Frontiers in Neurorobotics

TYPE Original Research
PUBLISHED 18 June 2025
DOI 10.3389/fnbot.2025.1587973

OPEN ACCESS

EDITED BY
Xianmin Wang,
Guangzhou University, China

REVIEWED BY
Likang Wu,
Tianjin University, China
Haimonti Dutta,
University at Buffalo, United States
Lars Wagner,
Technical University of Munich, Germany

*CORRESPONDENCE
Xiaofei Han
✉ fendyhanxiaofei@shisu.edu.cn

RECEIVED 05 March 2025
ACCEPTED 12 May 2025
PUBLISHED 18 June 2025

CITATION
Han X and Dou X (2025) User recommendation method integrating hierarchical graph attention network with multimodal knowledge graph.
Front. Neurorobot. 19:1587973.
doi: 10.3389/fnbot.2025.1587973

User recommendation method integrating hierarchical graph attention network with multimodal knowledge graph

Xiaofei Han[1,2]* and Xin Dou[2]

[1]Business College, California State University, Long Beach, CA, United States, [2]School of Business and Management, Shanghai International Studies University, Shanghai, China

In common graph neural network (GNN), although incorporating social network information effectively utilizes interactions between users, it often overlooks the deeper semantic relationships between items and fails to integrate visual and textual feature information. This limitation can restrict the diversity and accuracy of recommendation results. To address this, the present study combines knowledge graph, GNN, and multimodal information to enhance feature representations of both users and items. The inclusion of knowledge graph not only provides a better understanding of the underlying logic behind user interests and preferences but also aids in addressing the cold-start problem for new users and items. Moreover, in improving recommendation accuracy, visual and textual features of items are incorporated as supplementary information. Therefore, a user recommendation model is proposed that integrates hierarchical graph attention network with multimodal knowledge graph. The model consists of four key components: a collaborative knowledge graph neural layer, an image feature extraction layer, a text feature extraction layer, and a prediction layer. The first three layers extract user and item features, and the recommendation is completed in the prediction layer. Experimental results based on two public datasets demonstrate that the proposed model significantly outperforms existing recommendation methods in terms of recommendation performance.

KEYWORDS

user recommendation, hierarchical graph attention network, knowledge graph, multimodal, visual features, textual features

1 Introduction

In the era of rapid information expansion, users are increasingly overwhelmed by vast amounts of digital content, leading to issues such as information overload and difficulty in efficiently locating relevant content. This challenge is particularly pronounced on emerging short video, social media, and e-commerce platforms, where the volume and diversity of data far exceed users' cognitive and processing capacities. Consequently, intelligent recommendation systems have become essential tools for filtering massive datasets and delivering personalized content to users ((Liu et al., 2024; Wu et al., 2024; Yamada, 2021) and limited representation capacity, all of which hinder their performance in dynamic and complex environments (Zhang et al., 2019; Gao et al., 2022). With the rise of deep learning, the landscape of recommendation systems has evolved significantly. Techniques such as convolutional neural network (CNN) and recurrent neural networks (RNN) have shown strong capabilities in feature extraction and modeling temporal user preferences (Han et al., 2024; Wang et al., 2024, 2025, Yan et al., 2025).

Traditional recommendation approaches—such as content-based filtering (CBF), collaborative filtering (CF), and hybrid recommendation (HR)—primarily rely on users' historical interactions (e.g., purchases or browsing records) and similarity computations to provide recommendations (Javed et al., 2021; Wu et al., 2023). While these methods have achieved moderate success, they struggle with key issues such as the cold start problem, data sparsity, and limited representation capacity, all of which hinder their performance in dynamic and complex environments (Zhang et al., 2019).

With the rise of deep learning, the landscape of recommendation systems has evolved significantly. Techniques such as convolutional neural network (CNN) and recurrent neural networks (RNN) have shown strong capabilities in feature extraction and modeling temporal user preferences (Steck et al., 2021; Bandyopadhyay et al., 2024). For example, Visa et al. proposed a CNN-based feature extraction method, which uses matrix multiplication between users and items to uncover latent relationships, effectively solving the sparsity issue of similarity matrices and optimizing recommendation results (Visa and Patel, 2021). Cho et al. designed an RNN-based recommendation system that analyzes temporal data to precisely capture dynamically changing user needs, providing more accurate recommendations (Cho et al., 2014). More recently, graph neural networks (GNNs) have demonstrated remarkable effectiveness in capturing intricate relationships between users and items by modeling them as graph structures (Jiang et al., 2023). For instance, the GraphRec model proposed by Fan et al. constructs a user information model that combines social network data with user characteristics and uses a multilayer perceptron to extract features of target items, improving recommendation accuracy (Fan et al., 2019). Chen et al. improved the recommendation system's efficiency by leveraging social neighbor network information and using heterogeneous GNN methods (Chen and Wong, 2021).

Parallel to these advances, knowledge graphs (KG) have emerged as a promising auxiliary resource for enhancing recommendation performance. By formally encoding entities and their semantic relationships, KGs enable richer user-item interaction modeling. Existing KG-enhanced recommendation strategies include embedding-based, path-based, and joint learning approaches. These techniques have demonstrated improved interpretability and accuracy by incorporating external structured knowledge into recommendation pipelines (Guo et al., 2020; Zhang et al., 2024). Oramas et al. (2016) proposed a recommendation system that combines semantic and collaborative characteristics by extracting information from KG using both entity-and path-based strategies, transforming it into linear features. Path-based recommendations involve establishing user-item relation graph to uncover connections between entities, measure node similarity, and recommend content. Path analysis in KG reveals complex relationships between entities, allowing for precise exploration of user preferences through specific meta-paths, enhancing the interpretability of the recommendation system (Sun et al., 2020). Ma et al. (2019) developed the RuleRec algorithm, which extracts rules from an item-centric KG to identify various associations and provides recommendations using these inferred rules. Joint recommendation methods integrate path analysis and knowledge graph embedding approaches, where user interests are first captured via a propagation mechanism across the entire knowledge graph and then extracted through graph embedding techniques, ultimately completing the recommendation process (Yang et al., 2022). Wang X. et al. (2019) proposed the model, which combines user-item bipartite graph with knowledge graph and performs iterative diffusion in shared knowledge graph via GNN, enriching entity embeddings. Shi et al. (2021) introduced the method, modeling a heterogeneous information network, extracting multi-dimensional similarity matrices using different meta-paths, and integrating this information through deep learning network to complete the recommendation process.

Furthermore, multimodal recommendation systems—which integrate visual, textual, and sometimes audio data—have received growing attention for their ability to address the limitations of single-modal systems and further refine personalization (Li et al., 2019). By leveraging the complementary nature of different data types, multimodal methods can offer more comprehensive user representations and deeper insights into user preferences.

Motivated by the limitations of traditional and single-modal recommendation methods, this study proposes a novel recommendation framework that integrates hierarchical graph attention networks with a multimodal knowledge graph (HGAN-MKG). The model consists of four major components: a collaborative knowledge graph neural layer, an image feature extraction layer, a text feature extraction layer, and a prediction layer. Specifically, the collaborative KG layer captures deep user-item interactions via attention mechanisms and gated recurrent unit (GRU); the image layer applies a multi-path attention structure to analyze visual user behavior; the text layer uses multi-head self-attention and CNN to extract contextual features; and the prediction layer fuses all modalities to generate accurate recommendations. Experiments conducted on two benchmark datasets confirm the effectiveness of the proposed model, which outperforms several state-of-the-art baselines.

The remainder of this study is organized as follows: Section 2 introduces the underlying theory, Section 3 describes the design details of the algorithm, Section 4 presents experimental validation, and Section 5 concludes the research work.

2 Theoretical foundations

2.1 K-means clustering algorithm

The K-Means algorithm is an effective data clustering method that partitions a dataset into K clusters, with each cluster associated with its nearest center (Bock, 2007). This iterative algorithm continuously updates cluster centers and reassigns points to clusters until a predefined stopping criterion is met. The algorithm's principle is determined by solving an optimization function, with the sum of squared errors serving as the evaluation metric. The objective is to minimize the total cost function, as defined in Equation 1.

$$T = \sum_{m=1}^{K}\sum_{n=1}^{K}\left(x_n - \partial_m\right) \tag{1}$$

where ∂_m represents the centroid of cluster m, x_n is an element from the sample set, and K denotes the number of clusters.

In the context of recommendation systems, K-Means is applied to group users based on behavior, converting user activity data into vector form, and clustering users into K groups. This method identifies similarities between users, enabling personalized recommendations for items that users within the same group are likely to find interesting.

2.2 Attention mechanism technologies

2.2.1 Attention mechanism

The attention mechanism allocates different weights to various inputs based on their importance, prioritizing more relevant information while ignoring less significant details. As depicted in Figure 1, each input's Key, Value, and Query are processed as vectors, and the weight for each Key is determined by calculating its similarity to the Query. This results in a weighted sum of all Values, generating the final output. The calculation of similarity can be approached through four distinct strategies which is presented in Equation 2.

$$\begin{gathered} F(Q,K_i) = Q^T W K_i \\ F(Q,K_i) = \tanh\left(W\left[Q^T;K_i\right]\right) \\ F(Q,K_i) = V^T \tanh\left(WQ + UK_i\right) \\ F(Q,K_i) = Q^T K_i \end{gathered} \tag{2}$$

2.2.2 Graph attention mechanism

With the expansion of research in this field, GAN and their variants have emerged, becoming integral to GNN These mechanisms typically form the core of complex deep learning architectures involving multiple convolutional or pooling layers designed to handle graph-structured data. Each layer participates in the propagation and aggregation of features between nodes and their neighbors, updating node representations and performing classification tasks. The graph attention mechanism emphasizes the dynamic evaluation of relative importance between nodes, calculating weights accordingly. As a critical component of GNN, it plays a pivotal role in computing node weights and handling diverse graph structures and tasks.

This mechanism assigns weights based on input features, allowing for weighted aggregation of data for more precise and effective representation. By assessing similarities between nodes and their neighbors, attention coefficients are computed to allocate weights. This strategy enhances the significance of key nodes while mitigating noise interference. Typically, a trainable network model is used to determine attention coefficients, considering the unique features of nodes and their relative positions with neighboring nodes, ultimately generating attention weights. Once these weights are obtained, node feature vectors are combined with their corresponding weights, resulting in a weighted feature vector that represents either a node or the entire graph.

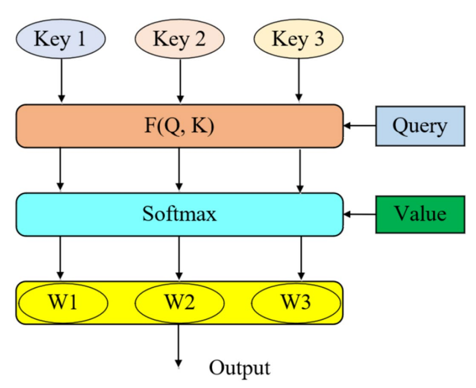

FIGURE 1
Allocation of attention mechanism weight.

2.2.3 Multi-head self-attention mechanism

The multi-head self-attention mechanism, a critical component of the Transformer model, has been widely adopted across various domains. Within the Transformer architecture, the attention mechanism consists of two key parts: scaled dot-product attention and multi-head attention, which together form the foundation of the model. The computation of scaled dot-product attention is as follows, as defined in Equation 3:

$$Attention(Q,K,V) = \text{Softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{3}$$

where Q represents the query vector, K and V are the key-value pairs, and d_k is the scaling factor.

Both single-head and multi-head self-attention are considered derivative forms of the attention mechanism, with multi-head attention enhancing the model's ability to manage long sequences and their complexity. Given an input sequence $[w_1, w_2, w_3, \ldots, w_T]$, where each w_i represents the vector form of the i_{th} word in the sequence.

The final representation $b=[b_1, b_2, b_3, \ldots, b_r]$ is derived from the weighted aggregation of attention heads. In a single-head attention mechanism, q_1, k_1, and v_1 constitute one "head." The multi-head self-attention mechanism multiplies specific w_1 values with multiple W^Q, W^K, and W^V matrices to generate multiple sets of q_1, k_1, and v_1, which is presented in Figure 2.

After obtaining the outputs from all heads, these feature vectors are concatenated and linearly transformed to generate the final representation as defined in Equation 4:

$$\begin{gathered} Multihead(Q,K,V) = Concat(head_1,\cdots,head_x)W^O \\ head_i = Attention\left(QW_i^Q, KW_i^K, VW_i^V\right) \end{gathered} \tag{4}$$

2.3 Knowledge graph

A knowledge graph is essentially a semantic structure represented in the form of a graph, containing multiple categories of entities and offering a more intuitive and transparent visualization of complex relationships. It effectively encodes semantic information between entities and provides a highly structured means of representation. Mapping the entities and their relations into a low-dimensional continuous vector space is a critical step toward knowledge modeling and enhancing recommendation systems. During this process, it is essential to preserve both the structural properties of the graph and the semantic consistency of the nodes, in order to minimize information loss or distortion.

Various techniques have been proposed for feature extraction in knowledge base construction, including distance-based embedding models, similarity-based conceptual models, and path-based relational learning methods. Among them, translational models have gained widespread adoption due to their simplicity and scalability. Representative distance-based translational models include TransE, TransR, and TransD. For instance, TransR models entities as collections of multi-attribute information and achieves triple-level embedding by projecting entities into relation-specific spaces.

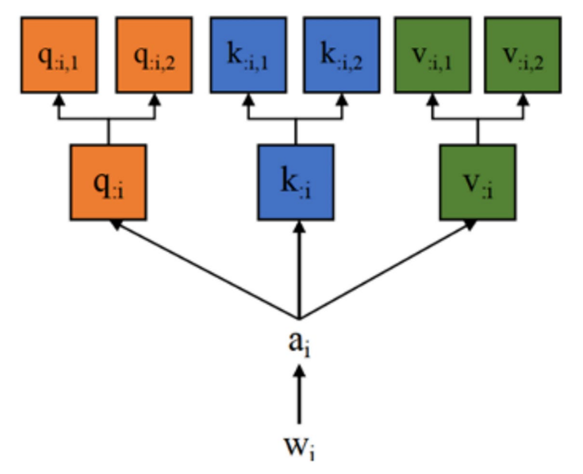

FIGURE 2
Multi-head self-attention.

In TransR, each relation r is associated with a distinct relation space, and a projection matrix M_r is defined to map entity vectors from the entity space to the corresponding relation space. Given a head entity h, a tail entity t, and a relation r, their representations in the relation space are represented in Equation 5:

$$\begin{aligned} h_\perp &= M_r h \\ t_\perp &= M_r t \end{aligned} \tag{5}$$

where $h,t \in \mathbb{R}^k$ represents the vector representation of the entity in the original entity space, $M_r \in \mathbb{R}^{d\times k}$ is the projection matrix corresponding to the relation r, and $h_\perp, t_\perp \in \mathbb{R}^d$ means the embedding representation of the head entity and the tail entity in the relation space. Through the above mapping, TransR can better model the differential impact of different relations on entity semantics.

Figure 3 illustrates the core concept of TransR. For each triple (h+r), the entities h and t are mapped to the relationship space r through a projection matrix W_r, resulting in representations h$_r$ and t_r. The goal is to ensure that (h$_r$+r) closely approximates t_r. The transformation of entities in the space is represented by the following Equation 6:

$$h_r = hW_r, t_r = tW_r \tag{6}$$

The score function for determining the proximity between head and tail entities is given by Equation 7:

$$g\left(h,r,t\right) = h_r + r - t_{r2}^{\,2} \tag{7}$$

According to Equation 7, the lower the score of the triple, the closer the head and tail entities are in the relationship space r, which increases the probability of the triple being correct.

3 Methodology

The proposed recommended model HGAN-MKG, first extracts knowledge graph information. It then utilizes the VGG19 network to extract image features of items and combines multi-head self-attention mechanisms with CNN to extract text features. Finally, the model fuses these three types of information for recommendation purposes. The proposed model consists of four key components: the collaborative knowledge graph neural layer, the image feature extraction layer, the text feature extraction layer, and the prediction network layer. The model architecture is illustrated in Figure 4.

3.1 Collaborative knowledge graph neural layer

In the collaborative knowledge graph neural layer, the bipartite graph between users and items in the knowledge graph is integrated to connect item features, thereby forming a knowledge graph. Let V={v_1, v_2, …, v_m} represent the set of items and U={u_1, u_2, …, u_n} represent the users. The set E={e_1, e_2, …, e_o} corresponds to the set of entities, where o, n, and m denote the total number of entities, users, and items, respectively. The matrix Y_{ij} indicates the interactions between users and items, defined as Y_{ij} =1 if an interaction exists, and Y_{ij}=0 otherwise. A knowledge graph G={(h, r, t)|h, r=ε, r∈R} is defined, where each triple consists of a head entity h, a relationship r, and a tail entity t. If there is an association between h and t, the elements in GGG represent entities and their associations.

In practice, the representation of entities and their relationships within a KG is typically approached using translation-based learning methods. These methods perform logical reasoning and mapping of entities and their relationships in a low-dimensional space for knowledge representation. In the TransE model, the triple (h, r, t) is expressed in vectorized form, where the relationship r is understood as a translation from h to t. By fine-tuning the vector representations of the triples, the equation h+r ≈t is satisfied. As technology has advanced, derivative techniques have emerged, such as the TransR model, where each entity is viewed as having multiple facets. Different relationships correspond to different aspects of an entity, and each semantic space corresponds to a specific relationship.

Through embedding techniques, item vector representations are obtained, and the attention mechanism is applied to explore the interactions between entities within the knowledge graph. This method not only integrates user data but also incorporates the internal relationships and complex hierarchical structures between entities in the knowledge graph. The training process based on the attention mechanism involves random sampling of nodes and the calculation of weight scores to capture direct relationships between entities. Subsequently, the attention mechanism analyzes these weights to identify and understand first-order entity relationships, which can be further extended to L-order entity relationships.

This paper delves into the training methods for the initial stages and extends the analysis to multi-layer L-structures. In this network layer, Equation 8 reveals the interactions between two entities, while Equation 9 measures the degree to which a user prefers specific relationships and entity information. For a specific item v_0, its neighboring nodes in the set $v^u_{U_{v_0}}$ are described by the formula, where (r, i, j) defines the connections between entities, and the attention mechanism is used to compute and evaluate the weights.

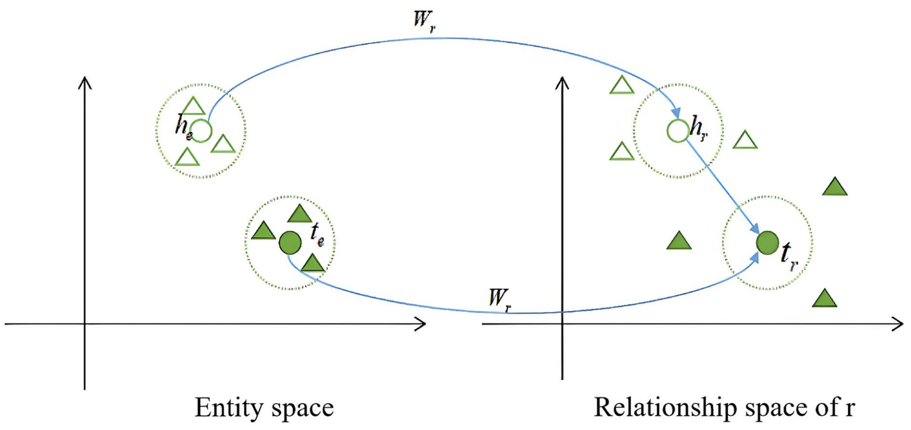

FIGURE 3
Graphical description of TransR.

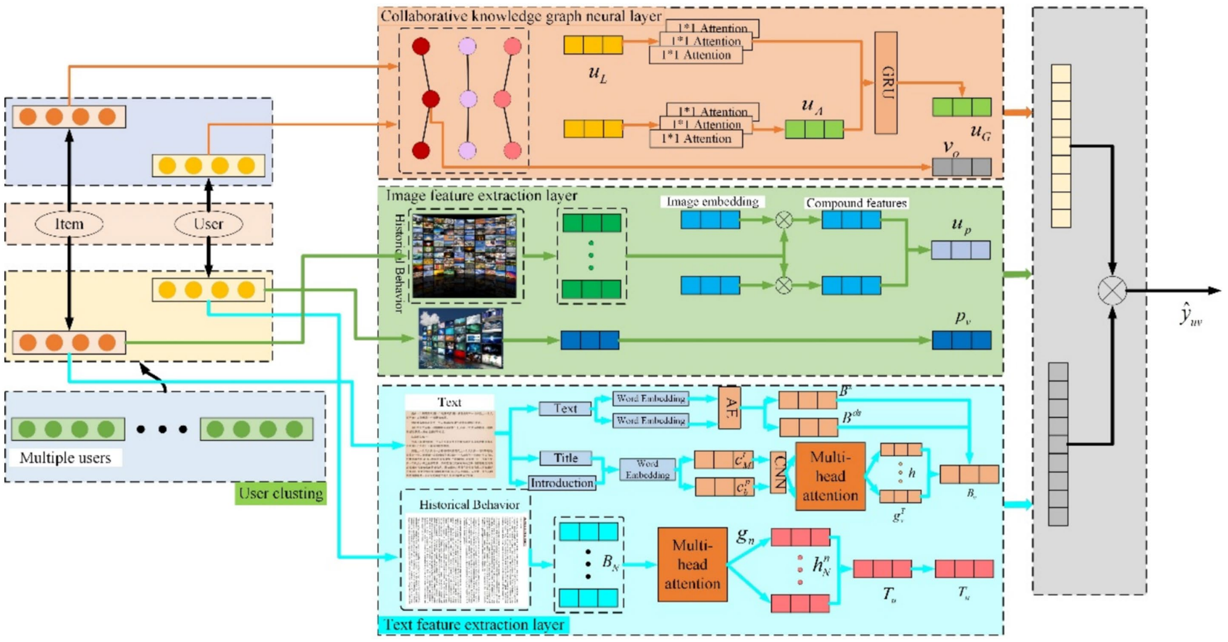

FIGURE 4
User recommendation method integrating hierarchical graph attention network with multimodal knowledge graph.

$$e_j r_{i,j} = \sigma\left(W_1 g concat\left(r_{i,j}, e_j\right) + b_1\right) \tag{8}$$

$$\pi^u_{r_i, e_j} = u \cdot r_{i,j} e_j \tag{9}$$

where $u \in R$, $r_{i,j} \in R$, j∈R, and $e_j \in R$ represent the user, entity connections, and tail entity vector representations, respectively, with www denoting the dimensionality. The structure uses a non-linear activation function σ and adjusts the weights W_1 and bias $b_1 \in R$ to assess the strength of user interest in different relationships and entity data. To construct the vectorized representation of neighboring entities for item υ_0, the vectors of adjacent entities e_0 are linearly fused as shown in Equation 10, followed by subsequent normalization.

$$v^u_{U_{v_0}} = \sum_{e^0 \in U_{v_0}} \bar{\pi}^u_{r_{i,j}, e_j} e^0$$
$$\bar{\pi}^u_{r_{i,j}, e_j} = \frac{\exp\left(\pi^u_{r_{i,j}, e_j}\right)}{\sum_{e^0 \in U_{v_0}} \bar{\pi}^u_{r_{i,j}, e_j} e^0} \tag{10}$$

To explore deeper entity information, this layer is extended from a single level to multiple layers, resulting in L-order vectorized descriptions of entities. This L -order vector representation aggregates data from neighboring entities up to $(L-1)$-order. The final vector representation of the item υ_L is obtained, where W_L represents the weight parameters and b_L denotes the bias. This is calculated precisely using Equation 11.

$$v_L = \sigma\left(W_L \cdot \left(v_{L-1} + v_{U_v L}\right) + b_L\right) \tag{11}$$

Ultimately, a summary analysis is conducted to derive a representation u_A, reflecting the user's short-term interests, which is illustrated in Equation 12:

$$u_A = \sum_{i=1}^{N} \alpha_i v_{u,i} \tag{12}$$

where $v_{u,i}$ represents the user's N-order preference $\{v_{u,1}, v_{u,2}, \ldots v_{u,N}\}$, and α_i represents the attention weight coefficients. These coefficients are calculated based on the spatial relationships between entities in the knowledge graph as illustrated in Equation 13:

$$\alpha_i = \frac{\exp(h,r,t)}{\sum_{i=1}^{N} \exp(h,r,t)} \tag{13}$$

To more accurately capture the user's final interests, vectors generated by the embedding layer are used to represent the user's long-term interests. At the same time, aggregated vectors processed by the attention mechanism reflect the user's short-term interests. A GRU model is then applied to integrate both long-term and short-term interests, forming a comprehensive representation of the user's interest preferences. GRU is adopted due to its efficient gating mechanism and relatively lower computational complexity compared to LSTM, while maintaining comparable performance in modeling sequential dependencies. Unlike Transformer-based models, which typically require large-scale training data and extensive tuning, GRU provides a lightweight and effective solution for learning user preferences in data-constrained or latency-sensitive scenarios. The initial set of items interacting with the user is denoted as v_o.

User long-term preference representations u_L are trained using historical interaction data. These are then combined with short-term preferences u_A and trained through the GRU model. After training, the selected hidden layer undergoes normalization to produce the final representation of the user's preference u_G. Ultimately, the long-term preferences u_L are fused with the short-term preferences u_A, and a deep GRU model is used for training. The resulting hidden layers are normalized to generate the final user preference expression u_G as shown in Equation 14:

$$u_G = \sigma\left(W_4\left[u_L, u_A\right] + b_4\right) \tag{14}$$

3.2 Image feature extraction layer

In the image feature extraction layer, K-Means clustering is first applied to the image features corresponding to the user's interaction history, aiming to uncover latent user preference patterns. Specifically, image features are extracted using a pretrained VGG19 model (employing its static components from Conv1 to FC7 layers), and uniformly transformed into 4,096-dimensional vector representations. These feature vectors are then input into a K-Means clustering algorithm to generate a set of representative cluster centroids, each corresponding to a latent category of visual preference. The resulting clusters are utilized not only for modeling user interests in visual content but also as input for subsequent dynamic feature learning. To further capture semantic representations of these clustered features at the individual user level, a trainable feature modeling module composed of three fully connected layers is constructed. This module is designed to generate high-level semantic representations of images. The computational process is defined as follows, as shown in Equation 15:

$$\begin{aligned} s_0 &= \mathrm{Tanh}\left(W_a p_o + b_a\right) \\ p_v &= W_c P \cdot \mathrm{ReLU}\left(W_b s_0 + b_b\right) + b_c \end{aligned} \tag{15}$$

where $p_o \in \mathbb{R}^{4096}$ denote the initial image feature vector extracted via VGG19, and s_0 the intermediate semantic representation. The matrices W_a, W_b, and W_c, along with the corresponding biases b_a, b_b, and b_c, represent the weights and biases of the fully connected layers, respectively. The input p refers to the feature vector obtained after clustering, and p_v denotes the final semantic representation vector of the image.

In this section, the first 18 layers of the VGG19 network are fixed as output p_o, which is used to extract the visual features p_v of items. The images representing user-item interactions capture the user's visual preferences. Since the contribution of these historical interaction images to capturing user preferences varies, an image aggregation network is employed to integrate these images differentially. The specific structure of the aggregation network is shown in Figure 5. The network utilizes a multi-channel attention mechanism, where the item's image features and the initial representation of the target item serve as query vectors. These vectors are then passed through various fully connected networks to form individual weights and weighted vectors. The weighted vectors from the two channels are then aggregated to obtain a visual representation of the user's historical behavior, calculated as shown in Equation 16:

$$u_p = \varphi p_u^1 + (1-\varphi) p_u^2 \tag{16}$$

This represents the set p_i of images from items previously interacted with by the user. The image of the i_{th} item is transformed into a vector through embedding technology, which also includes the embedded representation of the target item image p_v. The training parameters are denoted by φ, and the formulas for p_u^1 and p_u^2 are as shown in Equation 17:

$$\begin{aligned} p_u^1 &= \sum_{i=1}^{N} g_p\left(p_i, p_v\right) p_i \\ p_u^2 &= \sum_{i=1}^{N} g_p\left(p_i, p_o\right) p_i \end{aligned} \tag{17}$$

3.3 Text feature extraction layer

This layer is responsible for extracting textual features. K-Means clustering is first applied to the user data before

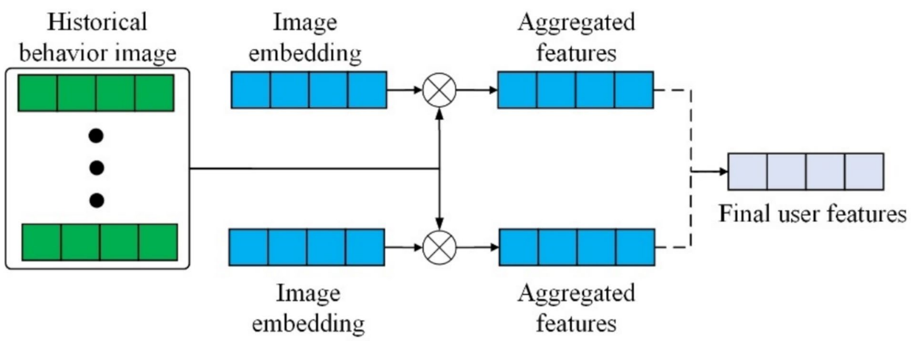

FIGURE 5
Structure of image feature aggregation.

proceeding with further operations. For text feature extraction, consider a movie as an example. A movie's textual features include its title, genre, and description, all of which are logically related. In this network layer, multi-head self-attention is employed to extract features from each component (movie title, genre, and description), which are then fused to form the final textual feature representation.

Regarding item categories, using movies as an example, when a user selects a movie for viewing in an app, they generally choose a broad genre, such as comedy, romance, or drama. If the user selects a comedy, they may further choose from sub-genres like slapstick, satirical, dark humor, or martial arts comedy. Similar patterns apply to other genres. Therefore, for category features, both broad and fine-grained categories are considered for feature extraction. During embedding, the ID is input and transformed into low-dimensional representations e^s and e^{ds}. The hidden category representation is then learned as shown in Equation 18:

$$\begin{aligned} B^s &= \mathrm{ReLU}\left(V_s \times e^s + v_s\right) \\ B^{ds} &= \mathrm{ReLU}\left(V_{ds} \times e^{ds} + v_{ds}\right) \end{aligned} \tag{18}$$

Next, the item title is used to obtain its feature representation. This part involves three steps. First, the word sequence of the item title is transformed into a low-dimensional semantic vector sequence, converting the sequence of title words $\left[w_1^t, w_2^t, \cdots, w_M^t\right]$ into a vector $\left[c_1^t, c_2^t, \cdots, c_M^t\right]$. The CNN component then extracts short-range contextual features from the words in the movie title. Using CNN, the context representation of the i_{th} word is derived as c_i^t, calculated as shown in Equation 19:

$$e_i^t = \mathrm{ReLU}\left(F_t \times c^t_{(i-X):(i+X)} + b_t\right) \tag{19}$$

where $c^t_{(i-X):(i+X)}$ represents the concatenation of word embeddings from positions $(i-X)$ to $(i+X)$. F_t and b_t are the kernel and bias parameters of the convolutional neural network. The CNN output sequence is $\left[e_1^t, e_2^t, \cdots, e_M^t\right]$. A similar approach is applied to the movie description sequence.

The final step uses multi-head self-attention to model the relationships between components, enabling better capture of distant textual features. The k_{th} attention head's representation of the i_{th} word is calculated as shown in Equation 20:

$$\begin{aligned} a_{i,j}^k &= \frac{\exp\left(e_i Q_k^w e_j\right)}{\sum_{m=1}^{M} \exp\left(e_i Q_k^w e_m\right)} \\ h_{i,j}^k &= V_k^w\left(\sum_{j=1}^{M} a_{i,j}^k e_j\right) \end{aligned} \tag{20}$$

where Q_k^w and V_k^w are the projection parameters of the self-attention head, and $a_{i,j}^k$ represents the relative importance of interactions between the i_{th} and j_{th} words. The multi-head representation h_i^w for the i_{th} word is obtained by concatenating the representations from the h independent attention heads.

Since the same word carries varying amounts of critical information across different item components, attention is used to assign weight proportions. The calculation for this is as shown in Equation 21:

$$a_i^w = \frac{\exp\left(a_i^w\right)}{\sum_{j=1}^{M} \exp\left(a_j^w\right)} \tag{21}$$

The final representation of each component is obtained by aggregating the weighted expressions of the words. The calculation is as shown in Equation 22:

$$B = \sum_{i=1}^{M} a_i^w h_i^w \tag{22}$$

Given that different components contain varying amounts of information—titles and descriptions may carry more relevant details, while categories more accurately represent the item's attributes—attention is employed to balance the weights, reflecting the amount of information each component carries. The calculation is as shown in Equation 23:

$$\begin{aligned} a_{tb} &= g_v^T \tanh\left(U_v^h \times B^{tb} + u_v\right) \\ a_{tb} &= \frac{\exp\left(a_{tb}\right)}{\exp\left(a_{sc}\right) + \exp\left(a_c\right) + \exp\left(a_{tb}\right)} \end{aligned} \tag{23}$$

Similarly, the attention weights for categories and sub-categories are denoted as a_s and a_{ds}. The final representation is the weighted sum of the component representations, as presented in Equation 24:

$$B = a_{tb} B^{tb} + a_{ds} B^{ds} + a_s B^s \tag{24}$$

Structure of text feature extraction is presented in Figure 6. Lastly, since user preferences may be related and users tend to browse items with similar categories, a multi-head self-attention mechanism is used to capture interactions between similar items, enhancing the representation of the user. Thek_{th} attention head's representation of the i_{th} item is calculated as presented in Equation 25:

$$\begin{gathered} \beta_{i,j}^{k} = \frac{\exp\left(B_i^T Q_k^n B_j\right)}{\sum_{m=1}^{M} \exp\left(B_i^T Q_k^n B_m\right)} \\ h_{i,j}^{n} = V_k^n \left(\sum_{j=1}^{M} \beta_{i,j}^{k} B_j \right) \end{gathered} \tag{25}$$

where Q_k^n and V_k^n are the self-attention head parameters, and $\beta_{i,j}^{k}$ represents the similarity between the j_{th} and i_{th} items. The multi-head representation for the i_{th} item is the concatenation of the representations from the h independent attention heads.

The user feature extraction part is shown in Figure 7. For user representation, the amount of user feature information carried by different items varies. Therefore, an attention mechanism is employed to better learn the user representation. The attention weight for the i_{th} item is calculated as presented in Equation 26:

$$\begin{gathered} a_i^n = q_n^T \tanh\left(V_n \times h_i^h + v_n\right) \\ a_i^n = \frac{\exp\left(a_i^n\right)}{\sum_{j=1}^{N} \exp\left(a_j^n\right)} \end{gathered} \tag{26}$$

3.4 Prediction layer

After processing through the collaborative knowledge graph neural layer, the image feature extraction layer, and the text feature extraction layer, user and item features are obtained. Let Q_u and Q_v represent the sets of user and item features, respectively. The calculation process is presented in Equation 27:

$$\begin{gathered} Q_u = \left\{u_G, u_p, T_u\right\} \\ Q_v = \left\{v_o, p_v, B\right\} \end{gathered} \tag{27}$$

These feature vectors are concatenated to form the final vector representation of the user and item, as expressed by Equation 28:

$$\begin{gathered} e_u = u_G \,||\, u_p \,||\, T_u \\ e_v = v_o \,||\, p_v \,||\, B \end{gathered} \tag{28}$$

The final step involves applying the dot product of the user and item vectors to estimate the target user's preference score for a specific item. The calculation is presented in Equation 29:

$$\hat{y}_{uv} = e_u^T e_v \tag{29}$$

4 Experimental results and analysis

4.1 Dataset description

Two publicly available cross-domain recommendation datasets were employed in this study, covering the domains of movies and books: *MovieLens-1M* and *Amazon-Book*. The MovieLens-1M dataset, provided by the GroupLens research group, has been widely used in movie recommendation research and contains user rating records for movies (Harper and Konstan, 2015). The Amazon-Book dataset, extracted from the book subset of the Amazon Review Corpus, captures user rating behaviors toward book products (He and McAuley, 2016). To ensure experimental consistency, interaction records in each dataset were partitioned into a training set (80%), a test set (10%), and a validation set (10%, sampled from the training set) for model tuning. Table 1 summarizes the key statistics of the two datasets, including the number of users, items, and interactions, as well as data sparsity. Additionally, it presents

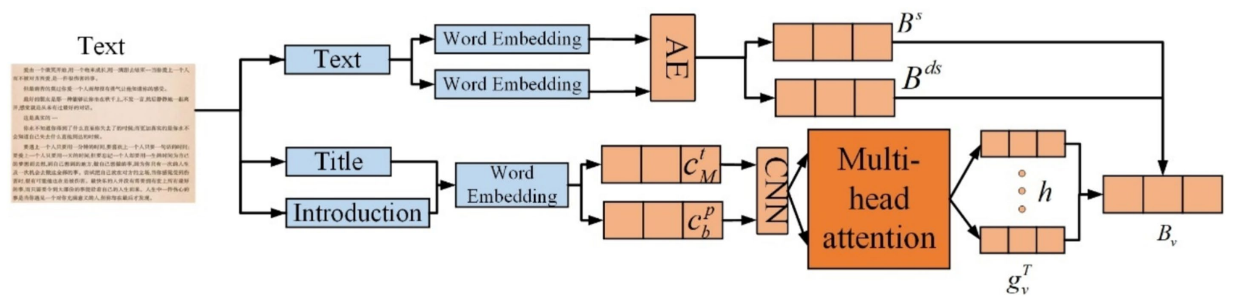

FIGURE 6
Structure of text feature extraction.

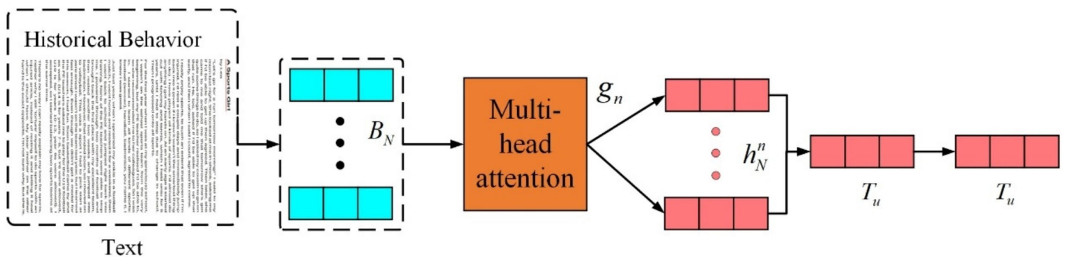

FIGURE 7
Structure of user feature extraction.

TABLE 1 Statistics of the datasets and corresponding knowledge graphs.

Dataset	User-project interaction				Knowledge graph related		
	Users	Items	Interactions	Sparsity	Entities	Relations	Triples
Movielens-1M	6,040	3,623	836,478	96.18%	73,988	51	385,923
Amazon-Book	70,679	24,915	847,733	99.95%	88,572	39	2,557,746

the corresponding KG statistics, including the number of entities, relations, and total triples.

From Table 1, it is observed that the Movielens-1M dataset has 836,478 interactions and a sparsity of 96.18%, with 385,923 triples in its knowledge graph. The Amazon-Book dataset contains 8,477,733 interactions, with a sparsity of 99.95% and 2,557,746 triples in its knowledge graph.

4.2 Performance metrics

The evaluation metrics used in this experiment are Recall and NDCG, both of which accurately describe recommendation performance.

4.2.1 Recall

This metric measures how many relevant items are correctly predicted within the top-X recommendations. It is computed as Equation 30:

$$\text{Recall} = \frac{TP}{\left(TP + FN\right)} \tag{30}$$

where TP represents the number of true positives, and FN represents the number of false negatives.

4.2.2 NDCG

NDCG gives more weight to higher-ranked results, reflecting the diminishing relevance of items further down the ranking. It is defined as Equation 31:

$$NDCG = \frac{DCG}{iDCG} \tag{31}$$

where ***DCG*** represents the discounted cumulative gain, and the weights of the arrangement order are summed. The earlier it is, the greater the proportion; ***DCG*** is the best arranged ***DCG***.

4.3 Baseline models and experimental setup

The following five baseline models were used in the experiments:

CKE (Zhang et al., 2016): A model based on collaborative filtering, which integrates text, image, and structural features within a single framework and uses TransR to enhance matrix factorization.

RippleNet (Wang et al., 2018): A model that continuously explores user preferences by incorporating a knowledge graph into the recommendation system, mitigating the cold-start problem.

KGNN-LS (Wang H. W. et al., 2019): This model constructs personalized graph representations for users using knowledge graphs, considering users' unique preferences for different relations within the knowledge graph. It introduces label smoothing to improve generalization.

KGAT (Wang X. et al., 2019): A model that uses attention mechanisms to aggregate higher-order information between entities, addressing sparsity issues and improving recommendation accuracy.

KGECF (Zhang et al., 2021): A knowledge graph-based recommendation system that extracts latent features related to items, creating a personalized knowledge subgraph. It employs an end-to-end collaborative learning framework to merge knowledge graph and user behavior data for higher accuracy.

In light of current trends in KG-based recommendation research, although several emerging models have been proposed, the aforementioned five baseline methods remain representative in key aspects such as multimodal fusion, graph-based modeling, and user preference construction. These methods have also been extensively validated across various datasets, providing a solid foundation for comparative analysis. To further enhance readability, Table 2 summarizes the modalities utilized and feature fusion strategies adopted by each model. This comparison highlights the superiority of the proposed method in terms of its multimodal fusion capabilities.

As shown in the table, the proposed model not only maintains strong collaborative modeling capabilities but also systematically integrates knowledge graph, image, and text modalities. By incorporating a more fine-grained multi-path attention mechanism

TABLE 2 Comparison of modality usage and feature fusion mechanisms across different recommendation models.

Model	Collaborative filtering	Knowledge graph	Graph modeling	Image modality	Text modality	Modality fusion mechanism
CKE (Zhang et al., 2016)	✓	✓	✗	✓	✓	Multimodal joint matrix factorization + TransR
RippleNet (Wang et al., 2018)	✓	✓	✗	✗	✗	KG path propagation mechanism
KGNN-LS (Wang H. W. et al., 2019)	✓	✓	✓	✗	✗	Label smoothing-based GCN with personalized aggregation
KGAT (Wang X. et al., 2019)	✓	✓	✓ (GAT)	✗	✗	Graph attention mechanism for high-order relations
KGECF (Zhang et al., 2021)	✓	✓	✓	✗	✗	Subgraph generation + joint collaborative modeling
Ours	✓	✓	✓ (HGNN + GRU)	✓	✓	Graph attention + GRU + multi-path visual attention + multi-head text attention

and a hierarchical structure, the model enables a more comprehensive representation of user preferences, thereby enhancing both recommendation accuracy and generalization performance.

The experiments were conducted on a Microsoft Windows 10 system, with an Nvidia 3,070 GPU, 32 GB of memory, and an AMD R7-6800H processor. The programming language used is Python 3.8.6, and PyTorch is utilized to implement the models. Item and user embedding sizes were set to 64, with a batch size of 256. The Adam optimizer is used with a learning rate of 0.002, and model parameters were initialized using the Glorot method. The training, validation, and test sets were split in an 80:10:10 ratio based on recommendation evaluation metrics.

4.4 Comparison of model performance

In this experiment, the proposed model is compared with the four baseline models across two datasets. Since recommendation accuracy is influenced by the number of recommendations, X is set to {5, 10, 15, 20, 25}. The performance results on the Movielens and Amazon-Book datasets are shown in Tables 3, 4, respectively.

The baseline comparison analysis for the two datasets shows that the recall rate (Recall) of all models increases with the number of recommendations (X). In this trend, the MKGAR model proposed in this study demonstrates the highest recall rate, significantly outperforming other models. This result confirms that the MKGAR model substantially improves the performance of recommendation systems.

Upon analyzing the experimental results, it is observed that the CKE and RippleNet models performed relatively poorly, while KGNN-LS, KGAT, and KGECF performed better. Specifically, the CKE model did not incorporate the TransR method, which led to an inability to fully capture the complex structure of the knowledge graph. This confirms the superiority of the TransR method in handling knowledge graph information. In contrast, RippleNet showed slight improvements over CKE by utilizing a knowledge graph, but it failed to fully explore higher-order connectivity and collaborative signals from users.

For the Recall metric, CKE and RippleNet exhibited similar performance across both datasets, while the proposed model showed a significant improvement over both. On the other hand, KGNN-LS, KGAT, and KGECF demonstrated clear advantages. KGNN-LS, which combines knowledge graphs and label smoothing regularization, improves recommendation accuracy and interpretability but focuses primarily on learning local structural features, potentially overlooking the impact of global information. The KGAT model, despite aggregating information by calculating spatial relationships between head and tail entities, did not adequately account for interactions between users. The KGECF model utilizes an attention mechanism to efficiently extract higher-order relational information from the knowledge graph, and integrates users' long-term and short-term preferences through gated neural networks, significantly improving recommendation accuracy. However, these models do not account for the potential influence of text and image features on the recommendations, resulting in lower performance compared to the proposed model. The proposed model leverages multi-perspective features—particularly textual and visual modalities—which play a crucial role in enhancing recommendation accuracy. This effectiveness has been empirically validated. For instance, in terms of Recall, the proposed model demonstrates significant improvements over KGNN-LS, KGAT, and KGECF on both the MovieLens-1M and Amazon-Book datasets. These results indicate that by effectively mining latent information from multiple modalities, the proposed approach exhibits strong recommendation capabilities.

Based on the results of hypothesis testing (two-sample *t*-test, $n = 5$, two-tailed test assuming equal variances), the *p*-values for the proposed method compared with each baseline across different recall positions on the two datasets are presented in Tables 5, 6. Significance levels are denoted as follows: $p < 0.05$ (*), $p < 0.01$ (**). The detailed results of the significance tests are shown below.

As observed from Tables 5, 6, the proposed method demonstrates statistically significant improvements over all baseline models under most settings, with $p < 0.01$ in the majority of comparisons. In particular, the comparison with KGECF at Recall@15 on MovieLens-1M and Recall@20 on Amazon-Book yields *p*-values slightly above the 0.01 threshold but still below 0.05, indicating moderate significance. The only non-significant result occurs at Recall@25 on MovieLens-1M when compared with KGECF

TABLE 3 Performance comparison on Movielens-1M dataset.

Model	Movielens-1M				
	Recall				
	5	10	15	20	25
CKE	0.0506 ± 0.0012	0.0627 ± 0.0011	0.0755 ± 0.0013	0.0805 ± 0.0016	0.0953 ± 0.0016
RippleNet	0.0479 ± 0.0010	0.0650 ± 0.0013	0.0801 ± 0.0014	0.0840 ± 0.0013	0.1033 ± 0.0015
KGNN-LS	0.1165 ± 0.0023	0.1521 ± 0.0018	0.1868 ± 0.0025	0.2251 ± 0.0027	0.2626 ± 0.0028
KGAT	0.1235 ± 0.0020	0.1647 ± 0.0024	0.2005 ± 0.0024	0.2417 ± 0.0026	0.2714 ± 0.0035
KGECF	0.1364 ± 0.0019	0.1771 ± 0.0021	0.2285 ± 0.0031	0.2669 ± 0.0027	0.2980 ± 0.0038
Ours	0.1440 ± 0.0017	0.1869 ± 0.0022	0.2337 ± 0.0028	0.2772 ± 0.0024	0.3034 ± 0.0031

TABLE 4 Performance comparison on Amazon-Book dataset.

Model	Amazon-Book				
	Recall				
	5	10	15	20	25
CKE	0.0438 ± 0.0010	0.0691 ± 0.0012	0.0737 ± 0.0012	0.0820 ± 0.0013	0.0888 ± 0.0013
RippleNet	0.0480 ± 0.0012	0.0674 ± 0.0015	0.0752 ± 0.0011	0.0841 ± 0.0013	0.0941 ± 0.0011
KGNN-LS	0.0852 ± 0.0014	0.1183 ± 0.0017	0.1244 ± 0.0015	0.1566 ± 0.0022	0.1775 ± 0.0024
KGAT	0.0952 ± 0.0016	0.1288 ± 0.0016	0.1389 ± 0.0018	0.1661 ± 0.0027	0.1804 ± 0.0029
KGECF	0.1068 ± 0.0016	0.1340 ± 0.0020	0.1507 ± 0.0022	0.1771 ± 0.0029	0.1849 ± 0.0026
Ours	0.1133 ± 0.0014	0.1415 ± 0.0017	0.1583 ± 0.0019	0.1850 ± 0.0027	0.1953 ± 0.0026

TABLE 5 *p*-values on the MovieLens-1M dataset.

Model	Movielens-1M				
	Recall@5	Recall@10	Recall@15	Recall@20	Recall@25
CKE	0.0000**	0.0000**	0.0000**	0.0000**	0.0000**
RippleNet	0.0000**	0.0000**	0.0000**	0.0000**	0.0000**
KGNN-LS	0.0000**	0.0000**	0.0000**	0.0000**	0.0000**
KGAT	0.0000**	0.0000**	0.0000**	0.0000**	0.0000**
KGECF	0.0003**	0.0000**	0.0238*	0.0011**	0.3232

(p = 0.3232). These significance testing results further verify the robustness and consistent performance improvements of the proposed model across different recall levels and datasets.

4.5 Ablation study

This section explores how user preferences related to visual and semantic features, derived from item images and text, can be integrated into item embeddings. To assess the specific impact of image and text features on the performance of the recommendation system, experiments were designed by modifying the user and item feature components in Equation 28. The models were then evaluated by removing either the image features (model labeled HGAN-MKG1) or the text features (model labeled HGAN-MKG2). By comparing these ablated models with the full model (MKGAR) across different top-K values (5, 10, 20, 25), we can gain detailed insights into the contribution of each modality to the recommendation results. The detailed metrics including NDCG and Recall are reported in Table 7.

As seen from the above results, both image and text features contribute positively to model performance across all top-K values. However, the magnitude of their impact is relatively modest. This is attributed to the redundancy and strong expressive power of the knowledge graph-based structural features, which may already encode rich item semantics and relationships. Thus, the additional visual and textual modalities, though helpful, offer only incremental improvements. These results highlight the robustness of the knowledge-aware representation while also confirming that multi-modal auxiliary features can enhance performance, especially under sparse or cold-start conditions.

4.6 Hyperparameter experiment

In the collaborative knowledge graph neural layer, the attention coefficient α_i plays a crucial role, as the attention score depends on the

TABLE 6 *p*-values on the Amazon-Book dataset.

Model	Amazon-Book				
	Recall@5	Recall@10	Recall@15	Recall@20	Recall@25
CKE	0.00000**	0.00000**	0.00000**	0.00000**	0.00000**
RippleNet	0.00000**	0.00000**	0.00000**	0.00000**	0.00000**
KGNN-LS	0.00000**	0.00000**	0.00000**	0.00000**	0.00001**
KGAT	0.00000**	0.00001**	0.00000**	0.00000**	0.00004**
KGECF	0.00013**	0.00021**	0.00038**	0.00212*	0.00023**

TABLE 7 Comparative experimental results of ablation study.

Top-K	Model	Movielens-1M (NDCG/Recall)	Amazon-Book (NDCG/Recall)
5	HGAN-MKG1	0.1771/0.1427	0.0581/0.0628
	HGAN-MKG2	0.1756/0.1425	0.0578/0.0623
	MKGAR	0.1783/0.1440	0.0594/0.1133
10	HGAN-MKG1	0.2202/0.1846	0.0753/0.1395
	HGAN-MKG2	0.2217/0.1841	0.0753/0.1402
	MKGAR	0.2240/0.1869	0.0774/0.1415
15	HGAN-MKG1	0.2858/0.2330	0.0844/0.1568
	HGAN-MKG2	0.2840/0.2321	0.0846/0.1573
	MKGAR	0.2875/0.2337	0.0869/0.1583
20	HGAN-MKG1	0.3343/0.2746	0.1045/0.1840
	HGAN-MKG2	0.3348/0.2754	0.1037/0.1836
	MKGAR	0.3361/0.2772	0.1064/0.1850
25	HGAN-MKG1	0.3759/0.3005	0.1086/0.1939
	HGAN-MKG2	0.3762/0.3009	0.1081/0.1935
	MKGAR	0.3777/0.3034	0.1114/0.1953

spatial relationship between entities in the knowledge graph. To evaluate the impact of the attention coefficient on the model performance, we varied the value of α within the range [0, 1] and compared the results with those obtained from the original model using α_i. The performance is evaluated using NDCG on the Amazon-Book dataset with the number of recommendations set to 20. The changes in NDCG with different values of α\alphaα are shown in Figure 8, illustrating the impact of different attention coefficients on model performance.

The analysis of the data in Figure 8 shows that when the attention coefficient α is set to the value of α_i, the NDCG reaches its peak at 0.3348. No value of α within the range [0, 1] exceeded this performance. Specifically, as α increases, the performance initially rises to a peak and then begins to decline. The highest NDCG is observed at α=0.4 with a value of 0.3346, while at α=1, the performance dropped to 0.3312. Notably, when α=1, the performance improved by 1.09% compared to the original α_i value. This result indicates that dynamically adjusting the attention coefficient based on the spatial relationship between

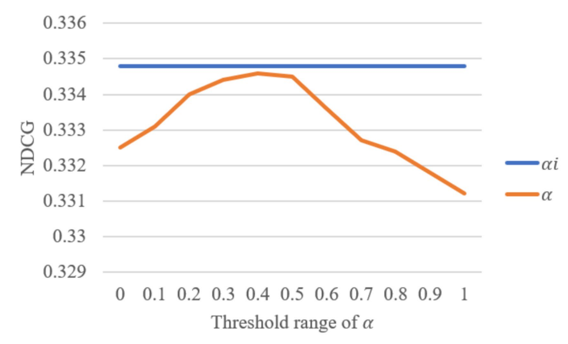

FIGURE 8
Impact of the attention coefficient α in the knowledge graph attention network layer on model performance.

entities, rather than using a static threshold, can more effectively enhance model performance. This is because a smaller threshold may introduce noise from irrelevant entities, while a higher threshold may excessively filter out nodes, reducing the amount of information. Therefore, dynamically adjusting the attention coefficient to align with the spatial relationships of entities is an effective strategy for optimizing recommendation system performance.

5 Conclusion

This paper introduces a recommendation method that combines hierarchical graph attention networks and multimodal knowledge graphs. The core components of the approach include the integration of knowledge graphs with graph neural attention mechanisms and multimodal feature fusion. In the collaborative knowledge graph neural layer, the knowledge graph serves as a structured representation, encompassing a wealth of entities, relationships, and attributes, while graph neural networks help uncover deeper collaborative relationships. Additionally, by incorporating image and text features (such as movie posters, book covers, names, descriptions, and categories), the model enhances its understanding of users' visual and semantic preferences. This approach not only effectively models user interests but also provides more accurate recommendations. Extensive experiments on the MovieLens and Amazon-Book datasets demonstrate that the proposed model significantly outperforms other knowledge graph-based recommendation models in terms of performance.

Data availability statement

The datasets presented in this study can be found in online repositories. The names of the repository/repositories and accession number(s) can be found at: https://grouplens.org/datasets/movielens/.

Author contributions

XH: Conceptualization, Writing – original draft. XD: Software, Writing – review & editing.

Funding

The author(s) declare that no financial support was received for the research and/or publication of this article.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Generative AI statement

The authors declare that no Gen AI was used in the creation of this manuscript.

Publisher's note

References

Bandyopadhyay, S., Thakur, S. S., and Mandal, J. K. (2024). Emotion detection for online recommender system using deep learning: a proposed method. *Innov. Syst. Softw. Eng.* 20, 719–726. doi: 10.1007/s11334-022-00437-7

Bock, H. H. (2007). Clustering methods: a history of k-means algorithms. In P. Brito, P. Bertrand, G. Cucumel, F. Carvalho and Y. Escoufier, eds., Selected contributions in data analysis and classification, *34*: 161–172, Springer, Cham

Chen, T., and Wong, R. C. W. (2021). An efficient and effective framework for session-based social recommendation. Proceedings of the 14th ACM International Conference on Web Search and Data Mining. Virtual Event, Israel: ACM, 400–408.

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., et al. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. arXiv [Preprint]. *arXiv:1406.1078*.

Fan, W., Ma, Y., Li, Q., He, Y., Zhao, E., Tang, J., et al. (2019). Graph neural networks for social recommendation. The World Wide Web Conference, 417–426.

Gao, C., Cai, G., Jiang, X., Zheng, F., Zhang, J., Gong, Y., et al. (2022). Conditional feature learning based transformer for text-based person search. *IEEE Transactions on Image Processing*, 31, 6097–6108.

Guo, Q., Zhuang, F., Qin, C., Zhu, H., Xie, X., Xiong, H., et al. (2020). A survey on knowledge graph-based recommender systems. *IEEE Trans. Knowl. Data Eng.* 34, 3549–3568. doi: 10.1109/TKDE.2020.3028705

Han, D., Qi, H., Wang, S., Hou, D., and Wang, C. (2024). Adaptive stepsize forward–backward pursuit and acoustic emission-based health state assessment of high-speed train bearings. *Struct. Health Monit.* 14759217241271036.

Harper, F. M., and Konstan, J. A. (2015). The movielens datasets: history and context. *ACM Trans. Intell. Syst. Technol.* 5, 1–19.

He, R., and McAuley, J. (2016). Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. Proceedings of the 25th international conference on world wide web, 507–517.

Javed, U., Shaukat, K., Hameed, I. A., Iqbal, F., Mahboob Alam, T., and Luo, S. (2021). A review of content-based and context-based recommendation systems. *Int. J. Emerg. Technol. Learn.* 16, 274–306. doi: 10.3991/ijet.v16i03.18851

Jiang, J., Zhao, H., He, M., Wu, L., Zhang, K., Fan, J., et al. (2023). Knowledge-aware cross-semantic alignment for domain-level zero-shot recommendation. Proceedings of the 32nd ACM International Conference on Information and Knowledge Management, 965–975.

Li, C., Niu, X., Luo, X., Chen, Z., and Quan, C. (2019). A review-driven neural model for sequential recommendation [A], the 28th International Joint Conference on Artificial Intelligence, 2866–2872.

Liu, Y., Zhou, X., Kou, H., Zhao, Y., Xu, X., Zhang, X., et al. (2024). Privacy-preserving point-of-interest recommendation based on simplified graph convolutional network for geological traveling. *ACM Trans. Intell. Syst. Technol.* 15, 1–17. doi: 10.1145/3620677

Ma, W., Zhang, M., Cao, Y., Jin, W., Wang, C., Liu, Y., et al. (2019). Jointly learning explainable rules for recommendation with knowledge graph. The World Wide Web Conference, 1210–1221.

Oramas, S., Ostuni, V. C., Noia, T. D., Serra, X., and Di Sciascio, E. (2016). Sound and music recommendation with knowledge graphs. *ACM Trans. Intell. Syst. Technol.* 8, 1–21. doi: 10.1145/2926718

Shi, C., Han, X. T., Song, L., Wang, X., Wang, S., Du, J., et al. (2021). Deep collaborative filtering with multi-aspect information in heterogeneous networks. *IEEE Trans. Knowl. Data Eng.* 33, 1413–1425. doi: 10.1109/TKDE.2019.2941938

Steck, H., Baltrunas, L., Elahi, E., Liang, D., Raimond, Y., and Basilico, J. (2021). Deep learning for recommender systems: a Netflix case study. *AI Mag.* 42, 7–18. doi: 10.1609/aimag.v42i3.18140

Sun, R., Cao, X., Zhao, Y., Zhou, K., Zhang, F., Wang, Z., et al. (2020). Multi-modal knowledge graphs for recommender systems. Proceedings of the 29th ACM international conference on information & knowledge management, 1405–1414.

Visa, M. R., and Patel, D. B. (2021). A deep learning approach of collaborative filtering to recommender system with opinion mining. Rising threats in expert applications and solutions: proceedings of FICR-TEAS 2020. Springer, Singapore, 119–131

Wang, X., He, X. N., Cao, Y. X., Liu, M., and Chua, T. S. (2019). KGAT: knowledge graph attention network for recommendation [C], Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2019, 950–958.

Wang, X., Jiang, H., Mu, M., and Dong, Y. (2025). A dynamic collaborative adversarial domain adaptation network for unsupervised rotating machinery fault diagnosis. *Reliab. Eng. Syst. Safety.* 255:110662.

Wang, H., Liu, Z., and Han, Z. (2024). HO2RL: A novel hybrid offline-and-online reinforcement learning method for active pantograph control. *IEEE Transactions on Industrial Electronics*. doi: 10.1109/TIE.2024.3477002

Wang, H., Zhang, F., Wang, J., Zhao, M., Li, W., Xie, X., et al. (2018). RippleNet: Propagating user preferences on the knowledge graph for recommender systems. International Conference on Information and Knowledge Management, Proceedings, 417–426.

Wang, H. W., Zhang, F. Z., Zhang, M. D., Leskovec, J., Zhao, M., Li, W. J., et al. (2019). Knowledge-aware graph neural networks with label smoothness regularization for recommender systems [C], Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ser. KDD'19, 968–977.

Wu, L., Li, Z., Zhao, H., Huang, Z., Han, Y., Jiang, J., et al. (2024). Supporting your idea reasonably: a knowledge-aware topic reasoning strategy for citation recommendation. *IEEE Trans. Knowl. Data Eng.* 36, 4275–4289. doi: 10.1109/TKDE.2024.3365508

Wu, L., Li, Z., Zhao, H., Wang, Z., Liu, Q., Huai, B., et al. (2023). Recognizing unseen objects via multimodal intensive knowledge graph propagation. Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 2618–2628.

Yamada, K. D., Lin, F., and Nakamura, T. (2021). Developing a novel recurrent neural network architecture with fewer parameters and good learning performance. *Interdiscip. Inf. Sci.* 27, 25–40.

Yan, J., Cheng, Y., Zhang, F., Zhou, N., Wang, H., Jin, B., et al. (2025). Multi-modal imitation learning for arc detection in complex railway environments. *IEEE Transactions on Instrumentation and Measurement.* doi: 10.1109/TIM.2025.3556896

Yang, Y., Huang, C., Xia, L., and Li, C. (2022). Knowledge graph contrastive learning for recommendation. Proceedings of the 45th international ACM SIGIR conference on research and development in information retrieval, 1434–1443.

Zhang, Y. H., Wang, W., Liu, H. Z., Gu, R., and Hao, Y. (2021). Collaborative filtering recommendation algorithm based on knowledge graph embedding. *Appl. Res. Comput.* 38, 3590–3596.

Zhang, S., Yao, L., Sun, A., Tay, Y. (2019). Deep learning based recommender system: a survey and new perspectives. *ACM Comput. Surv.* 51, 1–49. doi: 10.1145/3158369

Zhang, F., Yuan, N. J., Lian, D., Xie, X., and Ma, W. Y. (2016). Collaborative knowledge base embedding for recommender systems, Processing of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 353–362.

Zhang, J. C., Zain, A. M., Zhou, K. Q., Chen, X., and Zhang, R. M. (2024). A review of recommender systems based on knowledge graph embedding. *Expert Syst. Appl.* 250:123876. doi: 10.1016/j.eswa.2024.123876