Context-Based Higher-Order Relation-Aware Denoising GNN for Recommendations*

Haijun Liu[1*]

[1] *Fiberhome Telecommunication Technologies Co.,LTD, Wuhan 430068, China*

Abstract

Recommender systems serve as intelligent tools to alleviate information overload and provide personalised services to users. Existing recommender systems that rely on user behavior to generate data often face the problem of data sparsity. Moreover, graph neural network-based recommendation algorithms model user feature representations by treating neighbours equally, but ignore the problem of inconsistency in neighbour preferences in a given context. In this paper, we propose CRDG, a GNN model based on context-aware denoising. Specifically, we first construct user similarity graphs and item relevance graphs from historical interaction data and capture useful information from implicit neighbours with similar preferences through higher-order relationship models to alleviate the data sparsity problem exacerbated in existing denoising-based contextual recommendations. Then, to address the context inconsistency problem, we propose a denoising GNN model to aggregate information from contextually consistent neighbours. In addition to refine the influence of different types of neighbours, we propose a dual-attention model to assign different influence weights to different neighbours. Experimental results on several real datasets demonstrate the superiority and effectiveness of the proposed model.

Keywords

graph neural networks; recommendation system; graph attention; denoising; context consistency.

1. Introduction

The massive amount of information generated due to the rapid development of the Internet has brought about the problem of information overload. Recommender systems **are** one of the main solutions for addressing this problem. Among the many recommendation algorithms, Collaborative filtering (CF) has received extensive attention from researchers due to its simplicity and efficiency [1,2]. However, CF-based recommendation algorithms often face the problem of data sparsity [3].

In recent years, graph neural networks (GNNs) have shown great advantages in recommendation systems by virtue of their powerful modeling capabilities on non-Euclidean data [4].The GC-MC model proposed by Berg et al. applies GCNs to a matrix completion task with edge information and converts the matrix completion task into a link prediction problem, which is modelled using an end-to-end graph self-encoder [5].The NGCF proposed by Wang et al. et al. proposed NGCF to improve recommendation by stacking multiple embedding propagation layers to capture higher-order connectivity in the user-item graph [6]. While all of the above recommendation methods demonstrate strong performance, most of the current algorithms treat the neighbors' information equally and do not consider it in a specific recommendation context, leading to the problem of contextual inconsistency.

The Second International Conference of Young Scientists on Artificial Intelligence for Sustainable Development (YAISD), May 8-9, 2025, Ternopil, Ukraine

* Corresponding author.

† These authors contributed equally.

15572883917@163.com (L. Haijun)

0009-0000-7972-6549 (L. Haijun)

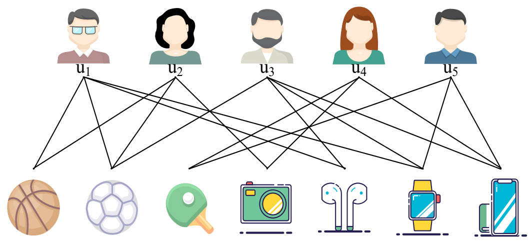

Figure 1: An illustration of context inconsistency.

Given a specific recommendation context (usually a user-item pair), the user may be inconsistent with the neighborhood information in the given context. For example, based on Fig. 1 when considering user u_1's preference for digital products, we argue that information aggregation for sports items is potentially noisy due to the inconsistency of users' requirements for different types of items. Some previous studies have reduced the impact of contextually inconsistent connections by filtering out some first-order neighbors [7], but this strategy exacerbates the sparsity of the data. To address this problem, we propose to leverage higher-order relations to capture useful information for target users from implicit neighbors with similar preferences. For example, in Fig. 1, based on historical interaction information, we argue that user u_2 and user u_3 can provide effective information for predicting user u_1 preferences. However, in the recommendation scenario of digital products, we believe that user u_3 can provide more effective information for modelling user's preferences, while aggregating users may introduce noise due to the fact that the same interactions of user u_2 and user u_1 are mainly reflected in the motion neighborhood, whereas they exhibit dissimilar preferences in the digital domain. In addition, exploiting item correlation is also helpful because an item and related items are likely to be purchased together by a specific group of people, a well-known example being beer and diapers. Most traditional CF-based algorithms utilize these findings, but GNN-based models ignore them.

The above analysis shows the drawbacks of ignoring specific contexts (i.e., a single user may have different context-consistent neighbors for different items). To this end, this paper proposes CRDG, a context-aware GNN recommendation model that aggregates useful information from contextually consistent neighbors. First, CRDG constructs user-similar collaboration graphs and item-related collaboration graphs from user-item history interactions for users and items, respectively, and models the impact of implicit neighbors in user-similar graphs and the role of item associations in item-related graphs through a higher-order relationship-aware module. Next, to mitigate the effect of context-inconsistent neighbors, we construct a context-aware denoising module. The model removes context-inconsistent neighbors by sampling and aggregates information only from context-consistent neighbors. Then, to refine the influence of neighbors, we propose a dual-attention model to assign weights to contextually consistent neighbors. Finally, we connect the initial feature representations modeled based on higher-order perceptual modules with the final features based on consistent neighbor aggregation for recommendation prediction. Extensive experiments on real datasets demonstrate the effectiveness of our model.

2. Formalization of problems

Let a bipartite graph $G=(U,V,E_{UV})$ of user-item interactions consist of two different types of node sets (User set $U=\{u_1,u_2,\cdots,u_m\}$ and Item set $I=\{i_1,i_2,\cdots,i_n\}$) and edge set E_{UV}, where m and n denote the number of users and items, respectively. We denote the interaction matrix as $A\in\mathbb{R}^{m\times n}$, where $a_{u,v}\in\{0,1\}$. and only $a_{u,v}=1$ denotes that user u interacts with item v, i.e., $(u,v)\in E_{UV}$, otherwise $a_{u,v}=0$.

Formally, the recommendation algorithm aims to construct an interaction prediction matrix $R \in \mathbb{R}^{m\times n}$ between users and items based on user-item interaction data. i.e., it first learns the latent feature representations of users and items, and then predicts user-item interactions based on the product of the feature representations of users and items.

$$r_{uv} = e_u \cdot e_v \tag{1}$$

where $e_u \in \mathbb{R}^d$, $e_v \in \mathbb{R}^d$ denote the final embedding of the user and the item, respectively, and d is the embedding dimension.

3. CRDG model

This section first outlines the overall framework of the proposed model. The overall architecture of the model is shown in Fig. 2. The four main component parts into which the model is divided will then be described in detail.

Higher-order relationship awareness module: In this module, we construct user similarity graphs and item relevance graphs based on historical user-item interactions, and then generate preliminary feature representations of users and items by aggregating implicit similarity information within the neighborhood via a higher-order relation-aware graph neural network (RGNN).

Context-aware denoising module: In this module, we first construct embedded representations of context pairs through the query layer, and then filter and denoise the candidate neighbors to obtain context-consistent neighbors.

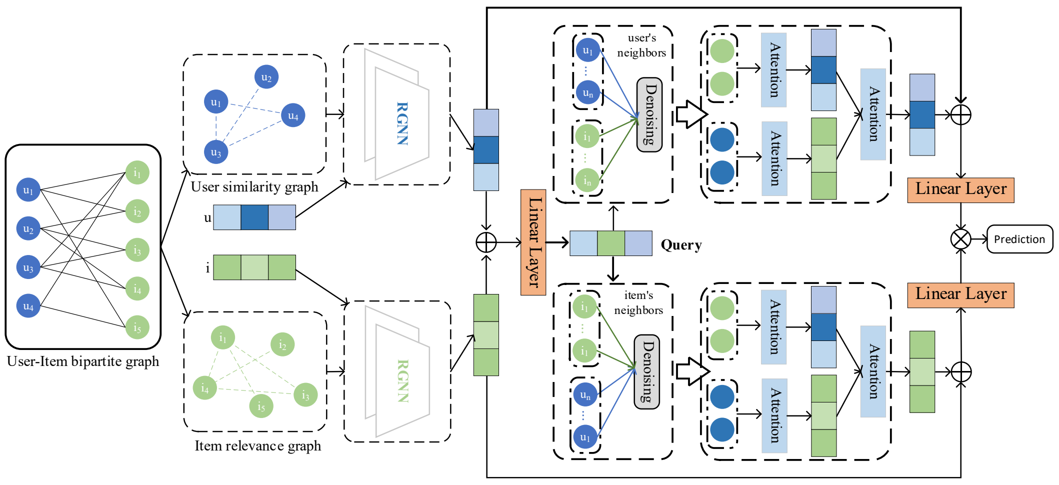

Figure 2: The CRDG model framework

Dual attention based consistent neighbors aggregation module: We assign different impact weights to context-consistent neighbors and then update the user and project representations based on their corresponding impact weights.

Prediction module: In this part, we link the preliminary representations of users and items obtained based on RGNN and the feature representations based on context-consistent neighbors aggregation as the final representations, and then output the prediction results based on the inner product of the point feature representations.

3.1. Higher-order relationship awareness module

3.1.1. Collaboration graph construction

In order to capture the influence of implicit neighbors during feature representation, based on the original user-item interaction bipartite graph $G = (U, V, E_{UV})$, we construct a user similarity graph

$G_U^s=(U,\mathcal{E}_U^s)$ and an item correlation graph $G_V^r=(V,\mathcal{E}_V^r)$, where $\mathcal{E}_U^s$ and $\mathcal{E}_V^r$ are the sets of edges of the two collaboration graphs, respectively:

$$sim(u_i,u_j)=\frac{\left|\mathcal{N}_V^G(u_i)\cap\mathcal{N}_V^G(u_j)\right|}{\sqrt{\left|\mathcal{N}_V^G(u_i)\right|\cdot\left|\mathcal{N}_V^G(u_i)\right|}} \tag{2}$$

where $\mathcal{N}_V^G(u_i)$ and $\mathcal{N}_V^G(u_j)$ denote the neighbors of u_i and u_j on the user item bipartite graph G, i.e., the set of interacting items, respectively. If $sim(u_i,u_j)>\tau$, we add an edge between u_i and u_j where τ is the hyperparameter. Similarly, we compute the correlation between items as follows.

$$rel(v_i,v_j)=\frac{\left|\mathcal{N}_U^G(v_i)\cap\mathcal{N}_U^G(v_j)\right|}{\left|\mathcal{N}_U^G(v_i)\cup\mathcal{N}_U^G(v_j)\right|} \tag{3}$$

where $\mathcal{N}_U^G(v_i)$ and $\mathcal{N}_U^G(v_j)$ denote the neighbors of v_i and v_j on the user item bipartite graph G, i.e., the set of users who have interacted with v_i and v_j, respectively. Similarly if $rel(v_i,v_j)>\xi$, we then add an edge between v_i and v_j, where ξ is a hyperparameter.

3.1.2. Higher-order relationship-aware graphical neural networks

We construct a higher-order relationship-aware graph neural network (RGNN) that can effectively aggregate the relevant information in the neighboring nodes in the two collaborative graphs, and the detailed process is as follows.

Aggregating Similar User Neighbours: for each user u_i, given the layer l feature representation $x_{u_i}^l$, we will update the user feature representation at layer $l+1$ as follows:

$$x_{u_i}^{l+1}=\sigma\left(\sum_{u_j\in\mathcal{N}_U^s(u_i)}\left(W_1^{s,l}x_{u_i}^l+W_2^{s,l}\left(x_{u_i}^l\odot x_{u_j}^l\right)\right)\right) \tag{4}$$

where $W_1^{s,l}$ and $W_2^{s,l}$ are the learnable parameters of layer l, $\odot$ denotes the element-wise product, σ is the LeakyReLU activation function, and $\mathcal{N}_U^s(u_i)$ denotes the neighbourhood of u_i on G_U^s, i.e., the similar user neighbourhood of user u_i.

Aggregating Related Item Nneighbours: Similarly, for each item v_i, given the layer l feature representation $x_{v_i}^l$, we will update the item feature representation at layer $l+1$ as follows:

$$x_{v_i}^{l+1}=\sigma\left(\sum_{v_j\in\mathcal{N}_V^r(v_i)}\left(W_1^{r,l}x_{u_i}^l+W_2^{r,l}\left(x_{v_i}^l\odot x_{v_j}^l\right)\right)\right) \tag{5}$$

where $W_1^{r,l}$ and $W_2^{r,l}$ are the learnable parameters of layer l, $\mathcal{N}_V^r(v_i)$ denotes the neighbourhood of v_i on G_V^r.

Finally, after L iterative propagation, we obtain the set of user and item feature representations, x_u^l and x_v^l where $l=[0,1,2,\cdots,L]$. Then connecting the feature representations of users and items at each level yields a preliminary feature representation $x_u^s=\left[x_u^0\,\|\,x_u^1\,\|\cdots\|\,x_u^L\right]$ of the user's similarity-based feature representation and a preliminary feature representation $x_v^r=\left[x_v^0\,\|\,x_v^1\,\|\cdots\|\,x_v^L\right]$ of the item's correlation-based feature representation , where x_u^0 and x_v^0 are the original input feature representations.

3.2. Context-aware denoising module:

To solve the context-inconsistent problem, we design a context-aware denoising model, which consists of a query layer and a context- consistent denoising-based module.

3.2.1. Query layer

A recommendation context refers to a user-item pair (u,v) of a user's preference for a specific item , and in order to capture the representation of a specific context, CRDG builds the query layer to select context-consistent neighbors specifically for a specific recommendation context (u,v). Specifically, it generates context embeddings by mapping the connection between the initial embeddings of users and items:

$$q_{u,v} = \sigma\left(W_q\left(x_u^s \oplus x_v^r\right)\right) \tag{6}$$

where $q_{u,v}$ is the context embedding, x_u^s and x_v^r are the preliminary feature representations of u and v, respectively. W_q is the learnable parameter, and $\oplus$ denotes the connection operation. Based on the query layer, we can dynamically sample neighbors according to different contexts.

3.2.2. Denoising based on context-consistent

In the higher-order relationship-aware module above, we ignore the interaction information between the user and the item. Therefore, in this phase we will introduce the user-item interaction graph G. After the introduction of G, for user u two different types of neighbor nodes exist, i.e., item neighbors $\mathrm{N}_V^G(u)$ in G and user neighbors $\mathrm{N}_U^s(u)$ in G_U^s. Similarly, item v has user neighbors $\mathrm{N}_U^G(v)$ in G and item $\mathrm{N}_V^r(v)$ neighbors in G_V^r. In order to realize the information transfer between heterogeneous nodes, we map the preliminary user and item feature representations to the same embedding space as follows:

$$\begin{cases} e_u^0 = W_U x_u^s, u \in U \\ e_v^0 = W_V x_v^r, v \in V \end{cases} \tag{7}$$

where W_U and W_V are learnable parameters. Given context (u,v), we obtain the embedding $q_{u,v}$ of (u,v) through the query layer, and then the context consistency score for user u's neighbor $n_u \in \mathrm{N}_U^s(u) \cup \mathrm{N}_V^G(u)$ is computed as follows

$$p\left(n_u; q_{u,v}\right) = \frac{\exp\left(-\left\|q_{u,v} - e_{n_u}^0\right\|_2^2\right)}{\sum_{n_u' \in \mathrm{N}_U^s(u) \cup \mathrm{N}_V^G(u)} \exp\left(-\left\|q_{u,v} - e_{n_u'}^0\right\|_2^2\right)} \tag{8}$$

Similarly, the context consistency score for item v 's neighbors B is calculated $n_v \in \mathrm{N}_V^r(v) \cup \mathrm{N}_U^G(v)$ as follows:

$$p\left(n_v; q_{u,v}\right) = \frac{\exp\left(-\left\|q_{u,v} - e^0_{n_v}\right\|_2^2\right)}{\sum_{n_v' \in \mathrm{N}_V^r(v) \cup \mathrm{N}_U^G(v)} \exp\left(-\left\|q_{u,v} - e^0_{n_v'}\right\|_2^2\right)} \tag{9}$$

where n_u and n_v denote the neighbors of user u and item v, respectively. It is worth noting that they can be item nodes as well as user nodes. Based on the above definitions, we can compute the consistency scores of all neighbors of u and v with the given context (u,v).

We then select neighbors with the top percent γ of the consistency score, where $0 \le \gamma \le 1$ is a hyperparameter. In this way, we can filter out most of the context-inconsistent neighbors, thus eliminating their negative impact. After completing the denoising process, user u retains two types of contextually consistent neighbors:

$$\tilde{\mathrm{N}}_U^s(u) = \left\{n_u \mid p\left(n_u; q^{u,v}\right) \in top_\gamma\left(\mathrm{N}_U^s(u)\right)\right\} \tag{10}$$

$$\tilde{\mathrm{N}}_V^G(u) = \left\{n_u \mid p\left(n_u; q^{u,v}\right) \in top_\gamma\left(\mathrm{N}_V^B(u)\right)\right\} \tag{11}$$

where $\tilde{\mathrm{N}}_U^s(u)$ denotes the set of context (u,v) consistent user neighbors of user u on G_U^s, and $\tilde{\mathrm{N}}_V^G(u)$ denotes the set of context (u,v) consistent item neighbors of user u on G. $top_\gamma(\cdot)$ denotes the selection of the top percent γ of elements in the set. Similarly, for item v we retain two contextually consistent neighbors:

$$\tilde{\mathrm{N}}_V^r(v) = \left\{n_v \mid p\left(n_v; q^{u,v}\right) \in top_\gamma\left(\mathrm{N}_V^r(v)\right)\right\} \tag{12}$$

$$\tilde{\mathrm{N}}_U^G(v) = \left\{n_v \mid p\left(n_v; q^{u,v}\right) \in top_\gamma\left(\mathrm{N}_U^G(v)\right)\right\} \tag{13}$$

where $\tilde{\mathrm{N}}_V^r(v)$ denotes the set of context-consistent item neighbors of item v on G_V^r and $\tilde{\mathrm{N}}_U^G(v)$ denotes the set of context-consistent user neighbors of item v on G. Next, to aggregate different types of contextually consistent neighbor information, we design a dual-attention based consistent neighbor aggregation module.

3.3. Dual Attention Based Consistent Neighbor Aggregation Module

Impact of different neighbors of the same type: Given that users u, $\tilde{\mathrm{N}}_U^s(u)$ are consistent user neighbors of u in G_U^s, we aggregate the features of these consistent user neighbors as follows:

$$e^l_{\tilde{\mathrm{N}}_U^s(u)} = \sum_{u' \in \tilde{\mathrm{N}}_U^s(u)} \alpha^l_{u,u'} e^{l-1}_{u'} \tag{14}$$

where $e^{l-1}_{u'}$ denotes the. feature representation of u's neighbor u' in layer $l-1$, and $\alpha^l_{u,u'}$ is the weight corresponding to u'. The calculation is as follows.

$$\alpha^l_{u,u'} = \frac{\exp\left(\sigma\left(W^l\left(e^{l-1}_u \oplus e^{l-1}_{u'}\right)\right)\right)}{\sum_{u'' \in \tilde{\mathrm{N}}_U^s(u)} \exp\left(\sigma\left(W^l\left(e^{l-1}_u \oplus e^{l-1}_{u''}\right)\right)\right)} \tag{15}$$

where W^l is the learnable parameter of layer. Similarly for user u's contextually consistent item neighbor $v' \in \tilde{\mathrm{N}}_V^G(u)$ on G we take the same aggregation approach and compute the following:

$$e^l_{\tilde{\mathrm{N}}_V^G(u)} = \sum_{v' \in \tilde{\mathrm{N}}_V^G(u)} \alpha^l_{u,v'} e^{l-1}_{v'} \tag{16}$$

$$\alpha_{u,v'}^{l}=\frac{\exp\left(\sigma\left(W^{l}\left(e_{u}^{l-1}\oplus e_{v'}^{l-1}\right)\right)\right)}{\sum_{v''\in \tilde{N}_{V}^{G}(u)}\exp\left(\sigma\left(W^{l}\left(e_{u}^{l-1}\oplus e_{v''}^{l-1}\right)\right)\right)} \tag{17}$$

where $e_{v'}^{l-1}$ denotes the potential embedding of item neighbor v' in layer $l-1$, $\alpha_{u,v'}^{l}$ is the weight parameter corresponding to v'.

Impact of different types of neighbors: We propose a secondary attention for aggregating information from consistent user neighbors and consistent project neighbors, the aggregated embedding obtained by user u is shown below:

$$AGG_{u}^{l}=\beta_{\tilde{N}_{U}^{s}(u)}^{l}e_{\tilde{N}_{U}^{s}(u)}^{l}+\beta_{\tilde{N}_{V}^{G}(u)}^{l}e_{\tilde{N}_{V}^{G}(u)}^{l} \tag{18}$$

where $e_{\tilde{N}_{U}^{s}(u)}^{l}$ and $e_{\tilde{N}_{V}^{G}(u)}^{l}$ are the aggregation embeddings of consistent user neighbors and consistent item neighbors of user u, respectively. $\beta_{\tilde{N}_{U}^{s}(u)}^{l}$ and $\beta_{\tilde{N}_{V}^{G}(u)}^{l}$ are the attention weights of consistent user neighbors and consistent item neighbors, respectively, as follows:

$$\beta_{\tilde{N}_{U}^{s}(u)}^{l}=\frac{\exp\left(\sigma\left(W^{l}\left(e_{u}^{l-1}\oplus e_{\tilde{N}_{U}^{s}(u)}^{l-1}\right)\right)\right)}{\sum_{g\in \tilde{N}_{U}^{s}(u)\cup \tilde{N}_{V}^{G}(u)}\exp\left(\sigma\left(W^{l}\left(e_{u}^{l-1}\oplus e_{g}^{l-1}\right)\right)\right)} \tag{19}$$

$$\beta_{\tilde{N}_{V}^{G}(u)}^{l}=\frac{\exp\left(\sigma\left(W^{l}\left(e_{u}^{l-1}\oplus e_{\tilde{N}_{V}^{G}(u)}^{l-1}\right)\right)\right)}{\sum_{g\in \tilde{N}_{U}^{s}(u)\cup \tilde{N}_{V}^{G}(u)}\exp\left(\sigma\left(W^{l}\left(e_{u}^{l-1}\oplus e_{g}^{l-1}\right)\right)\right)} \tag{20}$$

where W^{l} is the learnable parameter of layer l. Finally, the embedding of user u in layer l is updated as follows.

$$e_{u}^{l}=\sigma\left(W_{u}^{l}\left(e_{u}^{l-1}\oplus AGG_{u}^{l}\right)\right) \tag{21}$$

where W_{u}^{l} is the l-layer learnable parameter. Similarly given the item v, we can follow the above method to obtain the updated representation as follows:

$$e_{v}^{l}=\sigma\left(W_{v}^{l}\left(e_{v}^{l-1}\oplus AGG_{v}^{l}\right)\right) \tag{22}$$

where W_{v}^{l} is the l-layer learnable parameter. AGG_{v}^{l} is calculated similarly to AGG_{u}^{l}.

3.4. Prediction module

After completing the consistent neighbor information aggregation, we can get the hierarchical embedding of user and item features, i.e., e_{u}^{l} and e_{v}^{l} where $l=[0,1,2,\dots,L]$. We select the embedding values of the first and the last layer among them, so the final representations of users and items are as follows:

$$e_{u}=\sigma\left(W_{u}\left(e_{u}^{0}\oplus e_{u}^{L}\right)\right) \tag{23}$$

$$e_{v}=\sigma\left(W_{v}\left(e_{v}^{0}\oplus e_{v}^{L}\right)\right) \tag{24}$$

where W_{u} and W_{v} are the learnable weight matrices. Then we take the following loss function to measure the deviation between the predicted and true values:

$$L=\frac{1}{2|E_{UV}|}\sum_{u,v\in E_{UV}}\left(r_{uv}-a_{u,v}\right)^{2} \tag{25}$$

where E_{UV} is the edge set of user-item interactions, and each (u,v) in E_{UV} is considered as a recommendation context. d is the predicted value of user-item interactions.

4. experiment

In this section, the author mainly introduces the research content involved in the experiment, and then describes the datasets, evaluation metrics, experimental settings and experimental results used in this work.

4.1. Experimental dataset

Table 1
Statistical dataset

Dataset	Yelp	Amazon
Users	32,654	52, 643
Items	34,193	91, 599
Interaction	1,347,861	2, 984, 108
Density	0.121%	0.062%

We apply two publicly accessible datasets Yelp and Amazon to our model for performance validation, Table 1 summarizes the statistics of the datasets. For each dataset, we randomly selected 80% of the data as the training set, and the remaining 10% and 10% as the validation set and test set.

4.2. Baseline algorithm

The proposed model CRDG is compared with the following baselines.

FM [8]: A second-order cross term is added to the traditional linear model to represent the interaction between features by learning the auxiliary vectors of the features.

NCF [9]: Introducing deep learning to learn non-linear interactions between users and items.

GCN [10]: Learning Complex Relationships between Users and Items Using Spectral Convolutional Operators to Improve the Performance and Accuracy of Recommender Systems by Learning User-Item Interaction Graphs.

NGCF [6]: Learning User and Item Representations by Explicitly Encoding Collaborative Signals in Higher-Order Connections by Propagating Embeddings on User-Item Interaction Graphs.

LightGCN [11]: Simplify the NGCF model by retaining only the neighborhood aggregation operation to improve the recommendation effect and computational efficiency.

DiffNet++ [12]: Achieved better performance in recommendation tasks by modeling the user's interest and influence diffusion process

GraphDA [13]: A denoised and augmented user-item matrix is generated by capturing the correlation between user-user and item-item, and by top-K sampling.

4.3. Analysis of results

4.3.1. Contrast to the baseline algorithm

Table 2
Contrast to the baseline algorithm

Analysis of results	Yelp		Amazon	
	Recall@10	**NDCG@10**	**Recall@10**	**NDCG@10**
FM	0.2132	0.1745	0.1363	0.0921

NCF	0.2239	0.1797	0.1402	0.0968
NGCF	0.2435	0.1965	0.1479	0.1093
LightGCN	0.2539	0.2014	0.1581	0.1164
Diffnet++	0.2659	0.2158	0.1698	0.1263
GraphDA	0.2747	0.2179	0.1711	0.1264
CRDG	**0.2851**	**0.2215**	**0.1842**	**0.1303**

The performance comparison results are shown in Table 2. From the results, the following observations can be made:

First FM, NCF performs poorly on both datasets. This is due to the fact that traditional collaborative filtering-based methods are difficult to comprehensively model the interaction between users and items compared to graph-based recommendation algorithms. Secondly, among the graph-based learning methods Diffnet++ compared to the traditional graph learning methods NGCF and LightGCN it is designed with a diffusion method that can effectively cross the limitation of one-hop neighboring nodes, and thus can capture richer graph attributes. The reason why GraphDA is able to achieve a better approach than the above methods may be due to the fact that the interaction matrix based on denoising and enhancement can mitigate the effect of noise in the existing interaction matrix, which in turn can model the feature representation of the user and the project more efficiently.

The proposed CRDG model performance baseline algorithm for the following reasons:1. Higher-order relationship-aware module, which can effectively model the implicit similarity and relevance of users and items, can be used as a complement to enhance the information sparsity problem that exists in traditional recommendation.2. The denoising model based on can effectively alleviate the noise problem that exists in the process of aggregation of information from contextually inconsistent neighbors.3. Compared to the GraphDA and other denoising methods, our proposed dual-attention based consistent neighbor aggregation module can refine the influence of different neighbors more effectively.

4.3.2. Ablation analysis

In order to study the impact of each component, we designed three CRDG variants as follows.

CRDG-RGNN, removes the higher-order relation-aware module from CRDG, i.e., the initial feature representation is directly passed into the subsequent denoising and attention modules.

CRDG-Denoising, removes the context-based denoising step from CRDG, i.e., the dual attention module directly aggregates the representation information of the whole neighbors.

CRDG-Datt, replaces the dual-attention model with the traditional GNN model.

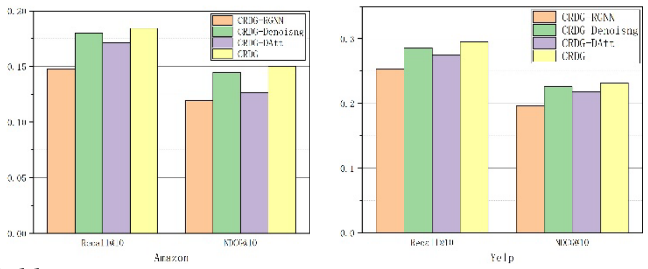

Figure 3: Results of ablation experiments.

The experimental results are shown in Fig3. We can observe that CRDG consistently achieves the best performance compared to the other variants, suggesting that all components are necessary to obtain the best results. CRDG-RGNN exhibits poor performance reflecting the importance of the higher-order relationship-aware module, for modeling user similarity and item relevance. CRDG-Denoising performs sub-optimally and greatly reflects the fact that a mechanism based on the dual attention mechanism can effectively refine the influence of neighbors.

4.3.3. Parametric sensitivity analysis

In this subsection, we investigate how the performance of our proposed model varies with some hyperparameters, including its embedding dimension d , the threshold τ of the user similarity graph and the threshold ξ of the item related graph , and the experimental results are as follows. The experimental results are shown below.

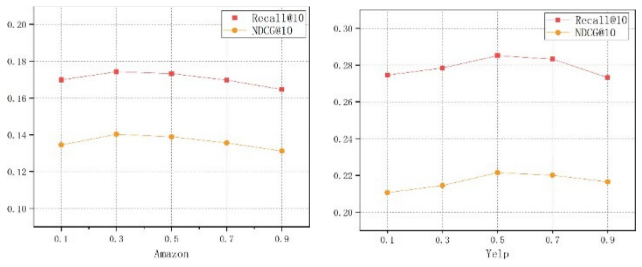

Figure 4: Effect of Embedding Dimension on the Model

The results in Figure 4 show that the model performance tends to show an increasing and then decreasing trend with increasing d . However, the optimal dimension d varies across datasets, with the best performance achieved when $d = 32$ on the Yelp dataset and better results when $d = 64$ on the Amazon dataset. The explanation for this result is that when d is small as d increases the model's modeling ability is stronger, but when d is too large it leads to overfitting problems.

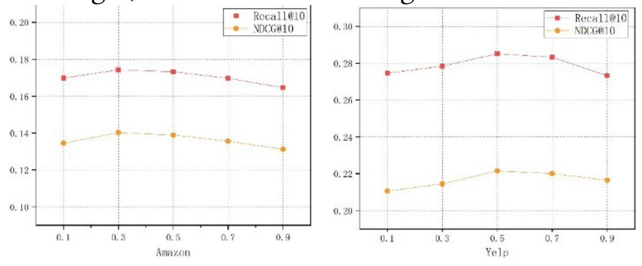

Figure 5: Effect of the threshold τ on the model

Figure 5 shows the effect of different user similarity thresholds τ on the model performance on both datasets. Specifically, when $\tau = 0.5$, CRDG performs best on Yelp. When $\tau = 0.3$, CRDG performs best on Amazon. As τ gradually increases from 0.1 to 0.9, the model performance shows a trend of increasing and then decreasing. The reason behind is that too small τ will weaken the denoising ability of the model, while too large τ will remove a lot of effective information in the denoising process, which makes the amount of information that the model can obtain decrease, thus leading to a decrease in model performance. Therefore, a suitable τ needs to be set to ensure the model's performance.

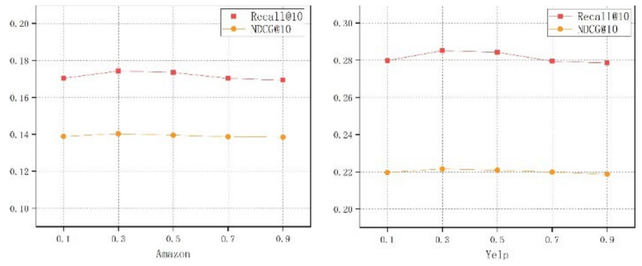

Figure 6: Effect of the threshold ξ on the model

Figure 6 shows the effect of the item relevance threshold on the model. Specifically, the model achieves the best performance on both datasets when $\xi = 0.3$. Similar to the user similarity threshold, the overall performance of the model shows an increasing and then decreasing trend as the threshold increases from 0.1 to 0.9, which is also due to the fact that when ξ is too small, it will lead to a decrease in the denoising ability, and when it is too large, it will lead to a decrease in the amount of information that can be obtained by the model.

5. Conclusion

In this paper, we propose a new context-based denoising recommendation model. This model constructs user similarity graphs and item relevance graphs to model the implicit similarity and relevance of users and items from their historical interactions. Then, the higher-order relation-aware graph neural network is used to learn the user similarity features and item relevance features. Considering the issue of inconsistent neighbors in context-based recommendation, we designed a context-aware denoising method. This method effectively filters out contextually inconsistent neighbors, improving the effectiveness of information aggregation. Finally, in order to refine the impact of different neighbors in the information aggregation process, we propose a dual-attention based consistent neighbor aggregation module to achieve adaptive propagation of information from different neighbors. We show through extensive experiments that much of the proposed model due to existing state-of-the-art methods and verify the effectiveness of the proposed scheme.

Declaration on Generative AI

The author(s) have not employed any Generative AI tools.

References

[1] Aljunid M F, Manjaiah D H, Hooshmand M K, et al. A collaborative filtering recommender systems: Survey[J]. Neurocomputing, 2025, 617: 128718.

[2] Koren Y, Bell R, Volinsky C. Matrix factorization techniques for recommender systems[J]. Computer, 2009, 42(8): 30-37.

[3] Vassøy B, Langseth H. Consumer-side fairness in recommender systems: a systematic survey of methods and evaluation[J]. Artificial Intelligence Review, 2024, 57(4): 101.

[4] Anand V, Maurya A K. A survey on recommender systems using graph neural network[J]. ACM Transactions on Information Systems, 2025, 43(1): 1-49.

[5] Van Den Berg R, Thomas N K, Welling M. Graph convolutional matrix completion[J]. arxiv preprint arxiv:1706.02263, 2017, 2(8): 9.

[6] Wang X, He X, Wang M, et al. Neural graph collaborative filtering[C]//Proceedings of the 42nd international ACM SIGIR conference on Research and development in Information Retrieval. 2019: 165-174.

[7] Yang L, Liu Z, Dou Y, et al. Consisrec: Enhancing gnn for social recommendation via consistent neighbor aggregation[C]//Proceedings of the 44th international ACM SIGIR conference on Research and development in information retrieval. 2021: 2141-2145.

[8] Rendle S, Gantner Z, Freudenthaler C, et al. Fast context-aware recommendations with factorization machines[C]//Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval. 2011: 635-644.

[9] He X, Liao L, Zhang H, et al. Neural collaborative filtering[C]//Proceedings of the 26th international conference on world wide web. 2017: 173-182.

[10] Kipf T N, Welling M. Semi-supervised classification with graph convolutional networks[J]. arXiv preprint arXiv:1609.02907, 2016.

[11] He X, Deng K, Wang X, et al. Lightgcn: Simplifying and powering graph convolution network for recommendation[C]//Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval. 2020: 639-648.

[12] Wu L, Li J, Sun P, et al. Diffnet++: A neural influence and interest diffusion network for social recommendation[J]. IEEE Transactions on Knowledge and Data Engineering, 2020, 34(10): 4753-4766.

[13] Fan Z, Xu K, Dong Z, et al. Graph collaborative signals denoising and augmentation for recommendation[C]//Proceedings of the 46th international ACM SIGIR conference on research and development in information retrieval. 2023: 2037-2041.