are actually positive, and Recall value increases with the model's precision. *Z* is the number of instances in the dataset, $\hat{y}_{ui}$ is the predicted rating of user *u* for item *i*, and y_{ui} is the actual rating of user *u* for item *i*. The smaller the RMSE value, the lower the prediction error of the model, and the higher the prediction accuracy.

Table 2
Confusion matrix

Predicted results	True results		Total
	Yes (P)	No(N)	
Yes	TP	FN	P(Actual is yes)
No	FP	TN	P(Actual is No)

3.2.2. Baselines

We have divided the baselines into three categories: recommendation methods that use only user-item interaction information (BPRMF and PinSage), recommendation methods that incorporate review text (DeepCoNN and NARRE) and recommendation methods based on GNNs (LightGCN and HA-GNNN).

1 BPRMF [16]: A method based on optimizing matrix factorization using BPR, which only uses user-item interaction data as recommendation information.

2 PinSage [26]: A method that uses a sampling strategy based on random walks to perform graph convolution operations on local subgraph nodes, improving the problem of losing distant nodes in graph convolution recommendation.

3 DeepCoNN [29]: A method that uses two parallel CNNs to extract text features from reviews, and finally uses FM to predict ratings.

4 NARRE [2]: Similar to DeepCoNN, this method also uses two parallel CNN to extract text features from reviews. Additionally, it introduces an attention mechanism to distinguish the influence of different comments.

5 LightGCN [8]: A method that uses GCN to model the high-order connectivity between users and items, and simplifies the redundant parts of bipartite GCN.

6 HA-GNNN [18]: This method uses self-attention graph neural networks to capture the dependency relationships between items, and uses soft attention mechanisms to learn high-order relationships in the graph. Finally, it uses fully connected layers to update item embeddings.

3.2.3. Parameter Settings

For all the baselines, we followed the hyperparameter settings described in their respective papers. For BPRMF, we varied the number of latent factors in the range of {50,100,200,300}. For PinSage, we selected the number of random walks to be {1,2,3} since longer random walks may not be beneficial. For DeepCoNN, we set the number of convolutional filters to be 100 and the number of convolutional layers to be 3. For the remaining models, we referred to the authors' descriptions and settings for further details.

For the matrix factorization part of the initial node embeddings in RWESA-GNNR, we followed previous work that has shown that the length of Markov random walks does not affect the final node embedding quality [9]. Thus, we set the Markov length to be 3 for each walk.

For the network part of RWESA-GNNR, we used the Adam optimization algorithm to update model parameters with a learning rate of 0.002. To avoid overfitting, we applied L2 regularization with a parameter value of 1e-6.

For the BERT part of RWESA-GNNR, we used the official BERTbase version for text processing. A BERTbase model contains 110 million learnable parameters, which is sufficient for our experimental requirements. For the text length issue in the review text, we padded the text with zeros for texts that are shorter than the required length and truncated texts that exceed the required length. For sentiment analysis tasks based on BERT embeddings, we used the parameters shown in Table 3.

Table 3
BERT Parameter Setting

Parameter	Setting
Max sequence length	128
Initial learning rate	2e-5
Attention dropout rate	0.1
Activation function	Gelu
Hidden layer dropout rate	0.1
Embedding output dimension	768
Layer normalization parameter	1e-12
Max text length	512
Optimization algorithm	Adam W
Max sequence length	128

3.3. Experimental Results and Comparisons

3.3.1. Overall Result Comparison

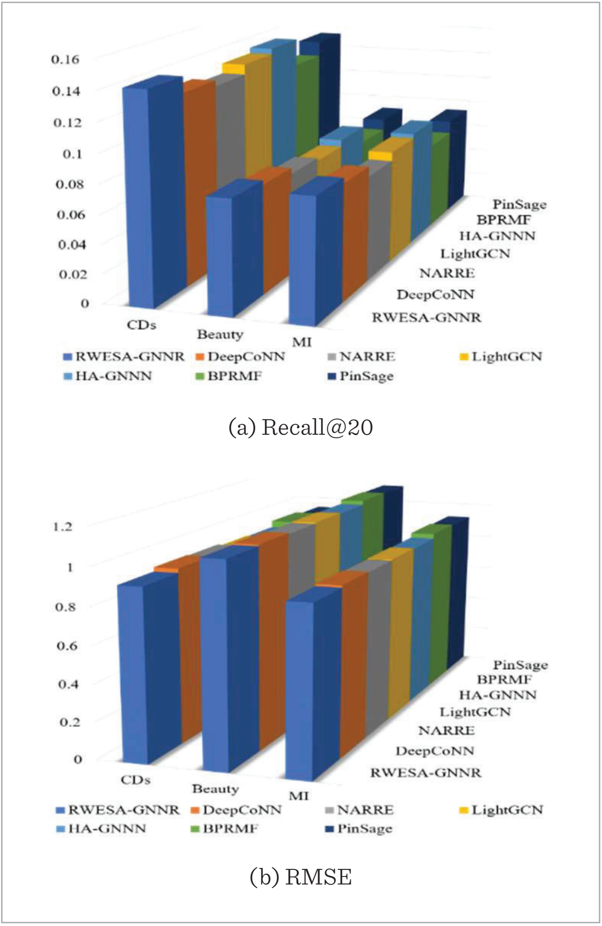
We conducted experiments under the optimal parameters and compared the Recall@20 and RMSE metrics of each model under the optimal parameters by setting the optimal parameters of each model according to the corresponding reference. The results are shown in Table 3. The bold data is the best performance in the same group comparison experiment, and the laicized data is the second best performance in the same group comparison experiment. The Improved value represents the growth rate of the best performance compared to the second best performance. As shown in Table 4, the RWESA-GNNR model proposed in this paper performed the best overall, which is consistent with our expectations.

To provide a more intuitive analysis of the effectiveness of each model, we show histograms of each model on the three datasets in Figure 4.

Table 4
Overall Performance Comparison of Compared Models

	MI		Beauty		CDs	
	Recall@20	RMSE	Recall@20	RMSE	Recall@20	RMSE
BPRMF	0.063	0.956	0.062	1.101	0.119	0.924
PinSage	0.072	0.939	0.070	1.096	0.130	0.916
DeepCoNN	0.080	0.939	*0.075*	1.085	0.132	0.904
NARRE	0.073	0.935	0.071	*1.078*	0.128	0.899
LightGCN	0.079	0.923	0.069	1.083	0.135	0.902
HA-GNNN	*0.082*	*0.919*	0.074	1.079	*0.139*	*0.898*
RWESA-GNNR	**0.084**	**0.883**	**0.078**	**1.016**	**0.143**	**0.872**
Improved	2.439%	3.917%	4%	5.751%	2.89%	2.895%

Figure 4
Comparison histogram of performance



(a) Recall@20

(b) RMSE

Firstly, the two evaluation metrics of the recommendation methods (BPRMF and PinSage) that only use user-item interaction information are the worst. BPRMF only utilizes the user-item interaction information and completes the information by matrix factorization. However, the user interaction data provided by RS is extremely sparse, and the matrix completion method based solely on interaction information only partially completes the data information, resulting in limited impact on the final recommendation effect. PinSage also shows that matrix factorization based solely on interaction information can only linearly represent the interaction features between users and items, and its capturing effect on the complex nonlinear relationship between users and items is not obvious.

Secondly, the two evaluation metrics of the recommendation methods (DeepCoNN and NARRE) that combine review text are both better than those of BPRMF and PinSage, which confirms that the utilization of review text can effectively alleviate the data sparsity problem in RS and also proves the significance of our proposed model. However, there are still some drawbacks of this type of method. DeepCoNN only extracts textual information from review without considering the sentiment information expressed by users in reviews. NARRE transforms the correlation between users and items into attention distribution rather than simple weights, which makes it difficult to understand why certain items are recommended to a certain user.

Then, the recommendation methods based on GNNs (LightGCN and HA-GNNN) have achieved the results of the first two types of methods, which proves the outstanding performance of GNNs in capturing high-order relationships. Specifically, LightGCN simplifies the embedding process by removing the nonlinear activation and feature transformation but does not consider the importance of the embedding of each node, which may lead to information limitations. HA-GNNN uses the attention mechanism to learn hidden features and uses fully connected layers to learn the representation of multimodal features, which has achieved good results in extracting node features by using graph neural networks alone.

Finally, our proposed RWESA-GNNR performs better than other baselines on each dataset. This is because we use the matrix factorization method based on random walk for initial embedding, which enables the nodes to have better topological structure features. By combining with specific aggregation functions, we can capture the heterogeneity distribution in the graph and explore deeper potential features. Meanwhile, we not only consider the semantic features of comment text but also the sentiment information, which provides weight reference for the extraction of features from comment text by users or items, and adds interpretability to the recommendation system's results.

3.3.2. Ablation Result Comparison

To further validate the effectiveness of RWESA-GNNR, we did RMSE ablation experiments for the key parts of the model - matrix factorization part, sentiment feature part, text feature part and GCN attention mechanism part, and the results are shown in Table 5. Where, Case_MF denotes the model that ablates random wandering in favor of unique thermal coding as the initial embedding, Case_SF denotes the model that ablates sentiment features, Case_TF denotes the model that ablates sentiment features, and Case_GA denotes the model that ablates the attention mechanism in the GCN.

First, the model that eliminates the random walk form has the most significant reduction in effectiveness, due to the fact that most GCN-based recommendation algorithms aggregate surrounding neighboring nodes through multi-layer convolution, which does not provide a comprehensive grasp of global information. Our proposed random walk-based graph embedding method is a node vector representation based on the topological similarity of the graph, which brings exactly the global information to the node features. Therefore, our proposed random-walking-based embedding method is the most effective.

Table 5
Results of RMSE ablation experiments with RWESA-GNNR

	MI	Beauty	CDs
Case_MF	0.933	1.121	0.975
Case_SF	0.921	1.096	0.976
Case_TF	0.923	1.083	0.923
Case_GA	0.920	1.099	0.924
RWESA-GNNR	0.883	1.016	0.872

Secondly, the sentiment feature module includes semantic and sentiment information of the review text, which has some impact on the model after elimination, which not only illustrates that adding review text as an additional feature can somewhat alleviate data sparsity, but also the importance of sentiment features.

Finally, the inclusion of an attention mechanism in the extraction of graph node features, which enables nodes with different edge weights to be modelled at different computational scales, also has the effect of bringing better recommendation results to the model.

In summary, each key part of our proposed RWESA-GNNR achieves good results.

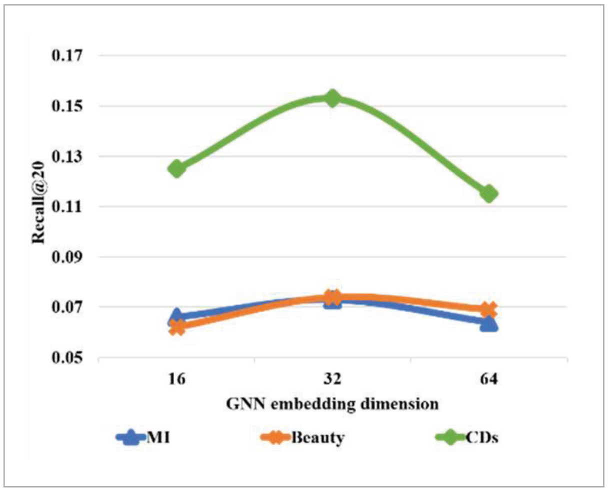
3.4. Hyperparametric Analysis

To better improve the recommendation of the model, we used grid search to debug the important hyperparameters of the model and visualized the results with the evaluation metric Recall@20.

We chose the appropriate GNN embedding dimension in the range of {16,32,64} and the results are shown in Figure 5. The best results were achieved when the GNN embedding dimension was 32, but the model performance deteriorated when the embedding dimension was larger, which might be due to the overfitting of the model caused by too large embedding dimension. Therefore, we set the GNN embedding dimension to 32.

Figure 5

The effect of the GNN embedding dimension



We chose the appropriate number of layers of the GNN in the range of {1,2,3,4}, and the results are shown in Figure 5. The best results were achieved with the number of layers of the GNN at layer 2, while deeper layers of the GNN did not improve the performance of the model much, probably because the representation between nodes was too similar after multi-layer neighbor aggregation, leading to smoothing problems in the model. Therefore, we set the number of layers of the GNN to 2.

We selected appropriate word embedding dimensions for the item text in the range of {50,100,200,300} and pre-trained the model using glove, and the results are shown in Figure 7. The model performance did not improve significantly as the word embedding dimension increased, probably because the smaller word embedding dimension was sufficient to capture the implicit information contained in the item text. Therefore, to speed up the training of the model, we set the word embedding dimension to 50.

Figure 6

The effect of the number of GNN layer

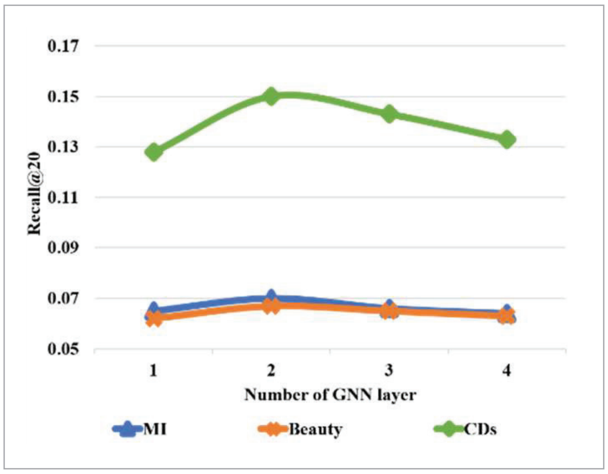
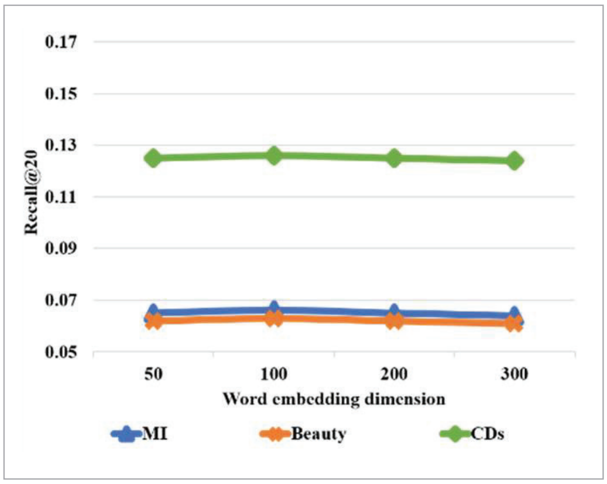


Figure 7

The effect of the word embedding dimension



4. Conclusion

In this paper, we propose a GNN recommendation algorithm based on random wandering embedding combined with sentiment analysis, called RWESA-GNNR. The model first uses random wandering-based matrix factorization to obtain embedding vectors of nodes, then uses GCN combined with attention to learn node representations of users and items, and then incorporates attentional mechanisms sentiment analysis to add sentiment attributes to nodes. RWESA-GNNR achieved better performance than the baselines on three publicly available datasets from Amazon.

In future research work, we will extend our work in two directions: first, the complexity of the model leads to less fast recommendations, especially for machines with less than high arithmetic power, so we intend to simplify the structure of the model to speed up the recommendations without compromising the results. Secondly, we only considered the effect of the sentiment attribute of the node on the recommendation algorithm, and future research could consider the effect of other node attributes on the recommendation algorithm.

Acknowledgement

This work was supported in part by the Jiangsu Provincial Department of Industry and Information Technology Key Technology Innovation Project Guidance Plan (Grant no. 141-62-65), in part by the Digital public service platform project of Jiangsu Provincial Department of Science and Technology (Grant no. 93208000931), in part by the Industry-university-research project of Jiangsu Provincial Department of Science and Technology (Grant no. 20221343), in part by the Jiangsu Province Vocational Education Big Data Technology "Double Teacher" Famous Teacher Studio Project.

References

1. Canturk, D., Karagoz, P., Kim, S., Toroslu, l. Trust-Aware Location Recommendation in Location-Based Social Networks: A Graph-Based Approach, Expert Systems with Applications, 2023, 213: 119048. https://doi.org/10.1016/j.eswa.2022.119048
2. Chen, C., Zhang, M., Liu, Y., Ma S. Neural Attentional Rating Regression with Review-Level Explanations. In WWW '18, 2018: 1583-1592. https://doi.org/10.1145/3178876.3186070
3. Chen, J., Dong, H., Wang, X., Feng, F., Wang, M., He, X. Bias and Debias in Recommender System: A Survey and Future Directions. ACM Transactions on Information Systems, 2023, 41(3): 1-39. https://doi.org/10.1145/3564284
4. Chen, Z., You, Z., Guo, Z., Yi, H. Luo, G. Wang, Y. Prediction of Drug-Target Interactions from Multi-Molecular Network Based on Deep Walk Embedding Model. Frontiers in Bioengineering and Biotechnology, 2020, 8: 338. https://doi.org/10.3389/fbioe.2020.00338
5. Church, K. Word2Vec. Natural Language Engineering, 2017, 23(1): 155-162. https://doi.org/10.1017/S1351324916000334
6. Fan, W., Ma, Y., Li, Q., He, Y. Zhao, E. Tang, J. Yin, D. Graph Neural Networks for Social Recommendation. In: WWW '19, 2019: 417-426. https://doi.org/10.1145/3308558.3313488
7. Guo, S., Chen, C., Wang, J., Liu, Y., Xu, K., Yu, Z., Zhang, D., Chiu, D. Rod-Revenue: Seeking Strategies Analysis and Revenue Prediction in Ride-on-Demand Service Using Multi-Source Urban Data. IEEE Transactions on Mobile Computing, 2019, 19(9): 2202-2220. https://doi.org/10.1109/TMC.2019.2921959
8. He, X., Deng, K., Wang, X., Li, Y., Zhang, Y., Wang, M. LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation. In SIGIR '20, 2020: 639-648. https://doi.org/10.1145/3397271.3401063
9. Huang, Z., Silva, A., Singh, A. A Broader Picture of Random-Walk Based Graph Embedding. In KDD '21, 2021: 685-695. https://doi.org/10.1145/3447548.3467300
10. Jiang, H., Cao, P., Xu, M. Yang, J. Zaiane, O. Hi-GCN: A Hierarchical Graph Convolution Network for Graph Embedding Learning of Brain Network and Brain Disorders Prediction. Computers in Biology and Medicine, 2020, 127: 104096. https://doi.org/10.1016/j.compbiomed.2020.104096
11. Lee, M., Kim, J., Goh, K., Lee, S. Son, S. Lee, D. Degree Distributions under General Node Removal: Power-Law or Poisson?. Physical Review E, 2022, 106(6), 064309. https://doi.org/10.1103/PhysRevE.106.064309
12. Liu, X., Wu, K., Liu, B., Qian, R. HNERec: Scientific Collaborator Recommendation Model Based on Heterogeneous Network Embedding, 2020, 8: 338. https://doi.org/10.1016/j.ipm.2022.103253
13. Noia, T., Ostuni, V., Tomeo, P., Sciascio, E. Sprank: Semantic Path-Based Ranking for Top-N Recommendations Using Linked Open Data. ACM Transactions on Intelligent Systems and Technology, 2016, 8(1): 1-34. https://doi.org/10.1145/2899005
14. Peng, Y., Yan, S., Lu, Z. Transfer Learning in Biomedical Natural Language Processing: An Evaluation of BERT and ELMo on Ten Benchmarking Datasets. arXiv preprint arXiv:1906.05474, 2019. https://doi.org/10.18653/v1/W19-5006
15. Reece, A., Danforth, C. Instagram Photos Reveal Predictive Markers of Depression. EPJ Data Science, 2017, 6(1), 15. https://doi.org/10.1140/epjds/s13688-017-0110-z
16. Rendle, S., Freudenthaler, C., Gantner, Z., Schmidt-Thieme, L. BPR: Bayesian Personalized Ranking from

Implicit Feedback. arXiv preprint arXiv:1205.2618, 2012.

17. Rodríguez, P., Bautista, M., Gonzalez, J., Escalera, S. Beyond One-Hot Encoding: Lower Dimensional Target Embedding. Image and Vision Computing, 2018, 75: 21-31. https://doi.org/10.1016/j.imavis.2018.04.004
18. Sang, S., Liu, N., Li, W., Zhang, Z, Qin, Q, Yuan, W. High-Order Attentive Graph Neural Network for Session-Based Recommendation. Applied Intelligence, 2022: 1-15. https://doi.org/10.1007/s10489-022-03170-7
19. Sgherzi, F., Parravicini, A., Santambrogio, M. A Mixed Precision, Multi-GPU Design for Large-Scale Top-K Sparse Eigenproblems. In ISCAS'22, IEEE, 2022: 1259-1263. https://doi.org/10.1109/ISCAS48785.2022.9937893
20. Song, W., Xiao, Z., Wang, Y., Charlin, L. Zhang, M. Tang, J. Session-Based Social Recommendation Via Dynamic Graph Attention Networks. In WSDM '19, 2019: 555-563. https://doi.org/10.1145/3289600.3290989
21. Song, X., Salcianu, A., Song, Y., Dopson, D., Zhou, D. Fast Wordpiece Tokenization. arXiv preprint arXiv:2012.15524, 2020. https://doi.org/10.18653/v1/2021.emnlp-main.160
22. Wang, X., He, X., Cao, Y., Liu, M. Chua, T. Kgat: Knowledge Graph Attention Network for Recommendation. In KDD '19, 2019: 950-958. https://doi.org/10.1145/3292500.3330989
23. Wu, S., Tang, Y., Zhu, Y. Wang, L. Xie, X. Tan, T. Session-Based Recommendation with Graph Neural Networks. In AAAI '19, 2019, 33(01): 346-353. https://doi.org/10.1609/aaai.v33i01.3301346
24. Xiao, B., Xie, X., Yang, C., Wang. Y. RTN-GNNR: Fusing Review Text Features and Node Features for Graph Neural Network Recommendation. IEEE Access, 2022, 10: 114165-114177. https://doi.org/10.1109/ACCESS.2022.3218882
25. Yin, R., Li, K., Zhang, G. Lu, J. A Deeper Graph Neural Network for Recommender Systems. Knowledge-Based Systems, 2019, 185: 105020. https://doi.org/10.1016/j.knosys.2019.105020
26. Ying, R., He, R., Chen, K. Eksombatchai, P., Hamilton, W. Graph Convolutional Neural Networks for Web-Scale Recommender Systems. In KDD '18, 2018: 974-983. https://doi.org/10.1145/3219819.3219890
27. Zhang, R., Bai, X., Pan, L., Dong, Z,. Song, R. Zero-Small Sample Classification Method with Model Structure Self-Optimization and Its Application in Capability Evaluation. Applied Intelligence, 2022, 52(5): 5696-5717. https://doi.org/10.1007/s10489-021-02686-8
28. Zhang, Y., Shen, J., Zhang, R., Zhao, Z. Network Representation Learning via Improved Random Walk with Restart, Knowledge-Based Systems, 2023: 110255. https://doi.org/10.1016/j.knosys.2023.110255
29. Zheng, L., Noroozi, V., Yu, P. Joint Deep Modeling of Users and Items Using Reviews for Recommendation. In WSDM '17, 2017: 425-434. https://doi.org/10.1145/3018661.3018665
30. Zhou, Y., Zhou, Y., Yu, D., Sun, J. Adaptive Social Recommendation Based on Negative Similarity, Journal of Computer Applications, 2023: 1. https://doi.org/10.11772/j.issn.1001-9081.2022071003.