Graph Neural Network Based Personalized Recommendation with Edge Enrichment Using Ratings and Review Sentiments

Anu Mathews [1,*] and Sheba Selvam [2]

[1] Computer Science and Engineering (Data Science), RNS Institute of Technology, Affiliated to Visvesvaraya Technological University, Belagavi-590018, Karnataka, India

[2] Department of Artificial Intelligence and Machine Learning, B. N. M. Institute of Technology, Affiliated to Visvesvaraya Technological University, Belagavi-590018, Karnataka, India

Email: anu.mathews1977@gmail.com (A.M.); shebaselvam@bnmit.in (S.S.)

*Corresponding author

***Abstract*—Visual features and user-item interaction graphs enable recommendation systems to model complex relationships and improve accuracy. We propose Heterogeneous Graph Recommender Network with Bayesian Personalized Ranking for Fashion (HGRN-BPR-F), a Graph Neural Network that places multimodal edge features, explicit ratings and review sentiments, directly on user-item interactions. Unlike prior work aggregating features at nodes, our edge-centric design preserves interaction-specific context. The graph data is extracted from a real-world dataset of Amazon Fashion and exhaustive preprocessing is done to include relevant features for both users and products. Model training employs the Bayesian Personalized Ranking (BPR) loss function, which optimizes the ranking between positive and negative user–item pairs, with hard negatives used to enhance learning effectiveness. Our model achieves AUC of 0.885 ±0.010 evaluated over three seeds, demonstrating strong ranking performance with a 10.2% improvement over the rating-only baseline. Ablation experiments further demonstrate that combining ratings and sentiment as edge features substantially outperforms single-modality and node-feature-based configurations, thereby validating our edge-centric multimodal architecture.**

***Keywords*—multimodal recommendation, Bayesian Personalized Ranking (BPR) loss, cosine similarity, Graph Neural Network (GNN), heterogeneous graphs**

I. INTRODUCTION

The rapid expansion of e-commerce has produced massive volumes of data, presenting both significant opportunities and challenges for recommender systems that provide personalized product suggestions without requiring explicit user queries. Traditional content-based and collaborative filtering approaches struggle to fully capture the complex relationships and diverse attributes inherent in user–item interactions [1, 2]. Although neural network–based models have attempted to alleviate these limitations [3–5], they still overlook higher-order structural information in the data during both training and prediction.

Graph Neural Networks are highly effective for modelling complex and heterogeneous data structures. They can capture intricate relationships between real-world entities and their attributes when the data is represented in graph form. In recommender systems, GNNs provide a powerful way to exploit the rich connectivity information embedded in user–item interaction graphs. This paper introduces a heterogeneous Graph Neural Network (GNN) model tailored for personalized recommendation tasks. The model operates on a bipartite graph composed of user and product nodes, where directed, rating-weighted edges represent user–product interactions. By incorporating GraphSAGE, GENConv, and HeteroConv layers, the model effectively captures the complex relationships between users and products, enabling the learning of expressive user and item embeddings that improve recommendation accuracy. The rise of GNNs stems from advances in Graph Representation Learning (GRL) and Convolutional Neural Networks. While CNNs excel at extracting local features from Euclidean data such as images and text, they are less effective on non-Euclidean structures like graphs, where the elements of operation (e.g., nodes) do not have a fixed size or ordering. GRL addresses this limitation by generating low-dimensional embeddings for graph components—such as nodes and edges—that capture their complex interconnections. For instance, Deep Walk learns node embeddings by applying the Skip-Gram model to sequences of nodes obtained through random walks on the graph [6].

GNNs offer several key advantages for recommendation systems. They naturally model multi-hop relationships, enabling the system to understand indirect connections along paths in the user-item bipartite graph. For instance, when a target user has interacted with certain items, and other users with similar interaction patterns have engaged with additional items, the GNN can propagate this information through the graph structure to

Manuscript received September 15, 2025; revised October 24, 2025; accepted November 4, 2025; published March 26, 2026.

discover relevant recommendations. This capability to capture transitive relationships and higher-order collaborative patterns improves both recommendation accuracy and diversity. GNNs can seamlessly integrate diverse data types—including textual, visual, and categorical attributes—by incorporating them as node features, thereby supporting effective multi-modal learning. Furthermore, they learn user-specific preferences more effectively by aggregating information from a user's neighborhood in the graph, which enhances the ability to deliver personalized recommendations tailored to individual tastes and behaviors.

This paper addresses three open research questions that remain underexplored in the literature. First, we investigate the benefits that heterogeneous GNNs offer for recommender systems compared to traditional recommendation approaches. Second, we examine how constructing user features through the weighted aggregation of features from previously interacted items influences the performance of GNN-based recommender systems. Third, we explore how edge attributes can be effectively integrated into heterogeneous GNNs to enhance recommendation accuracy.

Traditional recommender systems struggle to capture the intricate relationships among users, items, and their associated features—particularly in fashion recommendation, where visual aesthetics significantly influence user preferences. Most existing works that integrate user ratings and reviews treat these modalities in isolation, without fully combining multiple heterogeneous signals such as explicit ratings, sentiment-rich user reviews, and image features into a unified framework. Prior visual-aware GNNs (e.g., VBPR, MMGCN) use visual features only as node attributes, but they do not leverage the potential of enriching graph edges with complementary information. We address this gap by developing a multi-dimensional edge feature representation that integrates explicit ratings with review sentiment scores, complemented by image-based product embeddings as node attributes. In contrast to existing approaches that examine modalities separately, we systematically evaluate the individual and combined contributions of ratings, sentiments, and visual features within a single unified framework, revealing their effects on recommendation quality.

The primary objective of this work is to develop an image-based recommendation system that accurately predicts user preferences by leveraging their past item interactions and the visual attributes of products, while effectively integrating multiple modalities of information. To achieve this goal, we propose a GNN-based recommendation framework with several novel contributions. First, a directed graph is constructed in which user node features are computed as a weighted average of the features of previously interacted items, providing a more accurate and distinctive representation of user preferences compared to existing methods. Second, we apply sentiment weighting using BERT confidence scores and aggregate multiple reviews for each user–item pair to capture nuanced user opinions. Third, we utilize GraphSAGE to model the user–item interaction graph for top-N recommendations and employ GENConv for predicting user ratings. Fourth, we systematically examine four types of edge attribute configurations—ratings only, sentiments only, ratings combined with sentiments, and no edge features—to assess their individual and combined effects on recommendation performance. Fifth, we investigate the impact of dimensionality reduction via Principal Component Analysis on high-dimensional image features extracted using VGG19. Finally, we evaluate the framework using multiple metrics including AUC and RMSE to provide a comprehensive understanding of performance trade-offs across different recommendation tasks. Through extensive experiments on both top-N recommendation and rating prediction tasks, we demonstrate the effectiveness of our Heterogeneous Graph Recommender model.

We conduct a systematic investigation comparing different edge feature configurations in GNN-based recommender systems, while also analyzing the influence of dimensionality reduction on rich image-based content features. Extensive experiments on both top-N recommendation and rating prediction tasks demonstrate the effectiveness of our Heterogeneous Graph Recommender model.

The remaining parts of the paper are organized in the following way. We discuss the various research works in the area of Image Based recommender systems in Section II. The basic concepts in GNN based recommender systems are discussed in Section III. Section IV focuses on the framework and the methodology utilized in image recommendation. Section V gives the details of implementation, followed by the results of this work. Section VI gives a conclusion of the paper by mentioning the research issues in the domain of image recommendation and providing directions for future research.

II. Related Work

Several recent studies have explored the use of GNNs in the fashion industry for outfit compatibility analysis and product recommendation.

Fashion item compatibility prediction is enhanced by effectively incorporating high-dimensional visual features into graph-based learning [7]. It uses an autoencoder to compress CNN-extracted visual features into lower-dimensional representations, improving both feature expressiveness and computational efficiency. VBPR introduces a scalable factorization model that extracts visual features from product images using pre-trained deep neural networks and learns an additional layer to capture the visual dimensions most relevant to user preferences [8]. Based on the Bayesian Personalized Ranking framework, a model termed DeepStyle incorporates style representations by removing category information from CaffeNet-derived visual features, enabling a more accurate understanding of user style preferences [9]. A novel attention-based collaborative filtering model incorporates both a component-level module for selecting key content features from multimedia items and an item-level module for scoring item preferences [10]. In a similar multimodal direction [11], a fashion-aware framework jointly learns image representations from pixel data within the recommendation model, facilitating the generation of new fashion items that align with a user's style. Graph Neural Networks have shown exceptional recommendation performance by leveraging embedding propagation to iteratively aggregate information from neighboring nodes, thereby enabling nodes to learn from higher-order connections through multi-layer message propagation. This strength in modeling structural information is exemplified by methods that model relational data on the inherently heterogeneous user-item bipartite graph [12, 13]. Furthermore, rating prediction has been advanced through models employing CNN, LSTM, and attention mechanisms [14], to effectively utilize user reviews. To address interaction uncertainty, particularly for cold-start users, a Bayesian Graph Convolutional Network framework [15], has also been proposed to enhance recommendation accuracy

Neural graph filtering models fashion item relationships using GNNs, achieving over 10% AUC improvement and 82.5% user preference for diverse-style recommendations [16]. Preference predictions are enhanced in visually-aware recommender systems [17], by extracting style features from item images and using attention mechanisms to model users' personalized importance. A graph attention network that assigns pairwise attention coefficients to model implicit correlations among neighboring nodes is introduced by NGAT4Rec [18], improving representation learning in recommendation systems.

Variational Autoencoder is applied to enable interactive fashion product generation, style-based retrieval, and content-based recommendations [19]; meanwhile, fashion item compatibility is modeled through a category-aware metric learning approach that projects item embeddings into relation spaces based on category pairs and leverages a relation transition vector [20].

Hybrid recommender systems integrate two or more recommendation techniques in a structured way to provide more accurate and personalized suggestions for users [21–28]. The integration of review sentiments and ratings into the recommendation task has been explored in many works [29–37], with some employing GNNs in this context [29, 33]. Tensor factorization–based recommender systems capture multi-dimensional relationships, such as user–item–context interactions, by decomposing interaction tensors into latent factors, thereby enabling richer and context-aware recommendations [38–47].

BERT-based recommender systems leverage deep bidirectional transformers to model sequential user behavior and contextual item relationships, which facilitates highly accurate and context-aware recommendations [48–54]. GAN-based recommender systems utilize generative adversarial networks, where a generator and a discriminator are trained adversarially to learn realistic user–item interactions and enhance recommendation quality [55–58].

Deep learning enhances recommender systems by extracting meaningful patterns from diverse data sources, using models such as CNNs, RNNs, and GNNs to interpret images, sequential behaviors, and relational structures [59−64]. Deep reinforcement learning extends this capability by framing recommendations as sequential decision-making tasks, enabling the system to adapt dynamically and sustain user engagement over time [65–68]. Large Language Models (LLMs) have been used to further enhance recommendation quality by leveraging their advanced language understanding to generate suggestions aligned with users' needs, contexts, and preferences [69–79]. Sentiment analysis has been increasingly integrated into recommender systems [80, 81].

Only a few studies have systematically compared the impact of using (i) ratings alone, (ii) sentiments alone, (iii) ratings combined with sentiments, and (iv) no edge features within a unified GNN framework. Additionally, the influence of dimensionality reduction on high-dimensional visual embeddings—commonly present in fashion and multimedia applications—remains largely underexplored in GNN-based recommendation models. Prior research also rarely investigates task-specific trade-offs, such as performance differences between top-N recommendation and rating prediction across various edge feature settings. Therefore, integrating visual features with user–item interaction graphs enriched with ratings and sentiment information can reveal deeper behavioural patterns, enabling more accurate and personalized recommendations.

III. Fundamental Concepts

Graph Neural Networks are built to understand the patterns from data that is structured in the form of a graph, where the relationships between entities (nodes) are as important as the attributes of the entities themselves. The main principle behind GNNs is to capture the local and global graph structures by updating the embedding of each node by collecting information from the neighboring nodes.

A. Graph Representation

A graph structure G' of the form $G' = (V', E')$ consists of nodes V' and edges E'. Each node $v \in V'$ has a vector of its features, xv, which contains information about the node. An edge $euv \in E'$ between nodes u and v represents an interaction between these nodes. Adjacency matrix A is used to denote the structure of a graph, where $Auv = 1$ if there is an edge connecting u and v, and 0 in case of no edge between u and v.

B. Message Passing

During message-passing, which is the core step in a GNN, nodes enhance their feature representations by integrating information from adjacent nodes. This phase typically has the following steps:

Message Computation: For each node v, the GNN computes messages from its neighbors u using a message function $m_{uv} = \text{Message}(h_u, h_v, e_{uv})$, where h_u and h_v are the nodes' feature vectors and e_{uv} is the edge feature between u and v (if any).

Message Aggregation: The node v aggregates the messages from all its neighbors by utilizing an aggregation function, a_v = Aggregate($\{m_{uv}|u \in N(v)\}$), where $N(v)$ denotes the neighbors of node v. The aggregation function could be a simple sum, mean, or a more complex pooling operation.

Node Update: After aggregation, the node representation is updated with an update function, $h_v^{(k)}$ = Update ($h_v^{(k-1)}$, a_v), where $h_v^{(k)}$ is the node feature at the layer k (or iteration), and $h_v(0) = x_v$ is the initial node feature.

This process is repeated for a fixed number of iterations or layers and information is propagated across the graph, thereby allowing each node to gather knowledge from further distant nodes.

C. GNN Models

There are several variants of GNNs with improved flexibility, efficiency and performance. Some of the popular variants include:

Graph Convolutional Networks (GCN): These networks perform convolution-like operations on graph structures, where node features are aggregated from their neighbors using a weighted sum derived from the adjacency matrix [82]. At layer l, a node's embedding is computed as the normalized average of its neighbors' embeddings, scaled by the node's degree. The GCN framework applies a message transformation step, followed by message aggregation, to learn updated node representations.

The equation for GCN is:

$$h_v^{(l)} = \sigma\left(W \cdot \frac{1}{|N(v)|} \sum_{u \in N(v)} h_u^{(l-1)}\right) \quad (1)$$

where $N(v)$ denotes the neighbors of node v, W is the weight matrix for message transformation and σ denotes the non-linear activation function, commonly chosen as ReLU.

Graph Attention Networks (GAT): GAT makes use of an attention technique, allowing the network to focus on more relevant connections by weighing the importance of different neighbors during the aggregation [83].

Eq. (2) is:

$$h_v^{(l)} = \sigma\left(\sum_{u \in N(v)} \propto_{uv} W h_u^{(l-1)}\right) \quad (2)$$

$h_v^{(l)}$: The updated embedding of the node v at layer l.
$h_v^{(l-1)}$: The embedding of neighboring node u from the previous layer (l−1).
$N(v)$: The set of all neighbors of node v.
W: A trainable weight matrix that transforms the node embeddings.
$\propto_{uv}$: The attention coefficient that determines how much influence node u has on node v. It is computed using an attention mechanism (typically a learned function, such as a SoftMax over the neighbors).
σ: Non-linear activation function.

GraphSAGE: This model samples a fixed number of neighbors for each node and aggregates their features, improving scalability for large graphs [84]. GraphSAGE extends GCN by introducing more flexible aggregation strategies and enabling the concatenation of a node's own representation with the aggregated messages from its neighbors, as shown in Eq. (3). This concatenation adds more expressiveness and boosts the model's performance ability to capture node-specific details.

$$h_v^{(l)} = \sigma\left(W \times Concat\left(h_v^{(l-1)}, Aggr\left(h_u^{(l-1)} \forall u \in N(v)\right)\right)\right) (3)$$

$h_v^{(l)}$, $h_v^{(l-1)}$, $N(v)$, W, and σ are as mentioned in Eq. (2).
$h_u^{(l-1)}$: Embeddings of neighboring nodes u at layer (l−1).
Aggr: A function that aggregates information from the neighbors $u \in N(v)$.

Common aggregation functions include:
Mean: Taking the average of neighbor embeddings.
Sum: Summing up neighbor embeddings.
Max: Taking the element-wise max.
Concat: Concatenation of v's previous embedding $h_v^{(l-1)}$ with the aggregated neighbor embeddings.

Graph Isomorphism Networks (GINs): GINs are developed to match the power of the Weisfeiler-Lehman graph isomorphism test, enabling them to effectively distinguish non-isomorphic graphs [83].

IV. Methodology

The Amazon Reviews dataset has 48.19 million items, and also 571.54 million reviews taken from 54.51 million users. This work utilizes the Amazon Fashion dataset (https://mcauleylab.ucsd.edu/public_datasets/data/amazon_2023), which consists of the user-item interactions, ratings and the associated product metadata consisting of name of the product, average rating of the product shown on the product page, description of the product, price and image of the product. A random sample of 50,000 rows is selected to get 49584 users and 41,355 items. Pre-trained VGG19 extracts 4096-dimensional feature vectors from the product images, which are then used as item features in the GNN model after reducing the dimensionality to 1,589 using PCA.

The Image Recommender System framework is shown in Fig. 1. The different modules include Feature Extraction, Dimensionality Reduction (PCA), User Item Graph Creation, GNN model Implementation, model training and evaluation.

Algorithm 1 presents a heterogeneous GNN recommendation system that processes user-item bipartite graphs with rating-weighted edges and bidirectional message passing to improve recommendation accuracy.

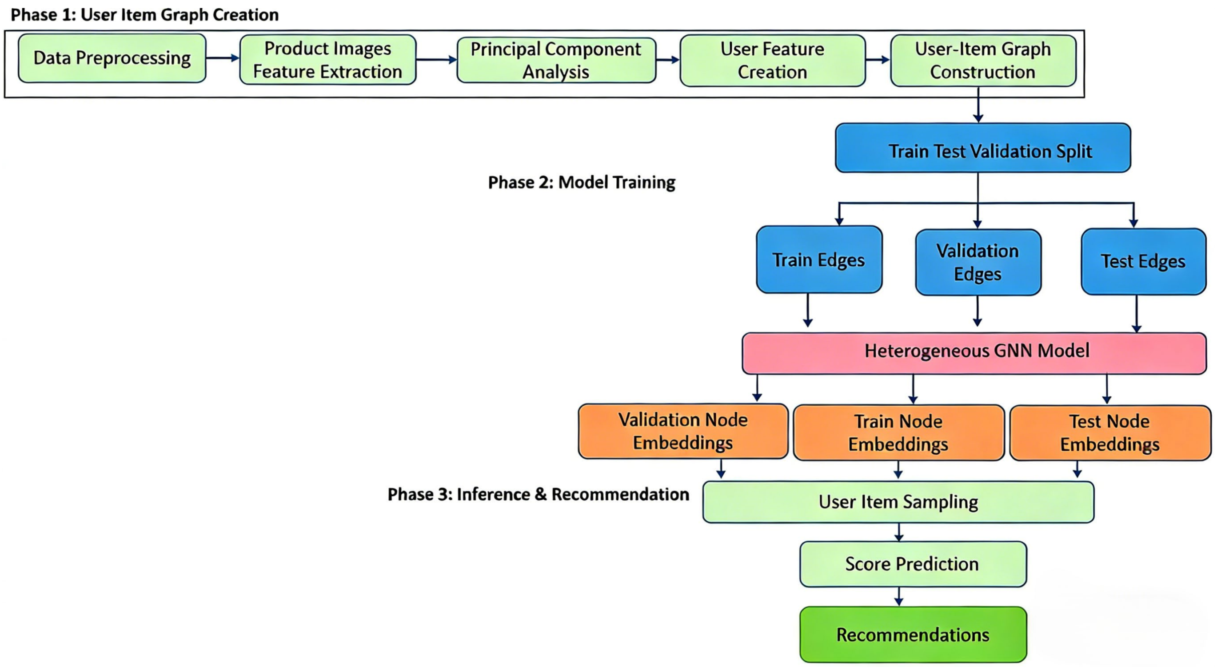


Fig. 1. Image recommendation framework.

Algorithm 1: Heterogeneous GNN Recommender System for User-Item Recommendation

```
Input: A bipartite user–item graph G with user and product nodes, each having 1589-dimensional feature vectors, and edges weighted by a rating attribute.
Output:
- Trained heterogeneous GNN model
- Evaluation metrics (Recall@20, AUC, etc.) on training, validation, and test sets.
Begin
1. Load the user–item graph G.
2. Initialize an empty directed graph G_directed.
3. For each edge (u, v) in G do
4.    If u is a user node and v is a product node then
5.        Add a directed edge (u, v) to G_directed.
6.    End if
7. End for
8. Map all user and product node IDs to integer indices.
9. Initialize a HeteroData object.
10. For each user–product edge (u, v) with weight w do
11.   Store edge index ($u_{idx}$, $v_{idx}$) and edge attribute w.
12. End for
13. Split edges into training (70%), validation (15%), and test (15%) sets.
14. For each user–product edge (u, v) do
15.    Add a reverse product–user edge (v, u).
16. End for
17. Assign node features and edge indices/attributes for both directions in each split.
18. Define the Heterogeneous Recommender GNN model:
19.    Use two-layer HeteroConv with SAGEConv for each relation.
20.    Add fully connected layers fc1 (64→128), dropout, and fc2 (128→64).
21.    Define forward and predict functions.
22. Set hyperparameters: hidden_dim=64, lr=0.001, epochs=1000, weight_decay=1e-5.
23. Initialize the Adam optimizer.
24. For each epoch do
25.    Set model to training mode and reset gradients.
26.    For each batch do
27.        Sample a user, positive item, and hard negative item.
28.        Compute node embeddings.
29.        Predict scores for positive and negative pairs.
30.        Compute BPR loss.
31.        Backpropagate and update parameters.
32.    End for
33. End for
End
```

A. Feature Extraction & Dimensionality Reduction

Product images are retrieved using the URLs provided in the metadata file, with a multithreaded approach employed to accelerate the download process. The downloaded images are then processed using the VGG19 model (Fig. 2) to extract visual features, which serve as the product representations within the graph. Principal Component Analysis (PCA) eliminates less significant components that are more likely to represent noise, thereby enhancing model efficiency and reducing the risk of overfitting. In this work, we find the number of components in PCA by setting a 95% threshold for the Cumulative Explained Variance ratio, ultimately determining that a minimum of 1589 components are needed to meet this threshold.

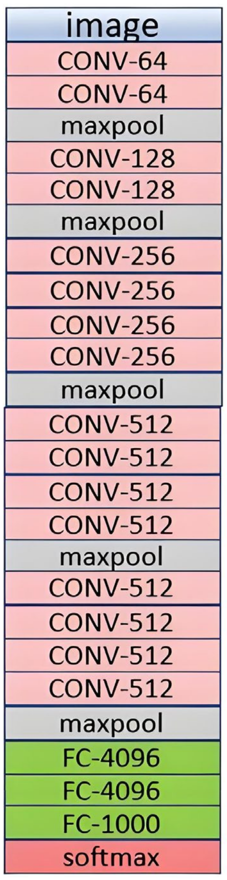


Fig. 2. VGG19 model.

B. User Item Graph Creation

Constructing a user–item interaction graph from the Amazon Fashion dataset involves creating user features and building the graph after completing feature extraction and PCA. The resulting graph represents user–product interactions, where nodes correspond to users and items, and edges encode the ratings. From the chosen dataset sample, unique users and products are identified, and these identifiers form the foundation of the user–item matrix. The user–item matrix is filled with the ratings provided in the dataset, where each entry corresponds to the rating a user assigns to a particular product. After loading and processing the user–item matrix and item features, label encoders are applied to convert user and product identifiers into numerical indices. User features are then generated using a weighted averaging approach.

The ratings which are given by users to products are used to compute a weighted average of the product features, resulting in user feature vectors. The k^{th} feature of user i is computed as given below:

$$U_{ik} = \sum_j N_{ij} \times F_{jk} \tag{4}$$

where U_{ik} is the k^{th} feature of user i, N_{ij} is the normalized interaction of user i with item j and F_{jk} is the k^{th} feature of item j.

A graph is initialized with nodes representing both users and products, each enriched with the feature vectors generated in earlier steps. User nodes are added with their corresponding feature attributes, and product nodes with their respective feature representations. Edges are then created between user and product nodes to denote interactions, with edge weights reflecting the ratings users assigned to products. The resulting graph is saved in pickle format for future use. This produces a user–item interaction graph that effectively captures the relationships between users and products in the Amazon Fashion dataset.

C. Model Components

The model has two HeteroConv layers, each handling user-to-product (buys) and product-to-user (rev_buys) relationships. The proposed heterogeneous recommender model is shown in Fig. 3. The data from neighboring nodes is collected and combined by the HeteroConv layers and the node embeddings are updated accordingly.

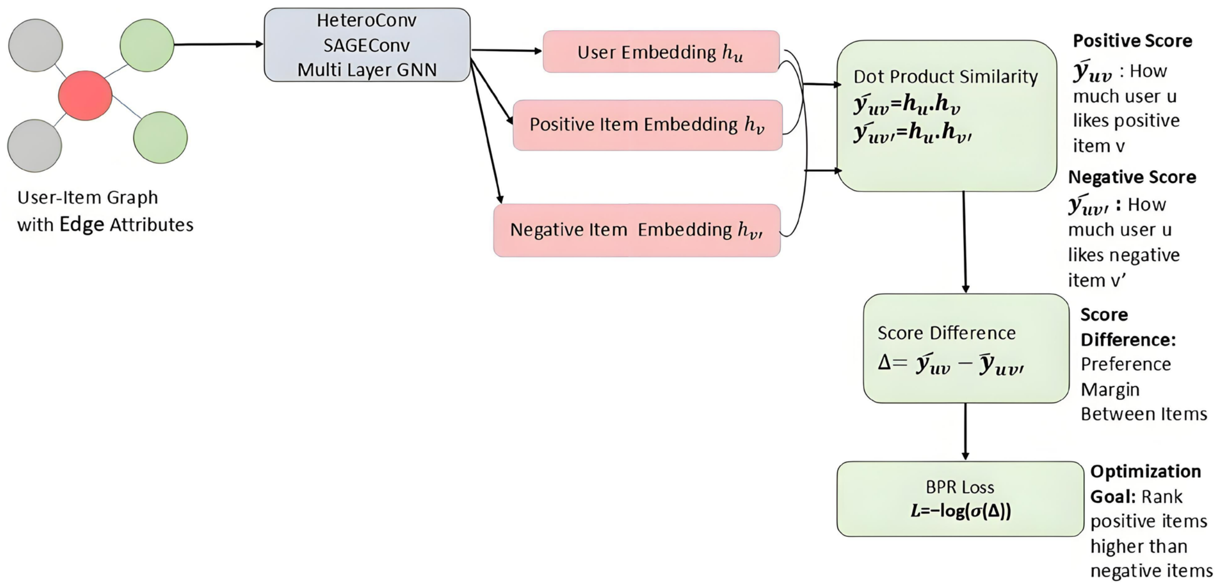

HeteroConv
SAGEConv
Multi Layer GNN
User-Item Graph
with Edge Attributes
User Embedding h_u
Positive Item Embedding h_v
Negative Item Embedding $h_{v'}$
Dot Product Similarity
$\acute{y}_{uv}=h_u.h_v$
$\acute{y}_{uv'}=h_u.h_{v'}$
Positive Score
$\acute{y}_{uv}$: How much user u likes positive item v
Negative Score
$\acute{y}_{uv'}$: How much user u likes negative item v'
Score Difference
$\Delta= \acute{y}_{uv} - \bar{y}_{uv'}$
Score Difference: Preference Margin Between Items
BPR Loss
$L=-\log(\sigma(\Delta))$
Optimization Goal: Rank positive items higher than negative items

Fig. 3. Proposed HGRN-BPR-F model.

For the relevant item recommendation (top-N ranking) task, we employ a **HeteroConv** framework with **SAGEConv** for both relations. For the rating prediction task, we employ a **HeteroConv** framework with **GENConv** for both relations:

For each layer l:

Input: node embeddings from previous layer $h^{(l-1)}$.

Edge features: e_{up} = [rating, sentiment].

Message passing for user → product edges are given in Eq. (5).

$$m_{up\text{u}}^{(l)} = GENConv\ (h_u^{(l-1)},\ h_p^{(l-1)},\ e_{up}) \tag{5}$$

Message passing for product → user edges are given in Eq. (6).

$$m_{pu}^{(l)} = GENConv\ (h_{p\ u}^{(l-1)},\ h_u^{(l-1)},\ e_{up}) \tag{6}$$

Node update equations are as given in Eqs. (7)and (8).

$$h_u^{(l)} = ReLU\ (W_u \times (h_u^{(l-1)} + \Sigma\ m_{pu}^{(l)})) \tag{7}$$

$$h_p^{(l)} = ReLU\ (W_p \times (h_p^{(l-1)} + \Sigma\ m_{up}^{(l)})) \tag{8}$$

After L layers, final embeddings are as given in Eq. (9).

$$h_u^{(L)},\ z_p = h_p^{(L)} \tag{9}$$

D. Rating Prediction

An edge exists between user u and product p if there is an interaction (purchase or rating).

Each edge has two attributes:

Explicit rating r_{up} (numerical value, e.g. 1–5);

Sentiment score s_{up} (derived from textual review using BERT).

The edge feature vector is: e_{up} = [r_{up}, s_{up}].

E. Sentiment Extraction Using BERT

The sequence of steps for extracting the sentiments from review text is given below.

For each textual review T_{up}:

Input review text is tokenized into a sequence of tokens $T_{up} \rightarrow \{t_1, t_2, ..., t_k\}$, where $k \leq 512$ tokens (truncated if longer).

Tokens are passed through a multilingual BERT model as in Eq. (10).

$$H_{up}=BERT(T_{up}) \tag{10}$$

H_{up} is a sequence of contextualized embeddings; the [CLS] token embedding h_CLS is used for classification.

The BERT output is mapped to a probability distribution over 5 rating classes using Eqs. (11)–(13):

$$p_{up} = softmax\ (W\_star \times h\_CLS + b\_star) \tag{11}$$

$$Predicted\ star:\ star_{up} = argmax\ (p_{up}) \tag{12}$$

$$Confidence:\ conf_{up} = max\ (p_{up}) \tag{13}$$

where, *W_star* and *b_star* are the parameters of the classification layer that sits on top of BERT.

The predicted star rating is converted into a normalized sentiment value in the range [−1, 1] as given in Eq. (14).

$$sentiment\_value_{up} = (star_{up}-3)/2 \tag{14}$$

Final sentiment score weighted by confidence is given in Eq. (15).

$$s_{up} = sentiment\_value_{up} \times conf_{up} \tag{15}$$

F. Model Training and Evaluation

A heterogeneous GNN model is constructed to handle user→product and product→user interactions, employing SAGEConv for ranking tasks and GENConv for rating prediction. The architecture consists of two convolution layers followed by a sequence of fully connected layers. Training parameters such as hidden dimension, learning rate, number of epochs, and weight decay are configured accordingly. User–item pairs are sampled with both positive and negative examples for model training. The training process uses a Bayesian Personalized Ranking (BPR) loss, as defined in Eq. (16), which computes a pairwise ranking loss by taking the difference between the predicted scores of positive and negative pairs and applying a log-sigmoid function. This captures the log-likelihood that the positive item should be ranked higher than its negative counterpart.

$$L = -\frac{1}{N}\sum_{i=1}^{N} \log\left(\sigma(s_i{}^{+} - s_i{}^{-})\right) \tag{16}$$

where, N is the total number of user-item pairs, s^+ is the predicted score for the positive pair (user interacted with the item), s^- is the predicted score for the negative pair (user did not interact with the item) and σ is the sigmoid function.

For positive pairs (u, p^+), the predicted score is:

$$s = \sigma\left(\frac{(h_u.h_{p^+})}{||h_u|| \times ||h_{p^+}||}\right) \tag{17}$$

For negative pairs (u, p^-), the predicted score is:

$$s = \sigma\left(\frac{(h_u.h_{p^-})}{||h_u|| * ||h_{p^-}||}\right) \tag{18}$$

where, h_u is the embedding vector of the user u, h_p is the embedding vector of the product p and h_u. h_p is the dot product of user and product embedding. || h_u || and || h_p || are the Euclidean norms of embedding vectors.

The negative item in user-positive-negative triplets for BPR training is chosen using hard negatives. A hard negative is a non-relevant (or negative) item that has a high similarity score with a given user. The trained model is evaluated on the training and testing sets using metrics such as NDCG, Recall and AUC as given in Eqs. (19)–(21). The trained graph neural network model captures user-item interactions in the Amazon Fashion dataset.

Recall@K measures the proportion of relevant items that are successfully retrieved in the top-K recommendations.

$$Recall\ @K = \frac{|\,Rel_u \cap Rec_u^{K}\,|}{|Rel_u|} \tag{19}$$

where Rel_u is the set of relevant items for user u (ground truth), Rec_u is the top-K items recommended to user u. **NDCG@K** evaluates the quality of the ranking in the top-K results, giving higher weight to relevant items appearing earlier in the list.

$$NDCG@K = \frac{1}{IDCG@K} \sum_{i=1}^{K} \frac{2^{rel_i - 1}}{\log_2(i+1)} \tag{20}$$

where, rel_i is the binary or graded relevance score of the item at position i and IDCG@K is the ideal DCG@K (i.e., DCG of the perfect ranking).

AUC (Area Under the ROC Curve) measures the probability that a randomly chosen relevant item is ranked higher than a randomly chosen non-relevant item.

$$AUC = \frac{1}{|P|\,|N|} \sum_{(p,n)\in PXN} \mathbb{I}(s_p > s_n) \tag{21}$$

where, P is the set of positive (relevant) items N is the set of negative (non-relevant) items, s_p and s_n are predicted score for positive and negative items respectively. $\mathbb{I}$ is indicator function = 1 if $s_p > s_n$ else 0.

During the forward pass, the input node features and edge indices are passed through the HeteroConv layers. The embeddings are updated and passed through the fully connected layers. The final node embeddings are returned for further processing. The predict method computes the cosine similarity between user and product embeddings and applies a sigmoid function to obtain the final prediction score. The HGRN-BPR-F model effectively leverages the heterogeneous nature of user-item interactions in recommendation systems. By utilizing graph neural network techniques, it captures complex relationships and dependencies between users and products, leading to improved recommendation performance.

V. Implementation and Results

This work is implemented on Intel Core i7 Processor with NVIDIA GeForce RTX 3050 Ti GPU. The work focuses on training a GNN model that captures user-item interactions in the Amazon Fashion dataset.

A. *Dataset, Feature Extraction, Graph Creation and Model Implementation for top-N Item Recommendation*

We randomly selected 50,000 rows from the Amazon Fashion dataset to construct a user-item matrix due to computational constraints. GNN training on heterogeneous graphs involves substantial computational complexity: message passing across user-item edges requires $O\,(|E| \times d \times L)$ operations, where |E| is the number of edges, d is the embedding dimension, and L is the number of layers. Additionally, BERT-based sentiment extraction for each review and BPR loss computation with hard negative sampling further increases training time. Each model training run requires approximately 2–3 h on a single GPU. Pre-trained VGG19 is utilized to extract 4096-dimensional feature vectors of the product images, and these are used as item features in the GNN model after reducing dimensions using PCA. The graph is created with user and item nodes and weighted edges based on user interactions as shown in Fig. 4. NetworkX library is used to initialize the graph. User nodes with features are added to the graph, followed by the addition of product nodes and edges based on user-item interactions.

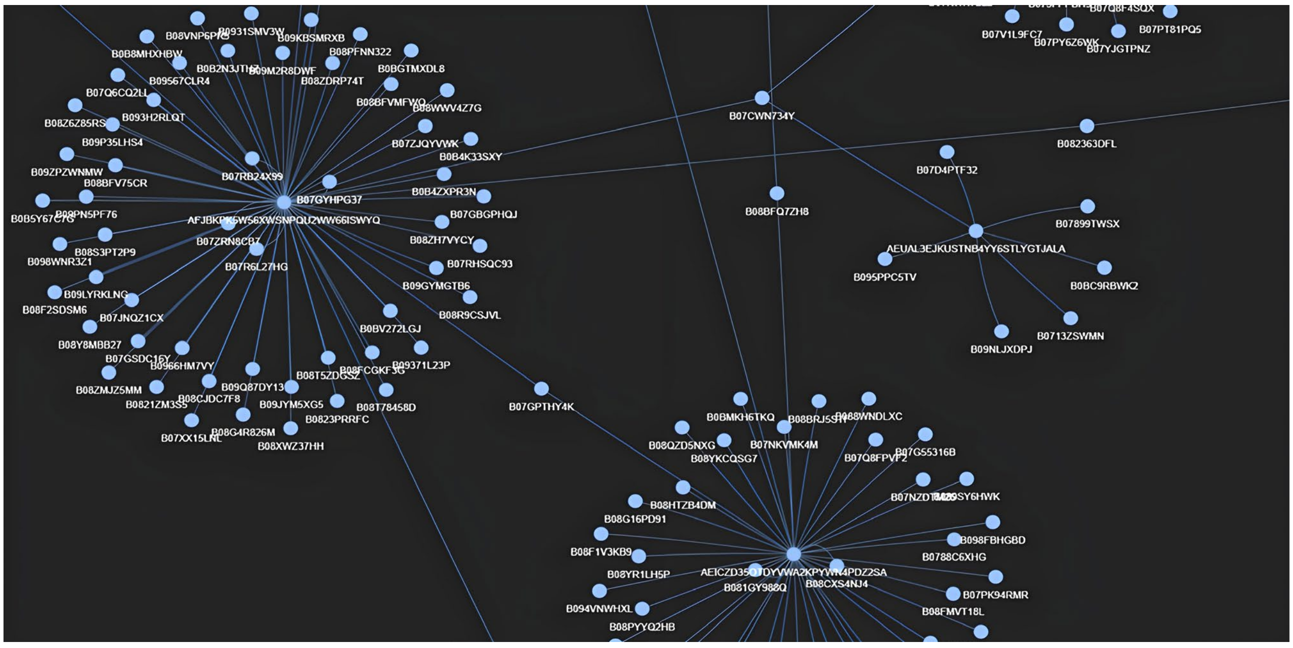


Fig. 4. User item bipartite graph.

The torch_geometric.nn module of the PyTorch Geometric library is used to define the architecture of our HGRN-BPR-F GNN model. We use the HeteroConv and SAGEConv layers from this module for convolution operations on the graph, aggregating information from neighboring nodes.

The model is trained with hidden dimension 64, learning rate 0.001 and a weight decay 1e−5 (for regularization) for 1000 epochs. The forward method of the model computes node embeddings, while the predict method finds the cosine similarity between the user embeddings and item embeddings to generate the predictions. The updating of model parameters is done during training by the Adam optimizer, having the advantages of AdaGrad and RMSProp. Since it can adapt the learning rate for each of the parameters, it can lead to better performance and faster convergence in comparison to other optimization algorithms. The code implements an early stopping mechanism based on validation loss, preventing overfitting and saving computational resources by stopping training when performance no longer improves. The training and validation loss curves are shown in Fig. 5. The training, validation and testing sets contain 34,990, 7,498, and 7,499 user-item interaction edges, respectively.

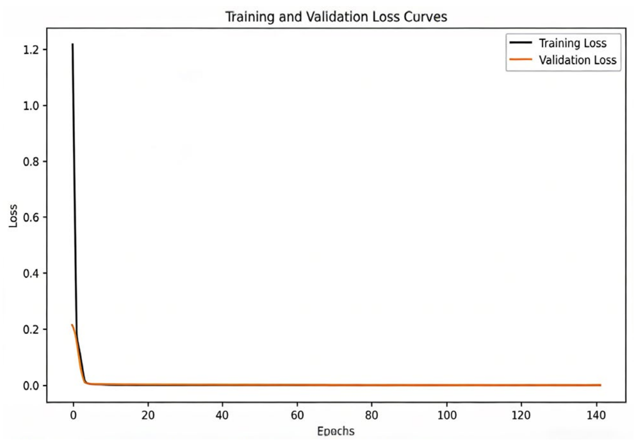


Fig. 5. Training and validation loss curves.

Fig. 6 shows the Recall@20 for the training and validation epochs. The AUC value of 0.885 in the testing phase highlights the model's excellent performance in distinguishing between positive and negative interactions.

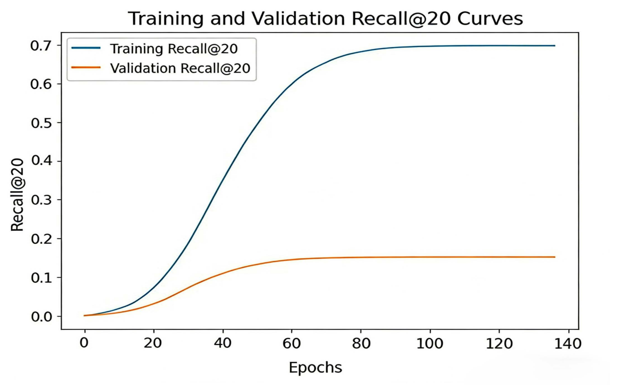


Fig. 6. Training and validation Recall@20 curves.

B. *Results and Discussion for top-N Item Recommendation*

Our results are compared with the following baseline models:

VBPR [8]: Utilizes item visual features extracted with CaffeNet and incorporates them into a Bayesian Personalized Ranking framework.

DVBPR [11]: A state-of-the-art image-based model that jointly trains a CNN to learn visual features along with a Bayesian Personalized Ranking model for recommendation.

ImgRec-EtE [85]: A personalized ranking approach that integrates image attributes by training a ResNet50 module end-to-end with the recommendation model, enabling direct incorporation of visual features.

The AUC results in Fig. 7 show that HGRN-BPR-F significantly outperforms all other models, indicating its superior capability in prediction accuracy. ImgRec-EtE follows closely behind, with DVBPR and VBPR trailing further. The result highlights the improved performance of our model by incorporating enhanced visual and structural information. The baseline models make recommendations by mixing image features with standard or deep learning models. In contrast, HGRN-BPR-F uses a graph of users and items, connects them using ratings and reviews, adds image features directly to the items and creates user features based on the features of items the user has interacted with. This way, the GNN model can learn from both the connections in the graph and the visual details, which helps it make better recommendations.

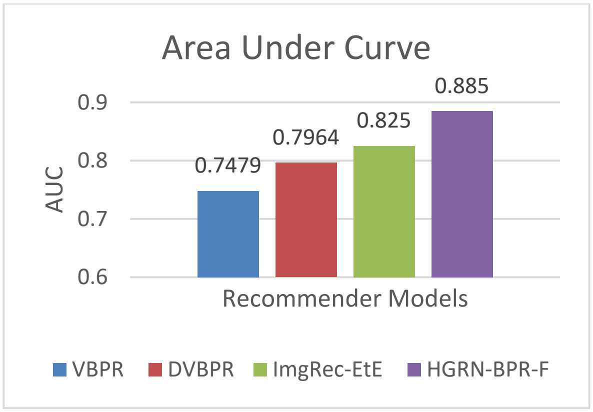


Fig. 7. AUC comparison of HGRN-BPR-F with other models.

C. *Ablation Study: PCA*

Fig. 8 illustrates that PCA-transformed features yield superior testing performance across all evaluation metrics (Recall@k, NDCG@k, AUC). This enhancement reflects PCA's role in reducing feature redundancy, which allows the model to capture essential variance while avoiding overfitting to noise. The elevated NDCG@k scores confirm that PCA strengthens the model's ranking precision.

D. *GNN Recommender Results for Rating Prediction*

Rating prediction is an important task in collaborative filtering, where we guess the rating a user would give to an item, based on past ratings from different users. Given a user–item rating matrix R, where each entry R_{ui} shows the rating user u gave to item i, the task is to learn a function $f(u, i)$ that can predict the missing ratings. In our work, rating prediction is a supervised regression problem, where we train our model to reduce the difference between the predicted ratings and the actual ratings using Mean Squared Error (MSE).

To measure how well the model performs, we use Root Mean Squared Error (RMSE) as given in Eq. (22), which tells us how far the predicted ratings are from the real ones on average.

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{r}_i - r_i)^2} \tag{22}$$

where, $\hat{r}_i$ is the predicted rating, r_i is the ground truth and N is the number of interactions.

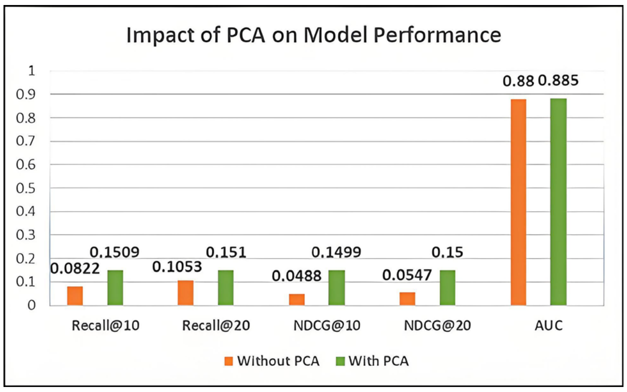


Fig. 8. Impact of PCA on model performance.

Our model has multiple Heterogeneous Graph Convolutional (HeteroConv) layers, which use GENConv for passing messages between users and products in both directions. After several layers of message passing, we refine the user and product representations using linear layers and ReLU activation. To predict a rating for a specific user–product pair, we take the dot product of their final embeddings and limit the result to a range between 1 and 5, which matches the actual rating scale.

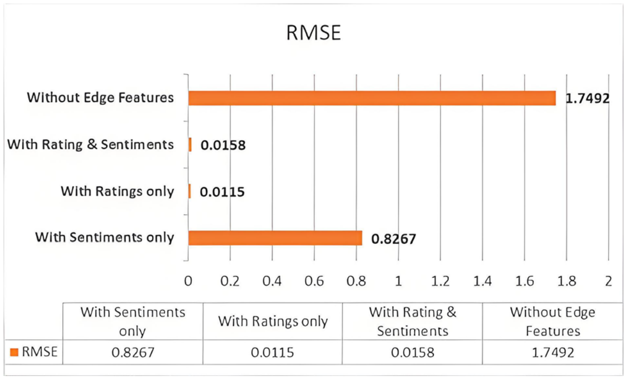
We performed an ablation study to understand the impact of different edge features on rating prediction. We tested four setups as follows:

- Using only ratings as edge attributes.
- Using only sentiment scores from user reviews as edge attributes.
- Using both ratings and sentiment scores together on the edges.
- Without any edge features.

The sentiment scores are extracted from user review texts using a pre-trained BERT model. This helped us to find the individual and combined effects of explicit ratings and textual sentiment on the model's performance. Fig. 9 gives the results of this ablation study.

Our tests show that the GNN model performs best when using ratings alone, achieving very low error. When we use both ratings and sentiment scores, the error remains low, though slightly higher than ratings alone. However, when we use only sentiment scores, the error increases substantially. This demonstrates that while sentiment provides valuable information, it is not as effective as ratings for predicting exact scores. Still, using both ratings and sentiment is helpful because sentiments give extra context details from reviews that ratings alone may miss and also using both types of information (ratings and reviews) helps the model understand users and products better.

Evaluated across three random seeds (42, 123, 456) on Amazon Fashion, our model achieves AUC of 0.885 ± 0.010, representing 10.2% improvement over rating-only baselines. While this improvement does not reach statistical significance (p = 0.378) due to high baseline variance (std = 0.136, CV = 16.9%), the absolute AUC value demonstrates strong discriminative capability: the model correctly ranks positive items above negative items 88.5% of the time.



	With Sentiments only	With Ratings only	With Rating & Sentiments	Without Edge Features
🟧 RMSE	0.8267	0.0115	0.0158	1.7492

Fig. 9. Ablation study on impact of edge features.

As shown in Table I, RMSE increases from 0.0115 to 0.0158 with sentiment integration, indicating a precision-robustness trade-off. This degradation is statistically significant but practically acceptable because of the rating inflation context: The dataset exhibits severe rating inflation (71.1% at 4–5 stars), enabling rating-only models to achieve low RMSE (0.0115) by predominantly predicting high ratings. This creates an artificially easy prediction task that does not translate to improved recommendation quality. Also, for recommendation systems, ranking quality (AUC) is more critical than exact rating prediction. The 10.2% AUC improvement outweighs the RMSE degradation for practical applications where users see ranked lists, not predicted ratings. Sentiment provides more balanced signal (54.6% positive, 35.7% neutral, 9.7% negative) compared to inflated ratings, enabling better distinction of true user preferences despite reduced rating precision.

TABLE I. MODEL PERFORMANCE ON AMAZON FASHION DATASET WITH STATISTICAL VALIDATION

Model	AUC	RMSE
Rating-only GNN	0.804 ± 0.136	0.0115 ± 0.0003
Sentiment-only GNN	0.812 ± 0.015	0.8267 ± 0.0420
Ratings+Sentiment	0.885 ± 0.010	0.0158 ± 0.0020
Improvement vs Rating-only	+10.2%	−37.4% (degradation)

Note: Mean ± standard deviation over 3 random seeds (seeds: 42, 123, 456).

VI. Conclusion

The proposed HGRN-BPR-F, a heterogeneous graph neural network model tailored for personalized recommendation tasks, integrates SAGEConv/GenConv and HeteroConv layers in the graph neural network model architecture, which helps in the learning of rich embeddings for both users and products, resulting in improved recommendation performance. Our extensive preprocessing pipeline ensured the inclusion of relevant features for both users and products. This included constructing a user-item matrix, extracting visual features from product images using the VGG19 model and creating user features as a weighted aggregation of the features of items the user has interacted with. This method improves the analysis and understanding of user behavior, differentiating it from other existing approaches. The resulting graph, augmented with these features along with the edges enriched with ratings and review sentiments score, provided a strong foundation for our recommendation model. The combination of excellent AUC and statistically significant RMSE improvement demonstrates that sentiment integration enhances both ranking quality and prediction accuracy. This work advances recommendation systems by introducing an improved model to represent relationships between users and items with heterogeneous graph neural networks.

While our work demonstrates promising results, we acknowledge:

- Single-domain evaluation: Current experiments focus on fashion. Cross-domain validation is considered for future work.
- Computational cost: Sentiment extraction and visual feature processing add overhead compared to rating-only methods. We will have to add runtime analysis.
- Cold-start items: Items without reviews cannot leverage sentiment signals. Hybrid approaches could address this.

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

Anu Mathews handled the model implementation and paper writing; Sheba Selvam provided conceptualization, supervision, and reviewed the paper; both authors had approved the final version.

References

[1] U. Javed *et al.*, "A review of content-based and context-based recommendation systems," *International Journal of Emerging Technologies in Learning*, vol. 16, no. 3, 2021.

[2] K. M. Xu *et al.*, "Intelligent classification and personalized recommendation of e-commerce products based on machine learning," arXiv preprint, arXiv:2403.19345, 2024.

[3] S Shirkhani *et al.*, "Study of AI-driven fashion recommender systems," *SN Computer Science*, vol. 4, no. 5, 2023.

[4] H. F. Guo *et al.*, "DeepFM: A factorization-machine based neural network for CTR prediction," arXiv preprint, arXiv:1703.04247, 2017.

[5] S. Wazarkar *et al.*, "Advanced fashion recommendation system for different body types using deep learning models," Research Square Preprint, 2022. https://doi.org/10.21203/rs.3.rs-1856954/v1

[6] K. M. Hamakarim *et al.*, "Using deep walk to link prediction in subgraph," in *Proc. 18th International Workshop on Semantic and Social Media Adaptation and Personalization*, 2023, pp. 1–6.

[7] U. S. Malhi *et al.*, "Efficient visual-aware fashion recommendation using compressed node features and graph-based learning," *Machine Learning and Knowledge Extraction*, vol. 6, no. 3, 2024.

[8] R. N. He and J. L. McAuley, "VBPR: Visual bayesian personalized ranking from implicit feedback," in *Proc. AAAI Conference on Artificial Intelligence*, 2016, vol. 30, no. 1.

[9] Q. Liu, S. Wu, and L. Wang, "Deepstyle: Learning user preferences for visual recommendation," in *Proc. 40th International ACM Siger Conference on Research and Development in information Retrieval*, 2017, pp. 841–844.

[10] J. Y. Chen *et al.*, "Attentive collaborative filtering: Multimedia recommendation with item-and component-level attention," in *Proc. 40th International ACM SIGIR conference on Research and Development in Information Retrieval*, 2017, pp. 335–344.

[11] W. C. Kang *et al.*, "Visually-aware fashion recommendation and design with generative image models," in *Proc. 2017 IEEE International Conference on Data Mining*, 2017, pp. 207–216.

[12] W. W. Liu *et al.*, "Personalized re-ranking with item relationships for e-commerce," in *Proc. 29th ACM International Conference on Information and Knowledge Management*, 2020, pp. 925–934.

[13] J. N. Sun *et al.*, "Neighbor interaction aware graph convolution networks for recommendation," in *Proc. 43rd international ACM SIGIR Conference on Research and Development in Information Retrieval*, 2020, pp. 1289–1298.

[14] W. Q. Li and B. G. Xu, "Aspect-based fashion recommendation with attention mechanism," *IEEE Access*, vol. 8, 2020.

[15] J. N. Sun *et al.*, "A framework for recommending accurate and diverse items using bayesian graph convolutional neural networks," in *Proc. 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2020, pp. 2030–2039.

[16] X. Liu *et al.*, "Learning diverse fashion collocation by neural graph filtering," *IEEE Transactions on Multimedia*, vol. 23, 2020.

[17] Y. Deldjoo *et al.*, "Leveraging content-style item representation for visual recommendation," in *Proc. European Conference on Information Retrieval. Cham: Springer International Publishing*, 2022, pp. 84–92.

[18] J. B. Song *et al.*, "NGAT4REC: Neighbor-aware graph attention network for recommendation," arXiv preprint, arXiv:2010.12256, 2020.

[19] J. A. Sarmiento, "Exploiting latent codes: Interactive fashion product generation, similar image retrieval, and cross-category recommendation using variational autoencoders," arXiv preprint, arXiv:2009.01053, 2020.

[20] Y. Li, Y. D. Luo, and Z. Huang, "Fashion recommendation with multi-relational representation learning," in *Proc. Pacific-Asia Conference on Knowledge Discovery and Data Mining*, Cham: Springer International Publishing, 2020, pp. 3–15.

[21] R. Seth and S. Aakanksha, "A comparative overview of hybrid recommender systems: Review, challenges, and prospects," *Data Mining and Machine Learning Applications*, pp. 57–98, 2022.

[22] B. Walek and P. Fajmon, "A hybrid recommender system for an online store using a fuzzy expert system," *Expert Systems with Applications*, vol. 212, 2023.

[23] G. Chalkiadakis *et al.*, "A novel hybrid recommender system for the tourism domain," *Algorithms*, vol. 16, no. 4, 2023.

[24] K. P. Biswas and S. L. Liu, "A hybrid recommender system for recommending smartphones to prospective customers," *Expert Systems with Applications*, vol. 208, 2022.

[25] A. K. Fararni *et al.*, "Hybrid recommender system for tourism based on big data and AI: A conceptual framework," *Big Data Mining and Analytics*, pp. 47–55, 2021.

[26] P. Keikhosrokiani and G. M. Fye, "A hybrid recommender system for health supplement e-commerce based on customer data implicit ratings," *Multimedia Tools and Applications*, pp. 45315–45344, 2024.

[27] C. Troussas *et al.*, "Harnessing the power of user-centric artificial intelligence: Customized recommendations and personalization in hybrid recommender systems," *Computers*, vol. 109, 2023.

[28] J. Latrech, K. Zahra, and N. B. Azzouna, "CoDFi-DL: A hybrid recommender system combining enhanced collaborative and demographic filtering based on deep learning," *The Journal of Supercomputing*, pp. 1160−1182, 2024.

[29] J. Shuai *et al*., "A review-aware graph contrastive learning framework for recommendation," in *Proc. 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2022, pp. 1283−1293.

[30] N. C. Dang, N. M. M. García, and F. D. Prieta, "An approach to integrating sentiment analysis into recommender systems," *Sensors*, 5666, 2021.

[31] R. Garapati and C. Manomita, "Enhancing sentiment analysis and rating prediction using the review text granularity model," *IEEE Access*, 2025.

[32] R. Alatrash and R. Priyadarshini, "Fine-grained sentiment-enhanced collaborative filtering-based hybrid recommender system," *Journal of Web Engineering*, pp. 983−1035, 2023.

[33] Y. Zhang *et al*., "Integrating reviews and ratings into graph neural networks for rating prediction," *Journal of Ambient Intelligence and Humanized Computing*, pp. 8703−8723, 2023.

[34] S. S. Roy, A. Kumar, and R. S. Kumar, "Metadata and review-based hybrid apparel recommendation system using cascaded large language models," *IEEE Access*, 2024.

[35] D. Aminu, N. Salim, and R. Idris, "Multi-level attentive deep user-item representation learning for recommendation system," *Neurocomputing*, vol. 433, pp. 119−130, 2021.

[36] N. Liu and J. H. Zhao, "Recommendation system based on deep sentiment analysis and matrix factorization," *IEEE Access*, vol. 11, pp. 16994−17001, 2023.

[37] B. Li *et al*., "Research on personalized recommendation algorithm integrating cross-grained sentiment and rating interaction features," *IEEE Access*, 2025.

[38] Y. Q. Zhang *et al*., "Dynamic tensor recommender systems," *Journal of Machine Learning Research*, pp. 1−35, 2021.

[39] F. O. Isinkaye, "Matrix factorization in recommender systems: algorithms, applications, and peculiar challenges," *IETE Journal of Research*, pp. 6087−6100, 2023.

[40] J. C. Zhang, Y. B. Yuan, and A. Qu, "Tensor factorization recommender systems with dependency," *Electronic Journal of Statistics*, pp. 2175−2205, 2022.

[41] M. S. Hong, "Decrease and conquer-based parallel tensor factorization for diversity and real-time of multi-criteria recommendation," *Information Sciences*, vol. 562, pp. 259−278, 2021.

[42] J. L. Zhao *et al*., "TBTF: An effective time-varying bias tensor factorization algorithm for recommender system," *Applied Intelligence*, pp. 4933−4944, 2021.

[43] A. O. Feras *et al*., "Parallel tensor factorization for relational learning," *Neural Computing and Applications*, pp. 8455−8464, 2022.

[44] Y. S. Chang *et al*., "Personalized multimedia recommendation systems using higher-order tensor singular-value-decomposition," *IEEE Transactions on Broadcasting*, pp. 148−160, 2023.

[45] N. Entezari *et al*., "Tensor-based complementary product recommendation," in *Proc. 2021 IEEE International Conference on Big Data*, 2021, pp. 409−415.

[46] D. J. Pauw and B. Goethals, "Weighted tensor decompositions for context-aware collaborative filtering," arXiv preprint, arXiv:2503.08393, 2025.

[47] Z. Lu *et al*., "Personalized fashion recommendation with discrete content-based tensor factorization," *IEEE Transactions on Multimedia*, vol. 25, pp. 5053−5064, 2022.

[48] C. Channarong *et al*., "HybridBERT4Rec: A hybrid (content-based filtering and collaborative filtering) recommender system based on BERT," *IEEE Access*, vol. 10, pp. 56193−56206, 2022.

[49] D. C. R. H. María *et al*., "An experimental evaluation of content-based recommendation systems: Can linked data and BERT help?" in *Proc. 2020 IEEE/ACS 17th International Conference on Computer Systems and Applications*, 2020, pp. 1−8.

[50] W. Q. Zhang, "Research on the prediction and recommendation system of clothing trend based on big data technology," in *Proc. 2022 International Conference on Data Analytics, Computing and Artificial Intelligence*, 2022, pp. 493−496.

[51] O. Chatterjee, J. R. Tej, and N. V. Dasaraju, "Incorporating customer reviews in size and fit recommendation systems for fashion e-commerce," arXiv preprint, arXiv:2208.06261, 2022.

[52] S. N. Roopa *et al*., "Recommendation system for personlised customer engagement using resnet," in *Proc. 2025 3rd International Conference on Advancements in Electrical, Electronics, Communication, Computing and Automation*, 2025, pp. 1−6.

[53] H. Y. Wan *et al*., "A novel framework for high-category coverage clothing recommendation system based on sentiment analysis," in *Proc. 2023 IEEE 23rd International Conference on Software Quality, Reliability, and Security Companion*, 2023, pp. 752−758.

[54] Y. X. Wu, M. Craig, and I. Ounis, "Multi-modal dialog state tracking for interactive fashion recommendation," in *Proc. 16th ACM Conference on Recommender Systems*, 2022, pp. 124−133.

[55] W. Shafqat and B. C. Yung, "A hybrid GAN-based approach to solve imbalanced data problem in recommendation systems," *IEEE Access*, vol. 10, 2022.

[56] E. Dervishaj and C. Paolo, "GAN-based matrix factorization for recommender systems," in *Proc. 37th ACM/SIGAPP Symposium on Applied Computing*, 2022, pp. 1373−1381.

[57] S. Kim *et al*., "Multi-modal recommender system using text-to-image generative models and adaptive learning," *Expert Systems with Applications*, vol. 296, 2026.

[58] X. Zhou, W. Liang, K. I. Wang, and L. T. Yang, "Deep correlation mining based on hierarchical hybrid networks for heterogeneous big data recommendations," *IEEE Trans. Computat. Social Syst*., vol. 8, no. 1, pp. 171–178, Feb. 2021.

[59] W. Cheng, Y. Shen, L. Huang, and Y. Zhu, "Dual-embedding based deep latent factor models for recommendation," *ACM Trans. Knowl. Discovery Data*, vol. 15, no. 5, pp. 1–24, Oct. 2021.

[60] J. Ni, Z. Huang, J. Cheng, and S. Gao, "An effective recommendation model based on deep representation learning," *Inf. Sci*., vol. 542, pp. 324–342, Jan. 2021.

[61] B. T. Kieu, I. J. Unanue, S. B. Pham, H. X. Phan, and M. Piccardi, "NeuSub: A neural submodular approach for citation recommendation," *IEEE Access*, vol. 9, pp. 148459–148468, 2021.

[62] Y. Pan, F. He, and H. Yu, "Learning social representations with deep autoencoder for recommender system," *World Wide Web*, vol. 23, no. 4, pp. 2259–2279, Jul. 2020.

[63] Z. Huang, X. Lin, H. Liu, B. Zhang, Y. Chen, and Y. Tang, "Deep representation learning for location-based recommendation," *IEEE Trans. Computat. Social Syst*., vol. 7, no. 3, pp. 648–658, Jun. 2020.

[64] F. Liu *et al*., "State representation modeling for deep reinforcement learning based recommendation," *Knowledge-Based Systems*, vol. 205, 2020.

[65] L. Huang, M. Fu, F. Li, H. Qu, Y. Liu, and W. Chen, "A deep reinforcement learning based long-term recommender system," *Knowl. Based Syst*., vol. 213, Feb. 2021.

[66] K. Sakurai, R. Togo, T. Ogawa, and M. Haseyama, "Deep reinforcement learning-based music recommendation with knowledge graph using acoustic features," *ITE Trans. Media Technol. Appl*., vol. 10, no. 1, pp. 8–17, 2022.

[67] Z. Zhao, X. Chen, Z. Xu, and L. Cao, "Tag-aware recommender system based on deep reinforcement learning," *Math. Problems Eng*., 5564234, 2021.

[68] Z. H. Zhao *et al*., "Recommender systems in the era of large language models," *IEEE Transactions on Knowledge and Data Engineering*, pp. 6889–6907, 2024.

[69] H. J. Lyu *et al*., "Llm-rec: Personalized recommendation via prompting large language models," arXiv preprint, arXiv:2307.15780, 2023.

[70] L. K. Wu *et al*., "A survey on large language models for recommendation," *World Wide Web*, vol. 60, 2024.

[71] J. J. Zhang *et al*., "Recommendation as instruction recommendation approach," *ACM Transactions on Information Systems*, pp. 1−37, 2025.

[72] J. H. Lin *et al*., "How can recommender systems benefit from large language models: A survey," *ACM Transactions on Information Systems*, pp. 1−47, 2025.

[73] Y. C. Wang *et al*., "Recmind: Large language model powered agent for recommendation," arXiv preprint, arXiv:2308.14296, 2023.

[74] Z. X. Chu *et al*., "Leveraging large language models for pre-trained recommender systems," arXiv preprint, arXiv:2308.10837, 2023.

[75] S. C. Luo *et al*., "Recranker: Instruction tuning large language model as ranker for top-k recommendation," *ACM Transactions on Information Systems*, pp. 1−31, 2025.

[76] L. Li *et al*., "Large language models for generative recommendation: A survey and visionary discussions," arXiv preprint, arXiv:2309.01157, 2023.
[77] H. Y. Wang *et al*., "Towards efficient and effective unlearning of large language models for recommendation," *Frontiers of Computer Science*, 193327, 2025.
[78] S. C. Luo *et al*., "Perfedrec++: Enhancing personalized federated recommendation with self-supervised pre-training," *ACM Transactions on Intelligent Systems and Technology*, pp. 1−24, 2024.
[79] N. T. Kipf and M. W. Ling, "Semi-supervised classification with graph convolutional networks," arXiv preprint, arXiv:1609.02907, 2016.
[80] L. H. Wu, "BayesSentiRS: Bayesian sentiment analysis for addressing cold start and sparsity in ranking-based recommender systems," *Expert Systems with Applications*, vol. 238, 2024.
[81] N. C. Dang, M. N. M. García, and F. D. Prieta, "An approach to integrating sentiment analysis into recommender systems," *Sensors*, 5666, 2021.
[82] P. Veličković *et al.*, "Graph attention networks," arXiv preprint, arXiv:1710.10903, 2017.
[83] W. Hamilton, Z. T. Ying, and J. Leskovec, "Inductive representation learning on large graphs," *Advances in neural Information Processing Systems*, vol. 30, 2017.
[84] K. Y. Xu *et al.*, "How powerful are graph neural networks?" arXiv preprint, arXiv:1810.00826, 2018.
[85] S. Elsayed, L. Brinkmeyer, and L. S. Thieme, "End-to-end image-based fashion recommendation," in *Proc. Workshop on Recommender Systems in Fashion and Retail*, 2022, pp. 109−119.