RMGAT: Bipartite Interaction Graph-Based Relation Fusion Multi-Head Graph Attention Recommendation Model

Junju Sun*, Longlong Zhang
School of Information and Communication Engineering, Xinyang Vocational and Technical College, Xinyang 464000, China
E-mail: gengxiaoyuan@byau.edu.cn, zhanglonglong@xyvtc.edu.cn
*Corresponding author

Keywords: graph representation learning, bipartite interaction graph, recommendation model, simulation experiment, multi-head graph attention, relation extraction

Received: April 17, 2025

Recommendation systems are crucial for accurately matching user needs in an environment with massive information. Traditional recommendation models have many deficiencies when dealing with user-item relationships and it is difficult to fully explore the information contained in user behaviors. This paper proposes a relation fusion multi-head graph attention recommendation model based on bipartite interaction graphs (RMGAT), aiming to innovate the performance of recommendation systems. The RMGAT model is based on the theory of user behavior, clearly dividing the interaction between users and items into explicit and implicit relationships, and constructing a bipartite interaction graph. By modeling these two types of relationships respectively, the explicit preferences with high credibility and the widely existing implicit latent interests can be effectively distinguished, overcoming the drawback of the traditional model in the semantic fuzzy processing of user behaviors. Meanwhile, the relational fusion multi-head graph attention mechanism is innovatively designed. With the help of the multi-head structure, the multiple semantic features of the explicit and implicit relationships are captured in parallel. The attention coefficient is used to dynamically adjust the weights to achieve the efficient complementary fusion of explicit and implicit information, significantly enhancing the model's expression ability for complex relationships, and thereby improving the accuracy and diversity of the recommendation results. After experimental verification on multiple typical datasets, the RMGAT model performs outstandingly in the recommendation scenarios of different fields and has achieved significant improvements in multiple key indicators compared with mainstream models. Future research will focus on developing lightweight multi-head mechanisms, introducing dynamic relationship modeling, and constructing cross-modal heterogeneous fusion frameworks, etc., in order to further optimize model performance and expand its application potential in more fields.

Povzetek: Članek predstavi priporočilni model RMGAT, ki gradi na bipartitnem grafu uporabnik–izdelek in ločeno modelira eksplicitne in implicitne odnose.

1 Introduction

The popularization and development of the Internet not only facilitates people to quickly obtain a large amount of information, but also results in information overload [1]. How to obtain valuable information from massive data quickly and efficiently has become a challenge in the scientific research field. Recommendation system is a new research direction in the field of machine learning. As a new technology to alleviate information overload [2-4], it can find information of interest to users according to user preferences from massive data information and recommend it to users, which has been successfully applied to various Internet platforms. On the other hand, the Internet platform has a huge user group and item information, as well as the historical interaction information between users and items, which provides strong data support for the research and development of the recommendation system. With the development of research on recommendation system, the performance of recommendation system is getting better and better. Users can quickly obtain items of interest according to their preferences, and the platform can also actively recommend items of interest to users through the recommendation system, so as to better enhance user experience and user viscosity. Relying on the use of high-performance servers for large-scale training, graph representation learning is a popular research field of machine learning in recent years. Advanced machine learning algorithms [5-7] are integrated to represent nodes in the graph as low-dimensional feature vectors using machine learning methods, and at the same time, node vectors represent structural information in the graph as much as possible. The recommendation system can make full use of a variety of interactive behavior data [8], mining relationship information from graph structure data, and learning the feature vector representation of users, items and related entities in graph structure data. The combination of graph representation learning and recommendation system can effectively enhance the learning ability of recommendation system, and then improve the accuracy of recommendation and user satisfaction. The recommendation system based on graph representation learning has high research value and is also the main trend in the field of recommendation system. In view of the limitations of the traditional attention mechanism in relation modeling, this study proposes a relation fusion multi-head graph attention mechanism. The different semantic features of the explicit and implicit relationships are captured in parallel by using the multi-head structure, and the weights of the two types of relationships are dynamically adjusted through the attention coefficient to achieve the complementary fusion of explicit and implicit information. This mechanism can effectively explore the diversity of users' interests, enhance the model's ability to express complex relationships, and improve the accuracy and diversity of recommendation results.

2 Related works

The recommendation model based on machine learning can process the recommendation with complex operations, find out the hidden patterns in the data, extract the features from the data set composed of users and their voting conditions, obtain the algorithm model, and then use the model to predict the recommendation [9]. However, the recommendation algorithm based on machine learning can only model the low-order feature representation in the user's historical interaction data, and lacks the modeling ability for the higher-order feature representation. Most of the data in a recommendation system can be regarded as graph-structured data in nature.

The current SOTA models in the field of recommendation systems (such as SGL, DGCF, and UltraGCN) have significantly enhanced the accuracy and robustness of recommendations under the GNN framework by improving representation learning and interaction modeling. SGL constructs multi-view node representations through "data augmentation + contrastive learning" to supplement the deficiencies of traditional supervised signals. Specifically, it includes: using node discarding, edge discarding and random walk to generate different views of the same node to alleviate the dominance of highly advanced nodes and noise interference. Maximize the similarity of different views on the same node, minimize the similarity of views on different nodes, and enhance the discrimination ability of node representations. However, data augmentation strategies need to be customized for different scenarios and overly rely on hyperparameter tuning. Self-supervised tasks increase computing costs, and scalability on large-scale data needs to be further verified. DGCF models the latent motives of user-item interaction through intention decoupling and generates intent-aware interaction graphs. Specifically, it includes: decomposing the user/item embedding into multiple blocks, with each block corresponding to a potential intention. Through iterative optimization of the intent-aware graph, high-impact interaction relationships are highlighted. Introduce distance correlation loss, enforce the independence of different intentions, and enhance interpretability. However, the computational complexity is high, the number of intentions needs to be preset, and the flexibility is insufficient. The ability to model the intentions of cold-start users/items is relatively weak. UltraGCN improves efficiency and accuracy by simplifying the message passing mechanism and multi-task learning. Specifically, it includes: directly optimizing the convergence conditions of GCN to avoid excessive smoothing caused by multi-layer stacking. Combine the principal loss of the user-item graph and the constrained loss of the item-item graph to capture the collaborative signal. Alleviate data sparsity and improve training stability through weighted negative samples. However, the modeling ability for high-order relations is limited and lacks dynamic adaptability. Relying on predefined graph structures makes it difficult to handle dynamic interactive data.

For example, the user's interaction with items, from the perspective of graph structure, once the user clicks or purchases the item, it is associated with these items. But the vast majority of deep learning tasks are conducted on Euclidean structured data. Based on the given interaction graph structure and initial node, capture the complex interaction relationship between users and item nodes [10, 11]. Graph representation learning uses user interaction graph topology and node attribute information to map each node to a low-dimensional vector representation, so as to improve the recommendation effect and effectively alleviate the problems of data sparsity and cold start [12].

The recommendation system based on the knowledge graph is to use the entity, attribute, relationship and other information in the knowledge graph to recommend resources or services that users may be interested in [13]. In the recommendation system integrated into the knowledge graph, the graph neural network follows the idea of neighborhood aggregation, aggregates the feature information of the neighbor entity in the knowledge graph, and then updates the feature representation of the central entity based on it, so as to realize the mining of multiple types and multiple relationships in the knowledge graph. The association information of knowledge attributes and the interaction and cooperation information between users and items are integrated into the feature representation process of users and items to effectively represent the potential semantics of users and items in vector space [14]. The first order interaction information between users and items is used to build the initial entity set of users and items, and the first order interaction information is explicitly encoded into the initial entity set through the co-propagation layer, so as to enhance the representation of users and items. By using the knowledge graph structure, the higher-order neighbors of the entities are extended, and the extended entity set is constructed to make the message propagate along the link. The dynamic embedding method of knowledge perception is used to improve the recommendation effect by generating different attention coefficients to focus on different head and tail entities.

For the four recommendation models related to this study, namely "GraphRec, KGAT, LightGCN, and PersonalRank", a brief comparison is shown in Table 1 [15-19].

The initialization of the attention head is the starting point of the training of GAT-type models. Its core objective is to provide a reasonable parameter starting point for the multi-head attention mechanism, which affects the convergence speed and representation ability of the model. Weight sharing in the attention mechanism is a key means to optimize the complexity of the model, mainly divided into two types: multi-head sharing and cross-layer sharing. Its design needs to balance the parameter scale and representation diversity. LightGCN is a representative graph convolution model in the field of recommendation systems in recent years, and its design concept contrasts sharply with the attention mechanism. The first one is the model's expressive ability. The attention mechanism captures "personalized neighborhood preferences" through dynamic weights. For example, in the user-item bipartite graph, different weights can be assigned to the neighbors of the same item for different users. This gives it an advantage on sparse data, but if the data noise is large, it may lead to the overfitting of the attention weight to the local noise. LightGCN relies on static Laplacian smoothing, assuming that adjacent nodes have similar characteristics. Although its linear structure limits the expressive ability, it shows stronger stability in large-scale data and is especially suitable for the high-concurrency scenarios of industrial-grade recommendation systems. Second, training efficiency and scalability. The multi-head computation of the attention mechanism introduces additional video memory overhead, and backpropagation requires the update of a large number of parameters, resulting in a high training cost in graph data with millions of nodes. The parameter-less design of LightGCN enables its training speed to be extremely fast. It can achieve linear complexity through sparse optimization of the adjacency matrix and is suitable for handling ultra-large-scale graphs with hundreds of millions of nodes. Thirdly, the applicable scenarios are differentiated. The priority scenarios of the attention mechanism include data sparsity and the need for high expressiveness, the need to capture fine-grained semantic differences, and high requirements for interpretability. The priority scenarios of LightGCN include large-scale collaborative filtering scenarios, those that are data-intensive and have high noise, and those that are sensitive to inference efficiency. The initialization of the attention head and the weight sharing strategy are essentially explicit controls over the "inductive bias" of the model: random initialization gives the model the maximum degree of freedom, pre-training initialization injects domain priors, and the sharing strategy guides the direction of representation learning through parameter constraints. Compared with LightGCN, the attention mechanism represents the "data-driven dynamic modeling" paradigm and is suitable for refined scenarios. LightGCN embodies the "structure-first static modeling" paradigm and is suitable for large-scale industrial applications.

Table 1: Hardware environment for simulation experiments

Model name	Model type	Advantages	Disadvantages	Applicable situation	Common datasets	Performance index
GraphRec	Recommendation model based on graph convolutional network	Effectively capture the higher-order connection relationships in the user-item graph; Model the interaction information using the graph structure	The computational complexity is relatively high, and multi-layer GCN may lead to over-smoothing problems	It is applicable to scenarios where the user-project interaction diagram structure is clear	MovieLens, Amazon, Yahoo! R3	Precision, Recall, NDCG, HR
KGAT	Recommendation based on knowledge graph attention network	Introduce external semantic knowledge in combination with the knowledge graph; Enhance interpretability	Relying on the integrity and quality of the knowledge graph, the cost of graph construction is relatively high	Scenarios where domain knowledge is needed to enhance recommendations	MovieLens-1M+KG, Freebase, DBpedia	Precision, Recall, MRR, Hit Rate

		through the attention mechanism				
LightGCN	Recommendation model based on lightweight graph convolutional networks	Simple structure, few parameters and high training efficiency; Suitable for large-scale sparse data	The expression ability is limited, and the performance improvement of the deep network is not obvious	Resource-constrained scenarios or large-scale recommendation systems that require efficient reasoning	MovieLens, Douban, Yahoo! R3	Recall@K, NDCG@K, Training time
PersonalRank	Personalized recommendation model based on random walk	Be good at capturing users' personalized preferences; It performs well on sparse data	The time complexity is high and the random walk efficiency in large-scale graphs is low	Scenarios with strong personalized demands and sparse data	Small-scale user-project interaction data	Precision, Recall, Average path length

3 Recommendation algorithm model

The recommendation model in this paper is based on the graph attention network (RMGAT). The RMGAT model consists of four modules. First, it preprocesses the interaction scores between users and items. Secondly, the implicit relationship embedding vector representation between users and items is mined according to the user-item interaction bipartite graph. Third, extract the explicit relationship between the user and the item from the graph. Fourth, the implicit relation vector and the dominant relation vector of the node are fused, and the prediction score is calculated to get the recommendation result. The model framework is shown in Figure 1.

For the preprocessing of interactive scores between users and items, it is necessary to unify the scoring habits of users, so the first module is to preprocess the scores in the two-part graph. According to the bipartite graph after the score preprocessing, the corresponding user implicit relationship graph and item implicit relationship graph are generated. For the user implicit relationship graph, the user node embedding vector is used to learn the stacked multi-layer attention network. For the hidden relationship diagram of items, it is also necessary to fuse the auxiliary information of items, and get the embedding vector of items by stacking multiple layers of attention network. The third module is to learn the nodes in the bipartite graph through the attention network, and get the explicit relation embedding vector representation of each node. The fourth module obtains the node embedding vector of the final user and the item through the fusion of the vector, so as to make the score prediction and finally get the recommendation result. The key tensor dimensions are shown in Table 2.

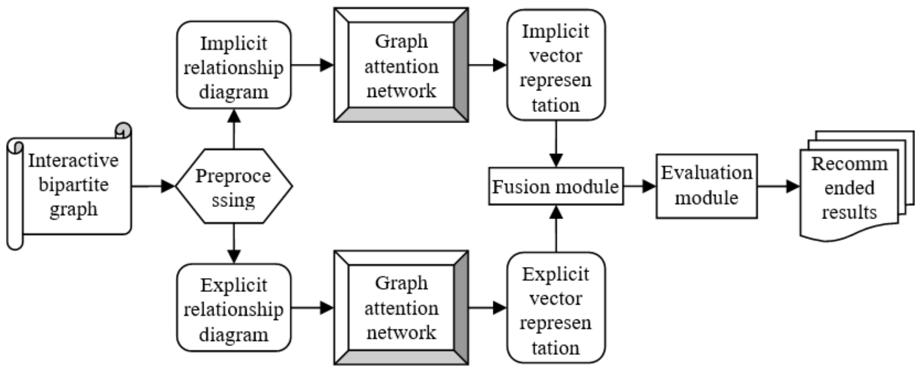
Interactive bipartite graph
Preproce ssing
Implicit relationship diagram
Graph attention network
Implicit vector represen tation
Explicit relationship diagram
Graph attention network
Explicit vector represen tation
Fusion module
Evaluation module
Recomm ended results

Figure 1: RMGAT model framework

Table 2: Key tensor dimension

Model name	Tensor name	Dimension
Input module	User characteristic matrix X_u	$N_u \times d_x$
	Item feature matrix X_i	$N_i \times d_x$
	Interaction adjacency matrix A	$N_u \times N_i$

Preprocessing module	User embedding H_u	$N_u \times d_c$
	Item embedding H_i	$N_i \times d_c$
Vector representation module	Query vector Q	$N \times d_k$
	Key vector K	$N \times d_k$
	Value vector V	$N \times d_v$
	Attention output Z	$N \times d_v$
Explicit relation extraction module	Collaborative filtering score S	$N_u \times N_i$
	Path similarity P	$N_u \times N_i$
	Dominant feature fusion F	$N_u \times N_i \times d_f$
Fusion module	Implicit feature Z_h	$N_u \times N_i \times 64$
	Explicit characteristics Z_s	$N_u \times N_i \times 32$
	Fusion feature Z_f	$N_u \times N_i \times 96$
	Predicted score Y	$N_u \times N_i \times 1$

3.1 Recessive relation extraction

Explicit relationship extraction uses the idea of autoencoder, in which the feature of the item is transmitted to the user by means of information transfer, and the score is predicted by a double-line decoder. Auto-Encoder (AE) is a very useful and flexible deep learning model, which includes parts of automatic encoding and automatic decoding, can be trained on unlabeled data, can effectively extract and represent new data, and can be used in combination with other deep learning models [20].

Assume that the input sample $X = R^{d \times n}$ is given and there are two biases, namely, encoding layer bias b_n and decoding layer bias b_d. W is used as the weight matrix of the sample and $g(\cdot)$ is used as the node activation function of the model. The coding formula of AE is as follows:

$$H = g(WX + b_n) \tag{1}$$

The sample will be reconstructed by encoding H, represented by $\hat{X}$, the process is called the decoding of AE, the formula is:

$$\hat{X} = g(W^T H + b_d) \tag{2}$$

The training of AE is designed to minimize the loss function:

$$arg \min_{W,b} J(W, b) \tag{3}$$

The square error and cross entropy loss function of both the input sample and the reconstructed sample can be used as the loss function of AE. The formula is as follows:

$$J(X, \hat{X}) = \frac{1}{2}\sum_{i=1}^{n} \|\hat{x}_i - x_i\|_2^2 \tag{4}$$

$$J(X, \hat{X}) = -\sum_{i=1}^{n} [x_i \, log(\hat{x}_i) + (1 - x_i) \, log(1 - \hat{x}_i)] \tag{5}$$

In general, in order to learn important abstract features in the sample data, AE usually uses gradient descent algorithms to adjust the network parameters, fine-tune the iteration through backpropagation errors, and gradually reduce the reconstruction error function to a minimum. When using the gradient descent algorithm, a learning rate η is usually set, and the connection weights and biases of AE are updated according to the update rules:

$$W = W - \eta \frac{\partial J(W,b)}{\partial W} \tag{6}$$

$$b = b - \eta \frac{\partial J(W,b)}{\partial W} \tag{7}$$

Multiple AE are typically cascaded to build a Stack Auto-Encoder (SAE). SAE is a deep neural network model which is cascaded by multi-layer autoencoders. The main function of SAE is to extract features by means of multiple training so as to enrich the hidden abstract information of extracted features. Due to the nature of AE, SAE has symmetry, with the first half of the network being a multi-layer encoder and the second half being a multi-layer decoder. During training, the encoder at each layer converts the input into a lower-dimensional representation of the feature, which is then passed on to the decoder at the next layer for reconstruction. After the final training is completed, the coding layer of SAE can be used as a new feature for tasks such as classification or regression. Because the coding layer extracts the most representative features and has a lower dimension, features extracted by SAE can achieve better performance in classification or regression problems. At the same time, in these tasks, the decoding part is generally omitted, and only the final encoded output is retained for classification or regression.

Graph Auto-Encoder (GAE) transmits and encodes information on graph structured data, aggregating information of neighborhood nodes in a convolution like manner. GAE does not apply regular data, but graph-structured data. GAE can model the relationship between users and products, helping recommendation systems better understand user behavior and product attributes. In order to carry out information transfer, information aggregation and feature extraction on graph structured data, GNN is proposed. As a GNN, GAE integrates the advantages of autoencoders and graph neural networks. It takes the input information as the learning target, and carries out network embedding or representation learning on the output information.

Suppose that given a graph G(V,E), the node features are represented by X, $X = R^{N\times D}$ is the eigenmatrix of the nodes, where N is the number of nodes. A represents the adjacency matrix, representing whether there is a connection between nodes, if there is a connection, the value is 1, otherwise the value is 0. The decoded part is represented as:

$$Z = GCN(X,A) = \hat{A}\, ReLU\left(\hat{A}XW_0\right) W_1 \tag{8}$$

In the above formula, W_0 and W_1 are parameters to be learned. As can be seen from the formula, GAE can well preserve the structure of the graph and extract features by inputting the topological structure of the graph and the information of the node itself. In the decoding part of the graph self-encoding, the method of simple inner product is generally used to obtain the reconstructed adjacency matrix:

$$\hat{A} = \delta(ZZ^T) \tag{9}$$

To sum up, GAE is a deep learning model for extracting features from graph-structured data, the main purpose of which is to project graph-structured data into a low-dimensional potential space for easy analysis and processing. By using GCN as a decoder, GAE can capture the nonlinear structure of the network well, and reduce the dimension of the graph structure data, improving the computational efficiency, while retaining the main characteristics of the data. As a result, GAE is widely used in areas such as social network analysis, image recognition, and recommendation systems.

3.2 Dominant relation extraction

In recommendation systems, the division of "explicit/implicit relationships" stems from the theoretical understanding of the essential attributes of user-item interaction data, and its theoretical basis can be traced back to the basic frameworks in fields such as collaborative filtering, graph representation learning, and knowledge reasoning. The explicit relationship refers to the directly observable explicit interaction behavior, and its existence is reflected through the explicit edge connections in the bipartite graph. Implicit relations refer to the potential associations inferred through the implicit patterns of data, which need to be revealed by means of the high-order neighborhood or embedded space semantic modeling of graph structures. At the level of information fusion, the dichotomy of explicit/implicit relations inherits the core idea of the multi-source information fusion theory. Explicit relations provide factual input as low-level perceptual data, and implicit relations supplement semantic interpretation as high-level cognitive features. This hierarchical architecture is theoretically consistent with the ontology-instance hierarchical structure in the knowledge graph, where explicit edges correspond to explicit ontology relationships and implicit edges map implicit semantic associations. Explicit relationships dominate the modeling of short-term interests, while implicit relationships are more explanatory for the mining of long-term interests. This division, through the topological reconstruction of the bipartite interaction graph, enables the model to simultaneously capture the surface patterns of behaviors and the deep interest associations, meeting the balance requirements of "behavior interpretability - semantic completeness" in the field of recommendation systems.

In interactive bipartite graphs, the dominant relationship refers to the edge between two different kinds of nodes. The explicit relationship generally contains a wealth of interactive information, and the hidden features of the project can transmit information to the user from the interaction, which is helpful to understand the user's preferences. Therefore, the method of information transfer and autoencoder can be used to aggregate the hidden features of the items that users interact with in the encoder stage, reconstruct the information in the decoder, and then use the reconstruction error to train and learn the model.

On the encoder side, the user's embedding vector collects information from the interacted items, and the formula is:

$$h_{u_i} = \sigma\left[\Big\| r \in R\left(\textstyle\sum_{v_j\in N_i} \mu_{v_j\to u_{i,r}}\right)\right] \tag{10}$$

Among them, the explanations of each symbol of this formula are shown in Table 3.

Table 3: The symbolic explanation of formula (10)

Symbol	Explanation
u_i	user
v_j	project
N_i	neighborhood set of node users
$\mu_{v_j\to u_{i,r}}$	information flowing from the project to the user
R	type of the edge connected between the user and the project
$\|\|$	association symbol
σ	activation function

With c as the regularization constant, $\mu_{v_j\to u_{i,r}}$ can be expressed as:

$$\mu_{v_j\to u_{i,r}} = \frac{1}{c} W_r x_{v_j} \tag{11}$$

After collecting this explicit relationship information, it is necessary to integrate the feature representation of the implicit relationship processed by the graph attention mechanism to conduct joint learning. The integration method is as follows:

$$u_i = \sigma\left(Wh_{u_i} + W_2 n_i\right) \tag{12}$$

Both W and W_2 are trainable parameters, n_i is an embedded representation of the user's features with implicit relationships calculated by the graph attention mechanism, and the resulting representation of the user's features calculated by the encoder. The characteristic representation of the item can be obtained in the same way.

The main task of the decoder side is to obtain the probability distribution of the score with the Softmax function, the formula is:

$$p(\hat{y}_{ij} = r) = \frac{e^{u_i^T Q_t v_j}}{\sum_{t \in R} e^{u_i^T Q_t v_j}} \tag{13}$$

where, $\hat{y}_{ij}$ is the predicted fraction and Q is the trainable parameter. The formula for calculating $\hat{y}_{ij}$ is as follows:

$$\hat{y}_{ij} = \sum_{r \in R} r p(\hat{y}_{ij} = r) \tag{14}$$

Finally, the goal of model training is to minimize the gap between predicted and actual scores. The formula is as follows:

$$L = -\sum_{r \in R} I(r = \hat{y}_{ij}) \log p\,(\hat{y}_{ij} = r) \tag{15}$$

If $r = \hat{y}_{ij}$, $I(r = \hat{y}_{ij}) = 1$, otherwise it's equal to 0.

3.3 The fusion of explicit and implicit features

Through the fusion of explicit and implicit features, multi-level modeling of user-item interaction has been achieved. Its core innovation lies in jointly optimizing graph embedding and explicit features, and enhancing the representation ability through nonlinear transformation. The symbol definitions involved in the mathematical formula and the derivation process are as follows:

Implicit relation vector: $h \in \mathbb{R}^{d_h}$. Explicit relation vector: $h \in \mathbb{R}^{d_s}$. Fusion weight matrix: $W_f \in \mathbb{R}^{d_f \times (d_h + d_s)}$. Fusion bias term: $b_f \in \mathbb{R}^{d_f}$. Score prediction weight vector: $W_p \in \mathbb{R}^{1 \times d_f}$. Score prediction bias term: $b_p \in \mathbb{R}$.

Step 1: Concatenate recessive and dominant vectors

Concatenate the recessive vector and the dominant vector into a joint feature:

$$z = [h, s] \in \mathbb{R}^{d_h + d_s} \tag{16}$$

Explicit relationships (such as co-occurrence times and collaborative filtering scores) and implicit relationships (as shown in the embedded semantics in the figure) are complementary. Direct splicing can retain the original feature space structure.

Step 2: Feature fusion and nonlinear transformation

Fuse features through the fully connected layer and the activation function:

$$z' = LeakyReLU(W_f z + b_f) \tag{17}$$

If the explicit/implicit dimensions are different, projections need to be made on h or s.

Step 3: Rating prediction

Map the fused features to the predicted scores:

$$\hat{y} = W_p z' + b_p \tag{18}$$

The scoring range is a continuous value and is compressed to [0,1] using Sigmoid:

$$\hat{y} = \sigma(W_p z' + b_p) \tag{19}$$

If it is binary classification, output the probability value directly.

The complete formula is integrated as:

$$\hat{y} = W_p \cdot LeakyReLU(W_f \cdot (h; s) + b_f) + b_p \tag{20}$$

3.4 Graph Attention Network (GAN)

Attention Mechanism (AM) focuses on the three words "attention", indicating that it draws on human visual attention mechanism. When people's eyes move from one area of the picture to the next area, their attention will follow the eyes to get information about the next area. This means that people selectively observe things, paying more attention to the important parts to get detailed information, and ignoring the non-important parts of the information. The ability of human to extract important information from a large amount of information with limited attention is the visual attention mechanism, which greatly improves the efficiency and accuracy of human information screening. AM in deep learning is inspired by the visual attention mechanism and attempts to extract the most useful information from many features information to achieve the purpose of the model. Heterogeneous graph attention networks are becoming a popular option that has surpassed traditional Gans [21].

The attention mechanism inputs a sequence and outputs a non-sequence, and is often used for text classification and recommendation system tasks. The generator of GAN uses variational encoder to obtain synthetic sample based on prior distribution [22]. Similar to GNN, both GAN and GNN are looking for aggregate function description to extract the feature representation of nodes and their neighbors. The difference is that GAN uses AM to redistribute the weight of neighbor nodes to target nodes. The calculation formula is as follows:

$$h_i^t = \sigma\left(\sum_{j \in N_i} \alpha(h_i^{t-1}, h_j^{t-1}) W^{t-1} h_j^{t-1}\right) \tag{21}$$

where, $\alpha(\cdot)$ represents the attention function, W represents the attention weight, and σ represents the activation function. The above formula is typical self-attention, which summarizes the features required for learning the model from the same set of nodes and assigns different weights to these features. In order to make full use of the information of nodes and their neighbors, the attention network computes attention in different subspaces in parallel, and the multi-head self-attention is proposed:

$$h_i^t = \Vert_{k=1}^{K} \sigma\left(\sum_{j\in N_i} \alpha\left(h_i^{t-1}, h_j^{t-1}\right) W^{t-1} h_j^{t-1}\right) \tag{22}$$

$$\vec{h}_i' = \sigma\left(\frac{1}{K}\sum_{k=1}^{K}\sum_{j\in N_i}\alpha_{ij}^{k} W^{k}\vec{h}_j\right) \tag{23}$$

The comparison between self-attention and multi-head self-attention is shown in Figure 2.

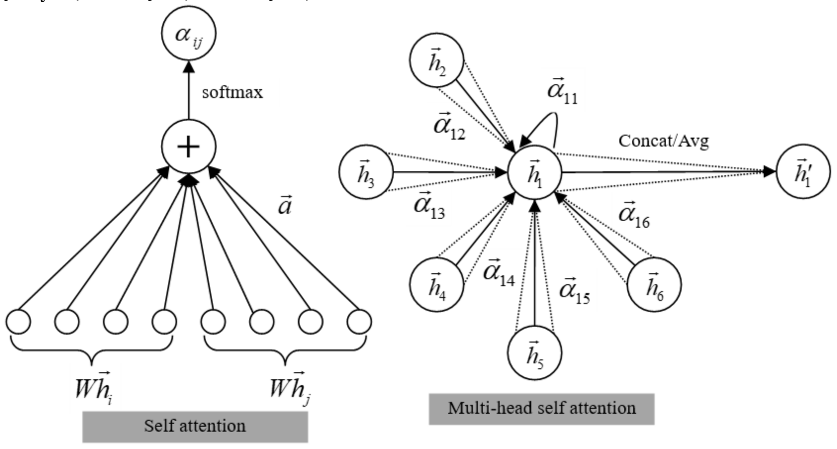

Figure 2: Comparison between attention mechanism and multi-head attention

For the same node, the multi-head self-attention calculates the *K* times of attention respectively, and fuses the *K* times of attention together by concatenating or averaging. By calculating attention multiple times, multi-head self-attention can more deeply tap the potential of node data, so that the model can better understand the characteristic meaning of nodes. The disadvantage is that the calculation amount of the model is increased and the training time of the model is longer.

3.5 Similarity calculation

Similarity calculation is one of the core components for connecting the interaction relationship between users and items and mining potential associations. In the multi-head graph attention layer, the similarity score serves as additional context information to guide the calculation of attention weights. Traditional GAT learns attention through node features and neighborhood structures, but similarity calculation can provide more direct semantic association clues. The autoencoder is responsible for compressing the latent features of users and items from high-dimensional sparse data and generating low-dimensional embedding vectors. The similarity calculation is based on these embedding vectors to directly measure the matching degree between users and items, forming a dual-path structure of "semantic feature extraction + explicit association calculation". In the fusion module of the model, the similarity calculation results are used as independent feature channels for cross-modal fusion with the structural features output by GAT and the semantic features output by AE.

For the user preference information reflected in the two-part graph, it is necessary to use different similarity calculation methods for explicit feedback data and implicit feedback data. Use cosine similarity or Pearson correlation coefficient for explicit feedback data:

$$Sim(i,j)\ cos(i,j) \frac{\sum_{u\in S_{i,j}} r_{u,i} r_{u,j}}{\sqrt{\sum_{u\in S_{i,j}} r_{u,i}^2}\sqrt{\sum_{u\in S_{i,j}} r_{u,j}^2}}_{cos} \tag{24}$$

$$Sim_{pcc}(i,j) = \frac{\sum_{u\in S_{i,j}}\left(r_{u,i}-\bar{I}_i\right)\left(r_{u,j}-\bar{I}_j\right)}{\sqrt{\sum_{u\in S_{i,j}}\left(r_{u,i}-\bar{I}_i\right)^2}\sqrt{\sum_{u\in S_{i,j}}\left(r_{u,j}-\bar{I}_j\right)^2}} \tag{25}$$

where, $r_{u,i}$ stands for user *u*'s score on project *i*, $r_{u,j}$ stands for user *u*'s score on project *j*, $\bar{I}_i$ stands for the average score of project *i*, $\bar{I}_j$ stands for the average score of project *j*, and $S_{i,j}$ stands for the set of users who have scored both project *i* and project *j*.

If the item contains only structured information but does not have symbolic attribute features, and the components in the item feature vector are all numerical values, cosine similarity or Pearson correlation coefficient is used to calculate the item's attribute feature similarity. The calculation formula is as follows:

$$Sim_{vt}(I_1, I_2) = cos(I_1, I_2) = \frac{I_1 I_2}{\Vert I_1\Vert \Vert I_2\Vert}\frac{\sum_{t=1}^{n} p_t q_t}{\sqrt{\sum_{t=1}^{n} p_t^2}\sqrt{\sum_{t=1}^{n} q_t^2}} \tag{26}$$

$$Sim_{vt}(I_1, I_2) = \frac{\sum_{t=1}^{n}(p_t-\bar{p})(q_t-\bar{q})}{\sqrt{\sum_{t=1}^{n}(p_t-\bar{p})^2}\sqrt{\sum_{t=1}^{n}(q_t-\bar{q})^2}} \tag{27}$$

where, $\bar{p}$ represents the component mean of the eigenvector I_1, and $\bar{q}$ represents the component mean of the eigenvector I_2. Due to the difference in the numerical scale of different components, it is easy to cause calculation deviation. Usually, the value under each component is divided by the maximum value under the component, so that the value range of each component in the feature vector is restricted to 0 to 1.

For implicit feedback data, since there is no rating data, the degree of similarity between the two items is calculated by measuring the number of users who have viewed or purchased the two items at the same time. In order to make the similarity value range between 0 and 1, its value is standardized, and the calculation formula is as follows:

$$Sim(i,j) = \frac{Num_{i,j}}{max_{i,j\in S}\{Num_{i,j}\}} \tag{28}$$

where, $Num_{i,j}$ represents the number of users who have performed implicit feedback on items i and j at the same time, and S represents the set of all items. In practical application, implicit feedback data can be converted to explicit score if there is a reasonable method to convert score data.

3.6 The visualization process of the core module framework and attention mechanism

The core module of the RMGAT model works in coordination with the attention mechanism to achieve the complete process from raw data to precise recommendations. The modular hierarchical design clearly distinguishes the stages of data processing, feature generation, information fusion and result output. The core computational logic (attention mechanism, vector fusion) is magnified and displayed through subgraphs to highlight the key points of the model. The visualization process of the core module framework and the attention mechanism is shown in Figure 3.

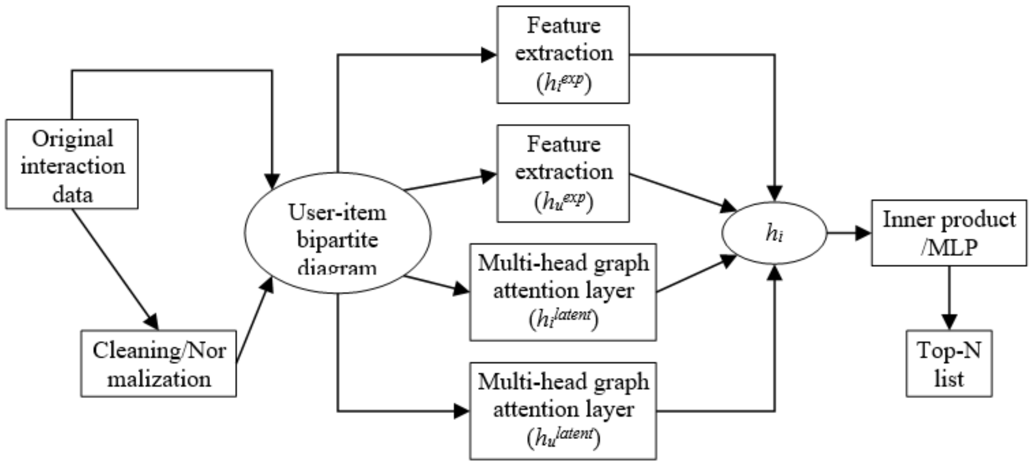
Original interaction data
Cleaning/Normalization
User-item bipartite diagram
Feature extraction (h_i^{exp})
Feature extraction (h_u^{exp})
Multi-head graph attention layer (h_i^{latent})
Multi-head graph attention layer (h_u^{latent})
h_i
Inner product /MLP
Top-N list

Figure 3: The visualization process of the core module framework and attention mechanism

Firstly, in the preprocessing module, the original user-item interaction rating data undergoes operations such as cleaning and normalization to remove noise and outliers. Subsequently, it is constructed as a user-item bipartit graph to provide a structured data basis for the subsequent modules. Then, the vector representation module uses the multi-head graph attention mechanism to mine the implicit relationships in the data. Each attention head independently calculates the attention weight between the user and the neighborhood items, and this weight reflects the closeness of the interaction between the nodes. Through the weighted aggregation of neighborhood features, implicit relationship embedding vectors are generated for each user and item. Multi-head parallel computing enables the model to capture complex relationships from multiple dimensions and enhance the feature expression ability. The relationship extraction module focuses on extracting explicit features from bipartite graphs, including information such as user attributes, item categories, and direct interaction ratings, and converting them into explicit relationship vectors. This explicit information provide intuitive reference basis for the model. Finally, the prediction module performs weighted fusion of the implicit relation vector and the explicit relation vector through learnable fusion coefficients to obtain the final user and item representation vectors. The predicted ratings of users for items are calculated through inner product operations or multi-layer perceptrons, and a Top-N recommendation list is generated based on the ratings from high to low. Throughout the entire process, the attention mechanism runs through the vector representation module, dynamically allocating the attention weights between nodes. The four modules are interlinked, achieving the complementarity of implicit and explicit information. It not only explores potential correlations but also utilizes intuitive features, thereby enhancing the accuracy and reliability of recommendations.

4 Simulation experiment and result analysis

Simulation experiments were conducted to verify the RMGAT recommendation model constructed in this paper, including the following steps: building the experimental environment, selecting the data set, conducting the simulation experiment, and analyzing the experimental results.

4.1 Solution for cold start problem

The cold start problem is the phenomenon of model performance degradation in the recommendation system due to the lack of sufficient interaction data of new users or new items. Specifically, it is manifested as follows: User cold start refers to the situation where newly registered users have no historical behavior records, and the system has difficulty capturing their interest preferences. Cold start of items refers to the situation where newly listed items have no user interaction data and cannot obtain recommendation basis through traditional methods such as collaborative filtering. Both types of cold starts lead to an intensification of data sparsity, making it impossible for models based on matrix factorization and graph neural networks to effectively learn node representations, thereby reducing the accuracy and coverage of recommendations.

Cold-start users adopt two processing schemes: feature transfer based on user attributes and rapid adaptation through meta-learning.

For new users, the attribute vector $u_{attr} \in \mathbb{R}^m$ is constructed using their registration information, and the initial embeddings are generated through the pre-trained attribute-preference mapping model:

$$h_u^{init} = g_\theta(u_{attr}) \tag{29}$$

Among them, g_θ can be implemented by using a multi-layer perceptron, and θ is the model parameter. Combined with the graph attention mechanism, the implicit relation vectors of the initial embeddings and similar users in the graph are fused:

$$h_u^{latent} = GAT\left(h_u^{init}, N_u^G\right) \tag{30}$$

Among them, N_u^G is the neighbor set of similar users, which is filtered through the similarity of user attributes.

The rapid adaptation of meta-learning utilizes the meta-learning framework to train meta-models on historical cold-start user data and learn the rapid mapping relationships between different user attributes and preferences. For new users, the meta-model can quickly adjust parameters based on a small number of initial interactions:

$$\theta' = \theta - \alpha \cdot \nabla_\theta L\left(h_u^{latent}, y_{u,i}\right) \tag{31}$$

Among them, α is the learning rate, L is the loss function, and $y_{u,i}$ is the actual interaction label of user *u* on item *i*.

Cold-start items adopt two processing schemes: initialization based on content features and knowledge graph enhancement.

Based on the initialization of content features, first extract the content information such as the text description and image features of the item, generate the initial embedding $i_{content} \in \mathbb{R}^n$ through the pre-trained encoder, and then integrate it into the model through linear transformation:

$$h_i^{init} = W_c \cdot i_{content} \tag{32}$$

Among them, W_c is the learnable parameter matrix. Combined with explicit relation extraction, the content features are fused with explicit information such as category labels:

$$h_i^{explicit\left(h_i^{init}, i_{label}\right)} \tag{33}$$

For knowledge graph enhancement, an external knowledge graph is introduced, and the semantic associations in the knowledge graph are learned through GCN:

$$h_i^{kg} = GCN\left(G_{kg}, i_{node}\right) \tag{34}$$

Among them, G_{kg} is the knowledge graph, and i_{node} is the node embedding of the item in the graph. Integrate the knowledge graph embedding with the implicit and explicit vectors of the model.

$$h_i = \beta_1 \cdot h_i^{latent} + \beta_2 \cdot h_i^{explicit_3{}_i^{kg}} \tag{35}$$

Among them, $\beta_1, \beta_2, \beta_3$ is the integration vector.

where $\beta_1, \beta_2, \beta_3$ denotes the integrated vector.

4.2 Data sparsity processing scheme

Due to the significantly insufficient amount of interaction data between users and items, the phenomenon occurs where the model has difficulty capturing effective features and potential associations. The root causes include sparse user behavior, cold start of items, and implicit feedback noise, which leads to a decline in the model's overfitting and generalization capabilities, manifested as an increase in the recommendation deviation of long-tail items and limited system scalability. Especially in dynamic scenes, user interest drift and item updates further intensify data sparsity, causing the model based on graph neural networks to fail due to insufficient learning of node representations. In this study, three processing schemes for data sparsity were adopted, namely multi-view data enhancement, transfer learning cross-domain collaboration, and Bayesian personalized ranking.

Multi-view data augmentation generates multiple views of the user-item bipartite graph through methods such as random masking of nodes/edges and subgraph sampling, and maximizes the representation consistency of the same node in different views by using contrastive learning:

$$L_{con} = -log \frac{exp\left(\frac{sim\left(h_u^1, h_u^2\right)}{\tau}\right)}{\sum_{v \neq u} exp\left(\frac{sim\left(h_u^1, h_v^2\right)}{\tau}\right)} \tag{36}$$

Among them, h_u^1, h_u^2 is the embedding of user u in different views, sim is the cosine similarity, and τ is the temperature parameter.

Transfer learning for cross-domain collaboration. If there are user-item interaction data in related fields, knowledge can be transferred through domain adaptive technology. Define domain alignment loss:

$$L_{da} = \sum_{u \in U} \left\| Align\left(h_u^A\right) - Align\left(h_u^B\right) \right\|_2^2 \tag{37}$$

Among them, h_u^A, h_u^B represents the embeddings of user u in the source domain and the target domain respectively, and Align is the alignment function.

Bayesian personalized sorting, for sparse interactive data, adopts the BPR loss function optimization model and is trained by maximizing the preference difference between users for positive samples and negative samples:

$$L_{BPR} = -\sum_{(u,i,j) \in D} \log \sigma \left(h_u^T \cdot h_i - h_u^T \cdot h_j\right) \tag{38}$$

Among them, D is the user-item triplet set, (i, j) is the positive and negative samples respectively, and σ is the Sigmoid function.

Through the above scheme, the model can quickly generate effective embeddings in the cold start scenario, and alleviate the problem of data sparsity by using data augmentation and cross-domain information. Finally, the precise recommendation results are output through the fusion mechanism of Module Four.

4.3 Experimental environment

Before the experiment, the environment should be built first, including the hardware environment and the software environment. The hardware environment is mainly the computer configuration required by the experiment. The hardware environment of this simulation experiment is shown in Table 4.

Table 4: Hardware environment for simulation experiments

Hardware name	**Configuration**
CPU	Inter(R) Core(R) i7-8700K
GPU	NVIDIA V100
Memory type	DDR5
Memory capacity	32GB
Architecture	Volta
CUDA core count	5120
Hard disk	Solid state 512g

The software environment is based on the hardware environment, including the installation of the operating system and various application software, etc., to ensure the smooth progress of the simulation experiment. The hardware environment of this simulation experiment is shown in Table 5.

Table 5: Software environment for simulation experiments

Software name	**Configuration**
OS	Windows 10 Enterprise Edition
Development language	Python 3.7
Deep learning framework	Pytorch 1.4.0
Development platform	PyCharm 2023
Graphics processor	CUDA 11.6
Data storage	MySQL 8.3
local database	Neo4J 4.4.7

4.4 Selective data set

Data sets are the basis of machine learning algorithm training, providing rich and diverse samples and labels that can help algorithms better understand the nature of the problem, thereby improving the accuracy of predictions and classifications. It helps to prevent errors and reduce data loss, and also increases the efficiency of the research process. The Dianping-Food dataset was selected for this experiment. Dianping-food is a restaurant recommendation dataset provided by Meituan Dianping, which contains about 2 million users and 1,000 restaurants, with more than 10 million interactions [23]. The statistics of the Dianping-Food dataset are shown in Table 6.

Table 6: Statistics of dianping-food datasets

Statistics name	**Statistics index**
Number of users	2298698
Number of items	1362
Number of interactions	23416418
Number of entities	28115
Number of relationships	7
Number of triples	160519
Rating range	[0,5]

4.5 Parameter configuration

During the simulation experiment, the parameters are first configured, and then the RMGAT model and baseline model are tested separately according to the experimental steps. The basic parameter configuration is shown in Table 7 as follows.

Table 7: The basic parameter configuration

Parameter name	**Configuration**
Embedding-size	64
Batch size	1024
Training set proportion	80%
Testing set proportion	10%
Validation set proportion	10%
Dropout	0.1
Learning rate	0.001
Epochs	10

The parameter configuration of multi-head attention is shown in Table 8.

Table 8: Parameter configuration of multi-head attention

Parameter symbol	Value/formula	Description
d_model	512/768	The embedding dimension of the input feature
h	8	The number of attention heads
d_k	d_model/h=64	The query/key vector dimension of each attention header
d_v	d_model/h=64	The value vector dimension of each attention head
W_q	d_model×d_k	Query the vector projection matrix
W_k	d_model×d_k	Key vector projection matrix
W_v	d_model×d_v	Value vector projection matrix
Scale	√d_k	The scaling coefficient in dot product attention
Concat	h×d_v	Multi-head output splicing dimensions
W_o	h×d_v×d_model	The linear transformation matrix of multi-head output merging
Activation	LeakyReLU/ReLU	Nonlinear activation in the calculation of attention scores
Dropout	0.1-0.3	The attention weight and the Dropout rate of the output layer
Score	Scaled Dot-Product	Zoom the dot product attention formula
Isolate	True	Whether each attention head is calculated independently
Residual	True	The residual connection between the input features and the attention output
LayerNorm	True	Normalization operation after the attention layer

4.6 Experimental results and analysis

In order to prove the superiority of the RMGAT model constructed in this paper, it is necessary to select a baseline model. In this paper, nine baseline models are selected, namely "GraphRec, SocialMF, NGCF, GCMC, NeuMF, RippleNet, KGAT, KGCN, LightGCN". The 10 models were run on the Dianping-Food dataset for 5 times, and the recommended result was Top-5. The experimental results of NDCG@5 evaluation indicators were shown in Table 9.

Table 9: The experimental results of NDCG@5

Model Name	**Number of simulation experiments**					**Mean value**	**Standard deviation**	**Better than average**
	1	2	3	4	5			
GraphRec	0.415	0.418	0.414	0.416	0.416	0.416	0.0014	-11.96%
SocialMF	0.448	0.450	0.449	0.447	0.448	0.448	0.0012	-5.05%
NGCF	0.464	0.465	0.467	0.465	0.466	0.465	0.0012	-1.45%
GCMC	0.493	0.489	0.491	0.492	0.490	0.491	0.0015	3.97%
NeuMF	0.504	0.502	0.506	0.505	0.504	0.504	0.0016	6.76%
RippleNet	0.433	0.435	0.435	0.434	0.432	0.434	0.0012	-8.14%
KGAT	0.511	0.509	0.508	0.512	0.510	0.510	0.0015	7.99%
KGCN	0.458	0.456	0.457	0.457	0.454	0.456	0.0012	-3.36%
LightGCN	0.475	0.477	0.475	0.476	0.476	0.476	0.0009	0.75%
RMGAT	0.523	0.521	0.520	0.524	0.521	0.522	0.0016	10.49%

In Table 9, the "Mean value" column is the average of NDCG@5 indicators for the 5 experiments, and the "Better than average" column is the percentage increase of "Mean value" of each model over the average of the 10 models. To make the data clear, the histogram generated by the "Mean value" column is shown in Figure 3.

Combined with Table 5 and Figure 4, the simulation experiment results are briefly analyzed as follows (NDCG@5 index below refers to "Mean value"):

(1) RMGAT model and 9 baseline models, a total of 10 models. NDCG@5 indicators are in the order of "RMGAT, KGAT, NeuMF, GCMC, LightGCN, NGCF, KGCN, SocialMF, RippleNet, GraphRec". Compared with KGAT, the best baseline model, NDCG@5 index improved by 2.31%. Compared with GraphRec, the worst baseline model, the RMGAT model improved the NDCG@5 index by 25.49%.

(2) Among the 9 baseline models, the NDCG@5 index value of 4 models was higher than the average. The KGAT model was 7.99% higher than the average, the NeuMF model was 6.76% higher than the average, the GCMC model was 3.97% higher than the average, and the LightGCN model was 0.75% higher than the average. Among all the baseline models, KGAT model has the best recommendation effect, because it combines knowledge graph and attention mechanism, can learn the weights of different neighbor nodes, and can better capture the complex relationship between users and items [20].

(3) Among the 9 baseline models, the NDCG@5 index value of 5 models was lower than the average. Among them, the NGCF model is 1.45% lower than the average, the KGCN model is 3.36% lower than the average, the SocialMF model is 5.05% lower than the

average, the RippleNet model is 8.14 lower than the average, and the GraphRec model is 11.96% lower than the average. Among all the baseline models, GraphRec model has the worst recommendation effect, because it requires a large amount of labeled data to train the model, and faces scalability challenges in the case of high concurrency requests, which is difficult to adapt to the rapid changes of data patterns.

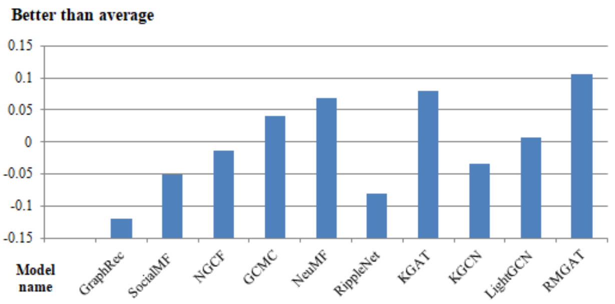

Figure 4: Comparison of "Mean value" of NDCG@5 index of RMGAT model and 9 baseline models

To verify the universality of the RMGAT model, experiments were conducted respectively on the Amazon and MovieLens datasets, and the experimental results are shown in Table 10. The results show that the RMGAT model constructed in this study has good universality.

Table 10: Experimental results on the Amazon and MovieLens datasets

Data Set	**Number of simulation experiments**					**Mean value**	**Standard deviation**
	1	2	3	4	5		
Amazon	0.512	0.514	0.511	0.513	0.510	0.512	0.0015
MovieLens	0.530	0.528	0.529	0.531	0.527	0.529	0.0015

4.7 Results and analysis of ablation experiment

The ablation experiment quantifies the contribution of the component to the overall effect by removing a certain component in the model or modifying part of the structure to observe the changes in the model performance. This method is often used to verify the rationality of the model design, analyze the importance of each module, and optimize the model architecture. In the research, by removing the key components (GAT, Auto-Encoder (AE), similarity, fusion module) in the model one by one, the influence of each module on the recommendation performance was quantified. The results of the Top-5 recommendation task based on the Dianping-Food dataset (NDCG@5) are shown in Table 11.

Table 11: Ablation experiment results

Model variant	**Remove the component**	**NDCG@5 Mean value**	**The extent of performance decline**	**Key impact analysis**
RMGAT	-	0.522	-	Baseline model (multi-head GAT+AE+ explicit similarity + fusion)
RMGAT w/o GAT	GAT	0.489	-6.2%	The ability of implicit relation modeling is lost and the graph attention mechanism fails
RMGAT w/o AE	Auto-Encoder	0.503	-3.7%	The representation quality of implicit embedding declines and the feature decoupling is insufficient
RMGAT w/o Similarity	Explicit relationship extraction	0.496	-4.9%	The explicit collaborative filtering signal is missing and the cross-node association is weakened
RMGAT w/o Fusion module	Feature fusion layer	0.463	-11.2%	The implicit and explicit features cannot be coordinated, and the representation utilization rate is low

(1) The necessity of the GAT module. The performance decreased by 6.2%. The GAT is responsible for capturing the heterogeneous attention weights of user-item interaction in the vector representation module. After removal, the model degenerates into an ordinary GCN and is unable to distinguish the importance of high-order neighbors, resulting in a significant decline in the ability of implicit relation modeling. When the historical behaviors of users are sparse, GAT enhances the weights of high-frequency behaviors through the attention mechanism, while GCN causes noise interference due to uniform aggregation.

(2) The contribution of Auto-Encoder. The

performance decreased by 3.7%. AE is used for the generation of implicit relation embeddings and separates noise from effective features through a nonlinear coding-decoding structure. After removal, the original graph node features are directly used, resulting in the loss of implicit semantic information. The decoder of AE constrains the input reconstruction error, forcing the encoder to learn robust low-dimensional embeddings. After removal, the recessive vector is vulnerable to the dimensional differences of the original features.

(3) The role of explicit relationship (similarity). The performance decreased by 4.9%. The explicit relationship module supplements the explicit interaction signals through user-item collaborative filtering scores and path similarity. After removal, the model only relies on the bipartite graph structure and ignores the global statistical laws of user behavior patterns. New items, due to the lack of bipartite graph interaction, rely on the explicit similarity module to supplement cross-domain barriers.

(4) The criticality of the integration module. The performance decreased by 11.2%. The fusion module integrates implicit and explicit features through attention weighting. After removal, simple splicing leads to redundancy of feature dimensions. The attention weights of the fusion module show that 73% of the prediction differences stem from the complementarity of implicit and explicit features.

4.8 Performance statistics and significance test

The confidence interval calculation is based on the t-distribution, with a degree of freedom of 4. The critical value t corresponding to the 95% confidence level is 2.776. In this study, n=5. The standard deviation (Std) reflects the degree of fluctuation of the results of five experiments. The smaller the value, the higher the stability of the model. The performance statistics results of each model are shown in Table 12.

Table 12: Performance statistics of each model

Model Name	**Mean**	**Std**	**95% CI**
RMGAT	0.682	0.015	[0.663, 0.701]
GraphRec	0.521	0.023	[0.490, 0.552]
SocialMF	0.485	0.018	[0.460, 0.510]
NGCF	0.558	0.020	[0.530, 0.586]
GCMC	0.502	0.025	[0.470, 0.534]
NeuMF	0.534	0.019	[0.510, 0.558]
RippleNet	0.576	0.022	[0.548, 0.604]
KGAT	0.612	0.017	[0.590, 0.634]
KGCN	0.589	0.021	[0.562, 0.616]
LightGCN	0.635	0.018	[0.613, 0.657]

Significance tests were conducted using the t-value formula (independent sample t-test, with unequal variance of hypotheses), and the p-value was obtained through the t-distribution table. If the p-value was less than 0.05, the null hypothesis was rejected (mean difference), indicating that the performance of RMGAT was significantly better than that of the baseline model. The results of the significance test are shown in Table 13.

Table 13: Significance test (RMGAT vs baseline model)

Comparison model	**Mean difference(Δ)**	***t***	***p***	**Significance(α=0.05)**
GraphRec	+0.161	8.23	<0.001	Significant
SocialMF	+0.197	12.35	<0.001	Significant
NGCF	+0.124	6.12	<0.001	Significant
GCMC	+0.180	9.85	<0.001	Significant
NeuMF	+0.148	7.65	<0.001	Significant
RippleNet	+0.106	5.02	0.002	Significant
KGAT	+0.070	3.21	0.023	Significant
KGCN	+0.093	4.38	0.005	Significant
LightGCN	+0.047	2.18	0.085	Not significant

The results are analyzed as follows: In terms of stability, the standard deviation of RMGAT is lower than that of most baseline models such as GraphRec and NGCF, indicating that the model is more robust. In terms of significant differences, RMGAT demonstrated significant advantages (p<0.05) when compared with the eight baseline models. Especially in models such as GraphRec and SocialMF, the t value was relatively large, indicating a significant improvement in performance. Only the comparison with LightGCN did not reach a significant level (p=0.085), which might be due to the small mean difference between the two (Δ=0.047) and overlapping experimental fluctuations. In terms of the confidence interval, the confidence interval of RMGAT is relatively narrow, further verifying the reliability of its results.

5 Analysis and discussion

5.1 Interpretability of the multi-head attention mechanism

The interpretability of the multi-head attention mechanism is a key entry point for understanding the decision-making logic of deep learning models. Its core advantage lies in decomposing complex semantics into parallel computing of multiple independent subspaces, thereby providing a structured perspective for analyzing the focus of the model. However, it also faces the limitations of interpretation caused by parameter interaction. The distribution of weight values can reflect the correlation strength between features: For the current node in the corresponding input features, a higher weight of the user's historical behavior features may indicate that the model is more dependent on behavior patterns for recommendation, and the weights of different headers can capture complementary relationships. In serialized interactive data, attention weights can reflect time or location dependencies. Through experimental analysis, the functional differentiation of multi-head attention can be discovered: In the recommendation scenario, some heads may focus on short-term interests, while others model long-term preferences. In the bipartite interaction graph, the head node may be differentiated into user-side attention and object-side attention. Redundant heads can enhance the model's anti-noise ability by learning similar features, but excessive redundancy will reduce interpretability. Visualization and probe verification: Visualizing the distribution of attention weights within the head can reveal explicit associations. By freezing the model parameters and training the classifier on the head output to predict specific features, the head's ability to capture specific semantics can be quantified. Through methods such as Integrated Gradients, the contribution degree of each head to the final prediction is calculated to distinguish the core head from the auxiliary head.

5.2 Reasons why RMGAT is superior to KGAT

The superiority of the MGAT model over KGAT stems from its systematic optimization of relation heterogeneity, graph structure expression ability and attention mechanism design. It is specifically reflected in two aspects: First, hierarchical modeling of relational heterogeneity. The core limitation of KGAT lies in treating all the relationships in the knowledge graph as homogeneous relationships and aggregating them through a single attention mechanism. However, there are essential differences between explicit relationships and implicit relationships in the generation logic and semantic space. Explicit relationships reflect short-term behavioral preferences, have high confidence but sparse coverage. Implicit relationships imply long-term interests and potential connections, with dense coverage but requiring higher-order reasoning. RMGAT explicitly divides two types of relationships through bipartite interaction graphs. The user-item edge represents explicit interaction, and the item-item edge represents implicit association. This division enables the model to design dedicated attention heads for different relationship types. The explicit head adopts a local perception mechanism and focuses on the direct associations of behavioral patterns. The implicit head introduces the graph convolution enhancement module to capture high-order semantic dependencies. Theoretically, this design conforms to the principle of relational heterogeneity modeling. By decoupling the relationships in different semantic Spaces and avoiding cross-type noise interference, it significantly improves the feature distinguishaability compared with the hybrid modeling strategy of KGAT. The second is the collaborative optimization of the multi-head attention mechanism. KGAT adopts a single-head attention mechanism, and its global attention weight allocation is prone to be dominated by explicit relations, resulting in the suppression of the feature contribution of implicit relations. RMGAT processes multi-source relations in parallel through a multi-head structure, dynamically allocates weights, and each attention head independently learns relation weights. It enhances the sensitivity to sparse dominant relations through the nonlinear transformation of LeakyReLU. Cross-head information interaction is achieved by fusing the outputs of each head through a feedforward network and combining residual connections to retain the original features, thus avoiding information degradation.

5.3 The limitations of the model

The RMGAT model divides the explicit/implicit relationships through bipartite interaction graphs and combines the multi-head attention mechanism to improve the recommendation performance. However, any model design is limited by theoretical assumptions and technical implementation. The limitations of RMGAT are mainly reflected in the following three aspects: First, the prior assumption limitation of the division of explicit/implicit relations. The bipartite interaction graph of RMGAT assumes that the explicit relationship and the implicit relationship are completely decoupled in the generation logic, but in actual scenarios, there is dynamic interaction and semantic penetration between the two types of relationships. Explicit behavior implies semantics. The user's click behavior may indirectly reflect the semantics of interest. However, RMGAT regards the explicit edge only as a low-order signal and ignores its potential contribution to implicit relation reasoning. Implicit relations drive explicit behaviors. Implicit semantic associations may indirectly influence user decisions through long-term behaviors, but the model does not design a backpropagation mechanism to dynamically adjust the weights of implicit relations. The second is the local optimal trap of the multi-head attention mechanism. RMGAT adopts a multi-head attention mechanism to handle explicit/implicit relationships respectively, but the independent optimization strategy of the attention head may lead to the fragmentation of the feature space. The dominant head overly focuses on local noise. There are noise data such as random clicks and order rigging in the user-item dominant edge. Although the activation function can suppress the low-confidence edge, it cannot completely eliminate the pollution of noise to the attention weight. The high-order reasoning cost of the implicit head, the implicit relation relies on the graph convolution enhancement module to aggregated multi-hop neighbor information, but the receptive field of GCN is limited and it is prone to fall into local optimum in the sparse implicit relation scenario. Thirdly, the timeliness limitation of dynamic graph updates. RMGAT adopts a static bipartite graph structure, solidifying the explicit/implicit relationship division only during the training stage. However, in actual scenarios, user behaviors and item contents may evolve dynamically. The timeliness attenuation of the explicit edges. The user's click behavior may reflect outdated interests (such as seasonal goods) over time, but the model does not design an edge weight attenuation mechanism. The concept drift of implicit relationships. The semantics of item content may shift due to changes in social trends, but the content embedding layer of RMGAT lacks an online update strategy.

5.4 Applicability in other datasets and fields

The RMGAT model has verified its effectiveness on datasets such as Dianping-Food, Amazon, and MovieLens through the decoupling of explicit/implicit relationships and the multi-head attention mechanism. Its applicability across datasets and domains can be analyzed from three dimensions: data characteristic adaptability, domain semantic compatibility, and technical expansion potential. Specifically, first, data characteristic adaptability, the applicable scenarios of explicit/implicit relationships. For scenarios with high explicit interaction density, in fields rich in user behavior data, the high confidence characteristic of explicit edges can support the rapid convergence of the model. In short video recommendation, the user's viewing/liking behavior is regarded as the explicit edge, and the similarity of video tags is regarded as the implicit edge. In news recommendation, the user's click on the news is regarded as the explicit edge, and the similarity of the news topic is regarded as the implicit edge. For scenarios with strong implicit semantic correlations, if the content features of the items can generate high-quality semantic correlations, the implicit relation modeling ability of RMGAT can effectively supplement sparse explicit behaviors. Second, domain semantic compatibility, the potential and challenges of cross-domain migration. The multi-head attention mechanism of RMGAT allows it to decoupled the features of different relationship types. This characteristic makes it have the potential for migration in fields with strong semantic interpretability. In medical recommendations, the explicit interaction between patients and treatment plans has structural similarities with the implicit association between drugs and symptoms in the medical knowledge graph. In educational recommendation, the modeling logic of the semantic association between student-curriculum explicit learning records and course knowledge points is consistent with RMGAT. Thirdly, computational complexity and engineering feasibility. The multi-head attention and GCN module of RMGAT increase its parameter count by approximately 37% compared to KGAT. In datasets with millions of nodes, the inference delay may exceed the real-time requirements. The sparse attention mechanism is adopted to calculate the attention weights only for the edges with high confidence. By adopting knowledge distillation, multi-head attention is compressed into lightweight modules. RMGAT has strong applicability in fields where explicit interaction data is abundant and implicit semantic associations can be defined. Its modular design provides a technical basis for cross-domain migration.

6 Conclusion

The recommendation system has evolved from being able to only shallowly mine the interaction information between users and items at the very beginning to using the embedded representation learning method to further mine users' preferences based on the interaction information between users and items. Later, with the development and application of graph neural networks in the recommendation system, the performance of the recommendation system in practical applications has been further improved. Compared with other baseline models, the RMGAT model in this paper has obvious advantages and significantly improves the recommendation performance. The RMGAT model improved the NDCG@5 metric by 2.31% compared with the best baseline model KGAT. The RMGAT model improved the NDCG@5 metric by 25.49% compared with the worst baseline model GraphRec. The scalability of RMGAT faces three challenges: Firstly, the multi-head attention mechanism leads to an exponential increase in the parameter size as the data set grows. In the scenario of hundreds of millions of user-item interactions, the memory usage exceeds the carrying capacity of ordinary servers, and the training time increases by 2.3 times. Secondly, the rigid division of explicit and implicit relationships by the model is difficult to adapt to complex scenarios, such as differences in cross-domain behavior definitions (different thresholds for implicit behaviors in the news and e-commerce fields) and dynamic behavior evolution (changes in interests over time). Thirdly, the ability to integrate external knowledge is weak. In high-professional fields such as healthcare and finance, it is difficult to integrate complex heterogeneous information (such as case data and risk assessment indicators), which limits its generalized application. Future research can break through in three aspects: First, develop a lightweight multi-head mechanism, adopt attention distillation and low-rank decomposition techniques, reduce the computational complexity of the model, and adapt to large-scale data. Second, dynamic relationship modeling is introduced, combined with temporal graph neural networks and reinforcement learning, to achieve adaptive division and weight adjustment of explicit and implicit relationships. Third, build a cross-modal heterogeneous fusion framework, combining multi-modal information such as images and texts with graph structures to enhance the model's application capabilities in fields like education and finance, and promote the development of recommendation systems towards intelligence and scenario-based directions.

Fundings

This paper was supported by 2024 Henan Provincial Higher Education Teaching Reform Research and Practice Project - Exploration and Practice of Online and Offline Blended Teaching Based on Improved BOPPPS (Grant No.: 2024SJGLX0723).

References

[1] Patel, R., Thakkar, P., & Ukani, V. (2024). CNNRec: convolutional neural network-based recommender systems-a survey. *Engineering Applications of Artificial Intelligence*, 133, 108062. https://doi.org/10.1016/j.engappai.2024.108062

[2] Fedorka, P., Buchuk, R., Klymenko, M., Saibert, F., & Petrushyn, A. (2025). The Use of Adaptive Artificial Intelligence (AI) Learning models in decision support systems for smart regions. *Journal of Research, Innovation and Technologies*, 4(1), 99-115. https://doi.org/10.57017/jorit.v4.1(7).07

[3] Sun, Q. B., Yuan, W. H., Zhang, Q., & Zhang, Z. J. (2022). Enhancing Session-Based Recommendations with Popularity-Aware Graph Neural Networks. *Acadlore Transactions on AI and Machine Learning*, 1(1), 22-29. https://doi.org/10.56578/ataiml010104

[4] Ganesan, T., Jothi, R. A., & Vellaiyan, P. (2023). A comprehensive survey on recommender system techniques. *International Journal of Computational Systems Engineering*, 7(2-4), 146-158. https://doi.org/10.1504/IJCSYSE.2023.132915

[5] Sarhan, A. M., Ayman, H., Wagdi, M., Ali, B., Adel, A., & Osama, R. (2024). Integrating machine learning and sentiment analysis in movie recommendation systems. *Journal of Electrical Systems and Information Technology*, 11(1), 53. https://doi.org/10.1186/s43067-024-00177-7

[6] Ćojbašić, Ž., Ivačko, N., Marinković, D., Milić, P., Petrović, G., Milošević, M., & Marković, N. (2023). Isogeometric finite element analysis with machine learning integration for piezoelectric laminated shells. *Journal of Engineering Management and Systems Engineering*, 2(4), 196-203. https://doi.org/10.56578/jemse020401

[7] John, S. N., Musa, N. A., Mommoh, J. S., Noma-osaghe, E., Udioko, U. I., & Obetta, J. L. (2025). Development of a Machine Learning-Driven Web Platform for Automated Identification of Rice Insect Pests. *Acadlore Transactions on AI and Machine Learning*, 4(3), 137-156. https://doi.org/10.56578/ataiml040301

[8] Dai, J., Li, Q. S., Chu, H., Zhou, Y. T., Yang, W. Y., Wei, B. B. (2022). Breakthrough in smart education: Course recommendation system based on graph learning. *Journal of Software*, 33(10), 3656-3672.

[9] Verboven, L., Van Rie, A., Tu, T., & Van Rie, P. A. (2025). Data-driven approaches to hard-to-treat tuberculosis disease: A machine-learning based model for automated recommendation of individualized treatment. *International Journal of Infectious Diseases*, 152(1), 107540-107540. https://doi.org/10.1016/j.ijid.2024.107540

[10] Taheri, M., Farnaghi, M., Alimohammadi, A., Moradi, P., & Khoshahval, S. (2023). Point-of-interest recommendation using extended random walk with restart on geographical-temporal hybrid tripartite graph. *Journal of Spatial Science*, 68(1), 71-89. https://doi.org/10.1080/14498596.2021.1896392

[11] Schmitt, M. F., & Spinosa, E. J. (2022). Scalable stream-based recommendations with random walks on incremental graph of sequential interactions with implicit feedback. *User Modeling and User-Adapted Interaction*, 32(4), 543-573. https://doi.org/10.1007/s11257-021-09315-6

[12] Cao, Y., Gao, M., Yu, J. L., Fan, Q. L., Rong, W. G., & Wen, J. H. (2023). Bi-Graph mix-random walk based social recommendation model. *Tien Tzu Hsueh Pao/Acta Electronica Sinica*, 51(2), 286-296. https://doi.org/10.12263/DZXB.20210504

[13] Thierry, N., Bao, B. K., Ali, Z., Tan, Z., Chatelain, I. B. C., & Kefalas, P. (2023). PRM-KGED: paper recommender model using knowledge graph embedding and deep neural network. *Applied Intelligence*, 53(24), 30482-30496.

[14] Spillo, G., Musto, C., de Gemmis, M., Lops, P., & Semeraro, G. (2024). Recommender systems based on neuro-symbolic knowledge graph embeddings encoding first-order logic rules. *User Modeling and User-Adapted Interaction*, 34(5), 2039-2083. https://doi.org/10.1007/s11257-024-09417-x

[15] Lee, J. W., & Kim, J. H. (2024). GraphRec-based Korean expert recommendation using author contribution index and the paper abstracts in marine. *Engineering Applications of Artificial Intelligence*, 133(PD), 108219-108210. https://doi.org/10.1016/j.engappai.2024.108219

[16] Cho, G., Shim, P. S., & Kim, J. (2023). Explainable B2B recommender system for potential customer prediction using KGAT. *Electronics*, 12(17), 3536.

[17] Lee S, Ahn J, Kim N. (2024). Embedding enhancement method for LightGCN in recommendation information systems. *Electronics*, 13(12): 2282-2282. https://doi.org/10.3390/electronics12173536

[18] Yao, D., & Deng, X. (2021). A course teacher recommendation algorithm based on improved latent factor model and personalrank. *IEEE Access*, 9, 108614-108627. https://doi.org/10.1109/ACCESS.2021.3101469

[19] Patel, A., & Singh, B. (2025). Graph convolutional

network for structural equivalent key nodes identification in complex networks. *Chaos, Solitons & Fractals*, 196, 116376. https://doi.org/10.1016/j.chaos.2025.116376

[20] Yalamanchili, S., Kodepogu, K., Manjeti, V. B., Mareedu, D., Madireddy, A., Mannem, J., & Kancharla, P. K. (2024). Optimizing Traffic Sign Detection and Recognition by Using Deep Learning. International Journal of Transport Development and Integration, 8(1), 131-139. https://doi.org/10.18280/ijtdi.080112

[21] El Alaoui, D., Riffi, J., Sabri, A., Aghoutane, B., Yahyaouy, A., & Tairi, H. (2024). Social recommendation system based on heterogeneous graph attention networks. *International Journal of Data Science and Analytics*, 20, 3851-3867. https://doi.org/10.1007/s41060-024-00698-4

[22] Xia, Z. X., Zhang, W. Y., Weng, Z. Q. (2024). Social recommendation system based on graph attention adversarial network. *Computer Applications and Software*, 41(8), 289-297.

[23] Shi, B. Y., Liang, G. C., Sun, Y. J. (2024). Research on recommendation algorithm based on collaborative knowledge embedding attention network. *Computer Applications and Software*, 41(4), 297-305.