Adaptive Influence Diffusion Graph Neural Networks for Recommendation*

Zhiyong Hu[1*,†], Tao Shang[2,†]

[1] *Fiberhome Telecommunication Technologies Co.,LTD, Wuhan, China*
[2] *Xidian University, Xi'an, 710071, China*

Abstract

Recommender systems have become a crucial intelligent tool for providing personalized services to users. Existing recommendation algorithms often face the problems of sparse data and unbalanced distribution of user interactions because they rely on user behavior to generate data. In addition, since recommendation algorithms based on graph learning capture user inter-item interactions through equal aggregation of neighbor information, it leads to ignoring the heterogeneity between user and item nodes as well as the variability of the influence of different nodes on the target node. To address the above issues, we propose an adaptive influence diffusion graph neural network. Specifically, we derive the user similarity graph and the item related graph from the user-item bipartite graph, and model the diffusion of influence between similar users and related items through a diffusion model. In addition, to model the heterogeneity between nodes and the variability of influence between nodes, we design a dual-attention mechanism to assign different influence weights to relationships that do not use the same type of relationship, in order to achieve adaptive propagation of information between heterogeneous nodes and between different nodes. Experimental results on several real data sets demonstrate the superiority and effectiveness of the proposed model.

Keywords

recommendation system; graph diffusion； deep learning

1. Introduction

Collaborative filtering (CF) has become the most widely used technique in the recommendation domain by analyzing the users' historical interaction information to learn the users' and items' latent traits to predict the users' future preferences [1,2]. However, in real-world recommendation scenarios, CF-based recommendation algorithms often face the challenge of data sparsity [3]. This is because recommender systems rely on user behavior to generate data. However, most users generate a limited amount of behavioral data, which significantly impacts the efficiency of CF-based recommendation algorithms.

In recent years, graph neural networks (GNNs) [4] have shown great advantages in recommendation tasks by aggregating users' domain information. For example, GC-MC [5] and NGCF [6] construct user-item interaction bipartite graphs from user-item interaction data and utilize the structure of user-item graphs to propagate the embedding information of the user and the item on it. However, some existing methods treat all nodes in the domain equally, thus ignoring the differences between different neighboring nodes, resulting in suboptimal recommendation results.

Fortunately, as shopping platforms are updated, more and more people prefer to express their opinions about the items they have purchased on these platforms. These reviews often have a significant impact on the shopping behavior of similar users. This is due to the fact that an item may be more appealing to a user if the past consumers of the item have similar spending habits as the target user [7]. In addition, when users purchasing items, related items are also more attractive to

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.
zhiyonghu35@gmail.com (H. Zhiyong); 13545222153@163.com (T. Shang)
0009-0003-3032-5412 ((H. Zhiyong); 0009-0009-8644-4967 (T. Shang)

them. Thus, modeling user similarity and item synergies can help us represent target user preferences and assist us in constructing and learning user preferences to alleviate the data sparsity problem and improve recommendation performance.

In summary, we develop a new recommendation model that constructs user similarity collaborative graph and item related collaborative graph for users and items respectively from user-item history interactions in order to construct similarity relationships between users and correlations between items under interaction information. A higher-order influence diffusion model is also used to model the diffusion influence between similar users and the synergistic attraction between related items. In addition, considering the heterogeneous influences of neighbors and the differences in the influences of different nodes, we propose a dual-attention model to aggregate information for neighbors in order to refine their different influences on the target user, and to achieve adaptive information dissemination among different types of neighbor nodes. Finally, the diffused information and the representation obtained based on the attention mechanism are connected for recommendation prediction. We conducted extensive experiments on various real-world datasets and verified the effectiveness of the proposed method in representation learning.

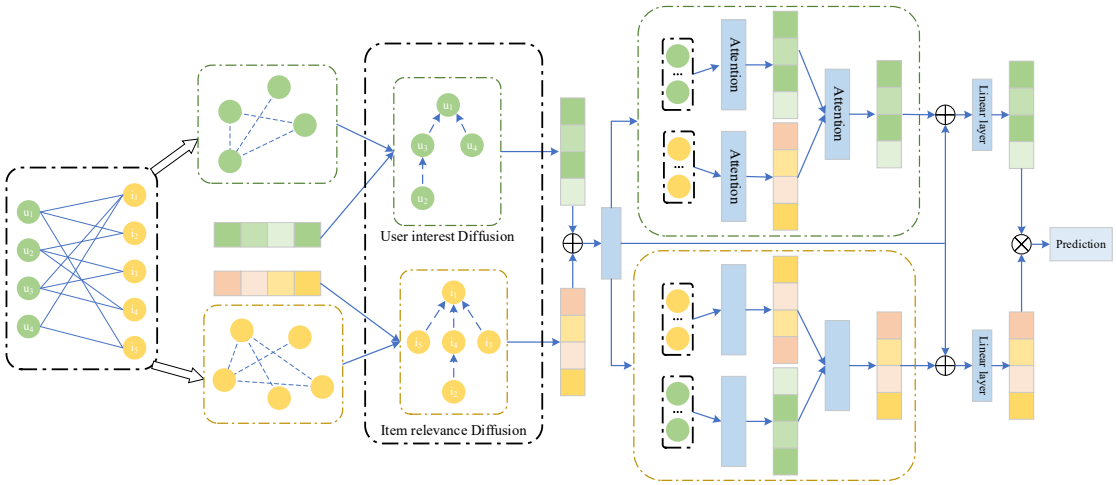

Figure 1: The AIDGR model framework

2. Formalization of problems

Based on the user interaction data we construct a two-part graph $B=(U,V,E_{UV})$ consisting of two different types of node sets (user set $U=\{u_1,u_2,\ldots,u_n\}$ and item set $V=\{v_1,v_2,\ldots,v_m\}$) and edge set E_{UV}, where $n=|U|$, $m=|V|$ are the number of users and items. We denote the interaction matrix as $R\in\mathbb{R}^{n\times m}$, where $r_{u,v}\in\{0,1\}$, $r_{u,v}=1$ denote that there is an interaction between user u and item v.

Formally, the recommendation algorithm aims to construct an interaction matrix $R\in\mathbb{R}^{n\times m}$ between users and items based on user-item interaction data. In this paper, we predict user-item interactions based on the original bipartite graph using the product of potential representations of users and items.

3. AIDGR model

Figure 1 shows an overview of AIDGR, which consists of three main modules. The first module is based on higher-order graph diffusion for feature representation In the module, we construct two auxiliary graphs based on the interaction information of users and items: user similarity graph and item relevance graph. Next, we employ a higher-order diffusion model to learn the initial user and item representations from the two auxiliary graphs. In the second module is dual-attention based

embedding aggregation, in this section we develop a dual-attention mechanism to assign different weights to neighbors, and finally we aggregate the neighbor information based on the weights to update the feature representations of users and items. The last module is the prediction module where we pass the diffusion based initial feature representation and aggregated neighbor feature representation into a fully connected network to generate the final feature representation. Finally the interaction between users and items is predicted based on the inner product of users and items.

3.1. Feature Representation Module Based on Higher-Order Graph Diffusion

In this section, we learn the initial representation of users and items from two collaborative graphs based on a higher-order diffusion model.

3.1.1. collaborative graph construction

Collaborative Graph Construction We construct two collaborative graphs, including user similarity graph $G_U^s = (U, E_U^s)$ and project related graph $G_V^r = (V, E_V^r)$, where E_U^s and E_V^r are the edge sets of the two graphs. We build the edges in E_U^s and E_V^r based on the Bipartite graph B construction. Specifically. We compute the similarity between users as follows:

$$Sim(u_i, u_j) = \frac{\left|N(u_i) \cap N(u_j)\right|}{\sqrt{\left|N(u_i)\right| \times \left|N(u_j)\right|}} \tag{1}$$

where $N(u_i)$ is the set of items that the user u_i interacts with, and if $Sim(u_i, u_j) > \eta$, we then insert an edge between u_i and u_j, where η is an adjustable threshold. Similarly, we compute the correlation between two items as follows:

$$Rel(v_i, v_j) = \frac{\left|N(v_i) \cap N(v_j)\right|}{\left|N(v_i) \cup N(v_j)\right|} \tag{2}$$

Where $N(v_i)$ denotes the set of users interacting with item v_i , if $Rel(v_i, v_j) > \varsigma$, we add an edge between v_i and v_j , and similarly ς is an adjustable threshold.

3.1.2. Higher-order diffusion models

In this module, we design T Layer Graph Diffusion to model the diffusion of potential feature representations over the collaborative graph. Given user $u \in U$, $x_u^{s,t}$ represents the potential feature representation of user u on G_U^s at the t layer. We update the representation of u on G_U^s at layer $t+1$ based on diffusion as follows:

$$x_u^{s,t+1} = \sigma\left(\sum_{u' \in \mathsf{N}_U^s(u)} (W_1^{s,t} x_{u'}^{s,t} + W_2^{s,t} (x_u^{s,t} \square\ x_{u'}^{s,t})) \right) \tag{3}$$

Where is $W_1^{s,t}$ and $W_2^{s,t}$ are the learnable parameters of the t layer, σ is the LeakyReLU activation function, $\mathsf{N}_U^s(u)$ denotes the neighbor of u on G_U^s, and u' is the neighbor node of u on G_U^s.Similarly, we also use a higher-order diffusion model to model the synergistic impacts between related projects on the G_V^r, and for a given project $v \in V$, we use the following diffusion to update the feature representation of the v :

$$x_v^{r,t+1} = \sigma\left(\sum_{v' \in \mathsf{N}_V^r(v)} (W_1^{r,t} x_{v'}^{r,t} + W_2^{r,t} (x_v^{r,t} \square\ x_{v'}^{r,t})) \right) \tag{4}$$

Where $W_1^{r,t}$ and $W_2^{r,t}$ are the learnable parameters of the t layer, $x_v^{r,t}$ denotes the potential feature representation of v on G_V^r at the t layer, σ is the LeakyReLU activation function, $\mathsf{N}_V^r(v)$ denotes the neighbor of v on G_V^r, and v' is the neighboring node of v on G_V^r.By diffusing the feature representations of the T graph diffusion layers we obtain a user representation $x_u^{s,T}$ and an initial feature representation $x_v^{r,T}$ of the item. These representations are then passed into a dual-attention based embedding aggregation for adaptive dissemination of user item information.

3.2. Embedding Aggregation Based on Dual Attention

After modeling the diffuse influence between similar users and related projects, to model the interaction information between users and projects, we introduce the user-project interaction graph B .Specifically, taking user u as an example, there are two different types of neighbor nodes for user u , i.e., user neighbor $\mathsf{N}_U^s(u)$ in G_U^s and project neighbor $\mathsf{N}_V^B(u)$ in B . Similarly, project v has project neighbor $\mathsf{N}_V^r(v)$ in G_V^r and user neighbor $\mathsf{N}_U^B(v)$ in B .In order to achieve the information transfer between different types of nodes, we introduce the learnable parameters W_U and W_V to map the initially obtained feature representations of the user and the project to the same embedding space, which are computed as follows:

$$\begin{cases} e_u^0 = W_U x_u^{s,T}, u \in U \\ e_v^0 = W_V x_v^{r,T}, v \in V \end{cases} \tag{5}$$

In order to refine the influence of different neighbors of users and projects, we propose a dual-attention model.

The impact of different neighbors on the same user. given that users u , $\mathsf{N}_U^s(u)$ are u 's friend neighbors in G_U^s. I aggregate the potential embeddings of these friend neighbors as follows:

$$e_{\mathsf{N}_U^s(u)}^l = \sum_{u' \in \mathsf{N}_U^s(u)} \alpha_{u,u'}^l e_{u'}^{l-1} \tag{6}$$

where $e_{u'}^{l-1}$ denotes the potential embedding of friend neighbor u' in the $l-1$ layer, and $\alpha_{u,u'}^l$ is the weight corresponding to u' , calculated as follows:

$$\alpha_{u,u'}^l = \frac{\exp\left(\sigma\left(W^l\left(e_u^{l-1} \oplus e_{u'}^{l-1}\right)\right)\right)}{\sum_{u'' \in \mathsf{N}_U^s(u)} \exp\left(\sigma\left(W^l\left(e_u^{l-1} \oplus e_{u''}^{l-1}\right)\right)\right)} \tag{7}$$

where W_u^l is the learnable parameter of the l layer , and σ is the LeakyReLU activation function. Similarly given user u , $\mathsf{N}_V^B(u)$ is the project neighbor of u on B . The potential embedding aggregation of these consistent project neighbors is as follows:

$$e_{\mathsf{N}_V^B(u)}^l = \sum_{v' \in \mathsf{N}_V^B(u)} \alpha_{u,v'}^l e_{v'}^{l-1} \tag{8}$$

$$\alpha_{u,v'}^{l}=\frac{\exp\left(\sigma\left(W^{l}\left(e_{u}^{l-1}\oplus e_{v'}^{l-1}\right)\right)\right)}{\sum_{v''\in \mathrm{N}_{V}^{B}(u)}\exp\left(\sigma\left(W^{l}\left(e_{u}^{l-1}\oplus e_{v''}^{l-1}\right)\right)\right)} \tag{9}$$

where $e_{v'}^{l-1}$ denotes the potential embedding of the project neighbor v' in the $l-1$ layer, $\alpha_{u,v'}^{l}$ is the weight parameter corresponding to v', W^{l} is the learnable parameter of the l layer, and σ is the LeakyReLU activation function.

Influence of different types of neighbor nodes. In response to the fact that different types of neighbors of the same user have different impacts, for this reason we propose a second-level attention mechanism for aggregating information obtained from two different perspectives: friend neighbors and project neighbors. Specifically, the user obtains the aggregated embedding as follows:

$$AGG_{u}^{l}=\beta_{\mathrm{N}_{U}^{S}(u)}^{l}e_{\mathrm{N}_{U}^{S}(u)}^{l}+\beta_{\mathrm{N}_{V}^{B}(u)}^{l}e_{\mathrm{N}_{V}^{B}(u)}^{l} \tag{10}$$

where $e_{\mathrm{N}_{U}^{S}(u)}^{l}$, $e_{\mathrm{N}_{V}^{B}(u)}^{l}$ are the aggregated embeddings of the friend neighbors and project neighbors of user CCC, respectively, $\beta_{\mathrm{N}_{U}^{S}(u)}^{l}$, $\beta_{\mathrm{N}_{V}^{B}(u)}^{l}$ are the attention weights of the friend neighbors and project neighbors, respectively, computed as follows:

$$\beta_{\mathrm{N}_{U}^{S}(u)}^{l}=\frac{\exp\left(\sigma\left(W^{l}\left(e_{u}^{l-1}\oplus e_{\mathrm{N}_{U}^{S}(u)}^{l-1}\right)\right)\right)}{\sum_{g\in \mathrm{N}_{U}^{S}(u)\cup \mathrm{N}_{V}^{B}(u)}\exp\left(\sigma\left(W^{l}\left(e_{u}^{l-1}\oplus e_{g}^{l-1}\right)\right)\right)} \tag{11}$$

$$\beta_{\mathrm{N}_{V}^{B}(u)}^{l}=\frac{\exp\left(\sigma\left(W^{l}\left(e_{u}^{l-1}\oplus e_{\mathrm{N}_{V}^{B}(u)}^{l-1}\right)\right)\right)}{\sum_{g\in \mathrm{N}_{U}^{S}(u)\cup \mathrm{N}_{V}^{B}(u)}\exp\left(\sigma\left(W^{l}\left(e_{u}^{l-1}\oplus e_{g}^{l-1}\right)\right)\right)} \tag{12}$$

Where W^{l} is the l layer learnable parameter and σ is the LeakyReLU activation function. The embedding of the final user u in the l layer is updated as follows:

$$e_{u}^{l}=\sigma\left(W^{l}\left(e_{u}^{l-1}\oplus AGG_{u}^{l}\right)\right) \tag{13}$$

where W^{l} is the l layer learnable parameter and σ is the LeakyReLU activation function. Similarly given the item v, we can get the updated embedding of v as described above:

$$e_{v}^{l}=\sigma\left(W^{l}\left(e_{v}^{l-1}\oplus AGG_{v}^{l}\right)\right) \tag{14}$$

Where W^{l} is the learnable parameter of the l layer, σ is the LeakyReLU activation function, and AGG_{v}^{l} is computed in a way similar to AGG_{u}^{l}.

3.3. Prediction and Model Optimization

After completing the contextually consistent neighbor information aggregation, we can obtain the layered embeddings of users and items, e_{u}^{l} and e_{v}^{l} where $l=[0,1,2,\ldots,L]$. In order to better model the potential features of users and items, we consider both the first and the last layers of user embeddings, because the first layer retains the original diffuse features of users and items, while the last layer provides a finer representation of the interaction features through the dual-attention mechanism for the item and user representations, and thus the final representations of users and items are:

$$e_{u}=\sigma\left(W_{u}\left(e_{u}^{0}\oplus e_{u}^{L}\right)\right) \tag{15}$$

$$e_{v}=\sigma\left(W_{v}\left(e_{v}^{0}\oplus e_{v}^{L}\right)\right) \tag{16}$$

where W_u and W_v are the learnable weight matrices and σ is the activation function. Finally, we obtain the recommendation results by the inner product of user and item feature representations:

$$\hat{r}_{u,v} = e_u \cdot e_v \tag{17}$$

In order to learn the AIDGR model parameters, we need to specify an objective function for optimization. For implicit feedback, the most widely used loss function is cross entropy, defined as:

$$\mathrm{L} = \sum_{u,v \in E_{UV}} r_{u,v} \log \hat{r}_{u,v} + \left(1 - r_{u,v}\right) \log(1 - \hat{r}_{u,v}) \tag{18}$$

where E_{UV} is the set of all observed ratings in the training set and $r_{u,v}$ is the value corresponding to the (u,v) pair in the interaction matrix R. In order to optimize the objective function, we use small batch adaptive moment estimation (Adam) [8] as the optimizer in our implementation, and a dropout strategy [9] to mitigate the overfitting problem.

3.4. Discussion

Space complexity. The model parameters consist of three parts, the user and item embedding $\Theta_1 = [\mathrm{P}, \mathrm{Q}]$, and the parameter set $\Theta_2 = \left[\left\{ W^{s,t}, W^{r,t} \right\}_{t=1}^{t=T}, \{W_U, W_V\}, Dual\ Attention(W^l)_{l=1,2,\ldots,L}, \{W_u, W_v\} \right]$. Since most embedding-based models [1] need to store embeddings for each user and item, the space complexity of Θ_1 is the same as that of traditional embedding-based models and grows linearly with the growth of users and items. For the parameters in Θ_2, which are shared among all users and projects, the dimension of each parameter is much smaller than the number of users and projects, so this additional storage cost is a negligible constant. Therefore, the space complexity of AIDGR from Θ_1 and Θ_2 is the same as the traditional embedding model。

Time complexity. The time complexity of the AIDGR model is designed into two main components: a feature representation module based on higher-order graph diffusion and an aggregated embedding module based on dual attention. Given M users and N items and T diffusion layers, it is assumed that the average number of interacted items per user is n_b and the average number of interacted users per item is m_b. Before entering into a feature representation model based on higher-order graph diffusion, we need to evaluate the similarity between users and the synergy between items, the modelling time cost of which can be expressed as $O(M^2 + N^2)$. Assume that the average number of similar users for users is m_s and the average number of collaborative items for items is n_r. Afterwards, the time spent on embedding updates of users and items based on the diffusion of higher-order graphs is $O(M \cdot m_s \cdot D + N \cdot n_r \cdot D)$. For the aggregated embedding module with dual attention at each layer, the main time cost is the transfer of information from neighboring nodes on different graphs to users and projects. Firstly, we need to compute the information of the neighboring nodes from different graphs, and its time spent is in computing the attention scores between each neighboring node, which has a time complexity of $O(M \cdot (m_s + n_b) \cdot D + N \cdot (n_r + m_b) \cdot D)$. Then, we aggregate information from different types of neighbouring nodes based on the attention mechanism, and the time complexity is approximated to $O(M + N)$ since there are only two types of neighbouring nodes in practice. In practice

$m_s, m_b, n_r, n_b \leq \min\{M, N\}$, the time overhead grows linearly with the number of users and items, and linearly with the diffusion depth and the number of dual attention layers. Therefore, the total time complexity of AIDGR is acceptable.

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

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

Table 1
Statistical dataset

Dataset	User #	Item #	Interaction #	Density
Yelp	32,654	34,193	1,347,861	0.00121
Amazon	52, 643	91, 599	2, 984, 108	0.00062

4.2. Evaluation indicators

To measure the performance of all methods, NDCG@K and Recall@K with K=10 are adopted. NDCG@K evaluates the ranking of the real items in the recommended list. Recall@K is the proportion of relevant items retrieved in the Top-K related items.

$$NDCG_k = \frac{DCG_k}{IDCG_k} \quad NDCG_k = \frac{DCG_k}{IDCG_k} \tag{19}$$

The above matrix sums the number of predicted items and divides it by the number of items in the test set corresponding to the user.

4.3. Baseline algorithm

The proposed model AIDGR is compared with the following baselines.

FM[10]: The model is a unified model based on latent factors, utilizing user and item attributes. In practice, we use user and item functions as described above.

NCF[11]: This is a deep learning based recommendation model that utilizes multi-layer perceptrons to learn user-item interaction functions.

GCN[12]: This model uses spectral graph convolution operators to learn local graph network structure and node features, and implements semi-supervised learning directly on graph-structured data. **NGCF[6]**: This is a graph-based recommendation model that models higher-order connectivity in the graph of user items and injects collaboration signals into the embedding process in an explicit manner. **LightGCN[13]**: This model is simplified from the standard GCN for the recommendation task and proposes a lightweight graph convolutional network **DiffNet++[14]**: This model achieves better performance in social recommendation tasks by modeling the user's interest and influence diffusion process. **DGCF[15]**: This model decomposes the embeddings of users and items into multiple semantic factors and unifies different semantics through an attention mechanism to model the diverse intentions behind user behaviors.

Table 2
Comparison of performance of all methods in Recall@10 and NDCG @ 10

Analysis of results	Yelp		Amazon	
	Recall@10	**NDCG@10**	**Recall@10**	**NDCG@10**
FM	0.2101	0.1751	0.1306	0.0951
NCF	0.2139	0.1677	0.1341	0.0962
GCN	0.2214	0.1734	0.1406	0.1025
NGCF	0.2435	0.1975	0.1466	0.1123
lightGCN	0.2579	0.1984	0.1521	0.1194
DiffNet++	0.2659	0.2058	0.1638	0.1203
DGCF	0.2708	0.2103	0.1657	0.1187
AIDGR	**0.2851**	**0.2215**	**0.1742**	**0.1403**

4.4. Analysis of results

4.4.1. Contrast to the baseline algorithm

Table 2 shows the comparison of the different methods on the two datasets. The following observations can be drawn from the results:

First FM, NCF performs poorly on these two datasets, probably because traditional collaborative filtering-based learning algorithms have difficulty in comprehensively dealing with the connectivity relationships between nodes compared to the strong modeling capabilities of graph structures on interacting data. The outperformance of DiffNet++ over traditional graph learning methods suggests that graph diffusion is able to cross the limitation of one-hop neighboring nodes in comparison to traditional graph learning methods, thus capturing richer graph properties. The superior performance of DGCF indicates that compared with traditional GNN models that model user behaviors based on a single intent, DGCF can better describe user preferences by modeling user behaviors with multiple intents.

Second, the proposed AIDGR model performance due to the baseline. The reasons are as follows, 1. AIDGR's higher-order diffusion operation based on the collaborative graph can effectively capture the influence between similar users and related items to enrich the feature representation of users and items, thus effectively alleviating the problem of data sparsity.2. The dual-attention mechanism can effectively differentiate the heterogeneity among nodes and the diversity of influences of neighboring nodes on the target node, and achieve adaptive information Propagation.

4.4.2. Ablation analysis

Influence of different components: the AIDGR consists of two key components, including 1) a higher-order influence diffusion model, and 2) a dual attention mechanism module. To study the impact of each component, we designed three AIDGR variants as follows:

AIDGR-Diff: removes the higher-order influence diffusion model from AIDGR.

AIDGR-Datt: replaces the dual-attention model with the normal GNN model.

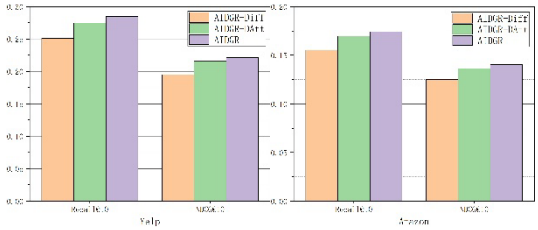

Figure 2: Contribution of each component on two datasets.

Figure 2 shows that AIDGR-Diff performs worse than AIDGR, which confirms the important role of higher-order influence diffusion in learning about potential users and item presentation. The diffusion model improves recommendation performance from two perspectives. First, the user similarity and item relevance maps contain useful information reflecting user influence and item collaboration appeal. Second, the diffusion process allows users and projects to aggregate information from implicit neighbors with similar interests and project attributes.

AIDGR also achieves higher performance than AIDGR-DAtt, which suggests that all neighbors cannot be considered equally in the information aggregation process. An explanation for this phenomenon is that not all neighbors have the same impact on users and it is crucial to distinguish between heterogeneous impacts.

4.4.3. Parametric sensitivity analysis

In this subsection, we investigate how the performance of our proposed model varies with a number of hyperparameters, including the embedding dimension d, the threshold η for the user similarity graph, and the threshold ς for the item correlation graph. experiments show that when d is very small, the performance usually increases with an increase. However, when d is larger than a specific value, the performance decreases. In addition, the optimal dimension d varies across datasets. AIDGR performs best on Yelp when $d=32$ and on Amazon when $d=64$. An explanation for this result is that larger values of d tend to lead to overfitting problems.

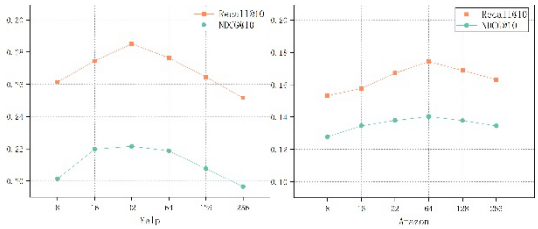

Figure 3: Effect of the embedding dimensionality d.

Figure 4 clearly shows the impact of different user similarity graph thresholds η validated in this paper based on two datasets. Specifically, when $\eta=0.5$, AIDGR performs best on Yelp. When $\eta=0.3$, AIDGR performs best on Amazon. As the threshold gradually increases from 0.1 to 0.9, the performance first increases and then gradually decreases. When $\eta=0.1$, the model introduces a lot of noise, which reduces its performance. When $\eta=0.9$, very little valid information is captured, which leads to performance degradation. Therefore, an appropriate η is needed to ensure the model performance.

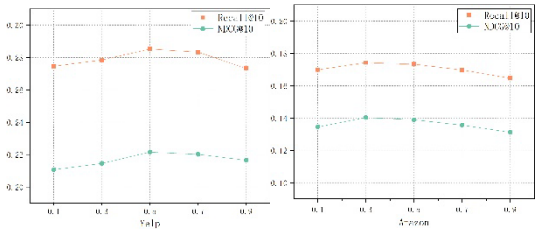

Figure 4: Effects of the threshold η.

Figure 5 shows the effect of thresholding ς on the item correlation graph, which is similar to the behavior of η. :When $\varsigma = 0.3$, the model performs best on Yelp and Amazon.

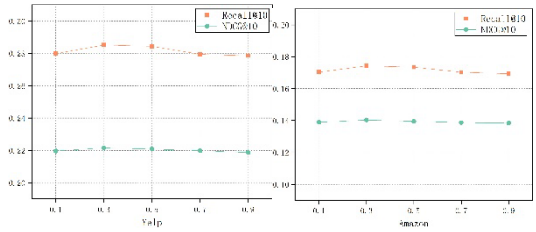

Figure 5: Effects of the threshold ς.

5. Conclusion

In this paper, we propose a new recommendation model, which constructs two synergistic similarity graphs for users and items from user-item history interactions to construct similarity relationships between users and correlation between items under the interaction information, and at the same time, uses a higher-order diffusion-of-influence model to model diffusion of influences between similar users and synergistic attraction between related items to alleviate the problems of data sparsity and user interactions that recommending is faced with. data distribution imbalance problem. In addition, considering the heterogeneity of neighbors and the differences in the influence of different neighbors, we propose a dual-attention model to perform information aggregation for neighbors to refine their different influences on the target nodes, and to achieve adaptive information dissemination among different neighbor nodes. Our extensive experiments show that the proposed model outperforms the 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] Koren Y. Factorization meets the neighborhood: a multifaceted collaborative filtering model[C]//Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining. 2008: 426-434.

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

[3] Adomavicius G, Tuzhilin A. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions[J]. IEEE transactions on knowledge and data engineering, 2005, 17(6): 734-749.

[4] Wu Z, Pan S, Chen F, et al. A comprehensive survey on graph neural networks[J]. IEEE transactions on neural networks and learning systems, 2020, 32(1): 4-24.

[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] Sarwar B, Karypis G, Konstan J, et al. Item-based collaborative filtering recommendation algorithms[C]//Proceedings of the 10th international conference on World Wide Web. 2001: 285-295.
[8] Diederik P. Kingma and Jimmy Lei Ba. 2015. Adam: A Method for Stochastic Optimization. In ICLR 2015 : International Conference on Learning Representations 2015.
[9] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. Journal of Machine Learning Research 15, 1 (2014), 1929–195.
[10] Rendle S. Factorization machines[C]//2010 IEEE International conference on data mining. IEEE, 2010: 995-1000.
[11] 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.
[12] Kipf T N, Welling M. Semi-supervised classification with graph convolutional networks[J]. arxiv preprint arxiv:1609.02907, 2016.
[13] 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.
[14] 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.
[15] Wang X, Jin H, Zhang A, et al. Disentangled graph collaborative filtering[C]//Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval. 2020: 1001-1010.