An Explainable Recommender System by Integrating Graph Neural Networks and User Reviews

Sahar Batmani
Dept. of Computer Engineering
University of Kurdistan
Sanandaj, Iran
s.batmani@uok.ac.ir

Parham Moradi (*)
School of Engineering
RMIT University
Melbourne, Australia
parham.moradi@rmit.edu.au

Narges Heidari
Dept. of Computer Engineering
Islamic Azad University, Ilam
Branch, Ilam, Iran
n.heidari@iau.ac.ir

Mahdi Jalili
School of Engineering
RMIT University
Melbourne, Australia
mahdi.jalili@rmit.edu.au

***Abstract*—This paper introduces an explainable Graph Neural Network (GNN)-based recommender system that integrates user-item interactions and user reviews to enhance recommendation accuracy and interpretability. The proposed method leverages Temporal Convolutional Networks (TCNs) as a language model to encode user reviews into vector representations, capturing temporal dynamics and contextual information. Additionally, it extracts opinion-aspect pairs from reviews, enabling the system to understand specific product features and user sentiments. Bipartite graphs are constructed to represent interactions between users/items and opinion aspects, facilitating the integration of user reviews into the GNN framework. A contrastive learning approach is employed to combine these graphs with TCN-generated review embeddings, enhancing the system's ability to capture complex relationships. Finally, a recommendation strategy is proposed which considers relevant opinion-aspects as explanations for recommendations. The experiments conducted on several benchmarks reveal that our method outperforms its competitors.**

Keywords—Recommender System, Explainability, Graph Neural Networks, Temporal Convolution Networks, User Reviews

I. INTRODUCTION

Recommender systems are intelligent systems designed to suggest products, services, or content to users based on their preferences and behaviors. These systems have become crucial to online platforms, from e-commerce sites like Amazon to streaming services like Netflix and social media networks like Facebook. Several methods have been developed for recommender systems, and among them, graph neural networks (GNNs) have achieved notable success due to their ability to model complex interactions between users and items [1]. There are several GNN-based recommender systems, such as Graph Attention Networks (GATs) [2], and GraphSAGE [3] PinSage [4], Neural Graph Collaborative Filtering (NGCF) [5], Lightweight Graph Convolutional Networks(LightGCN) [6], and Light Graph Contrastive Learning (LightGCL) [7]. While these models have demonstrated significant success in improving recommendation accuracy and personalization, they often need help with the black-box problem, needing both interpretability and explainability. Providing explanations helps users understand why a particular product or item is recommended, enhancing their trust and satisfaction by clarifying the reasoning behind recommendations. Furthermore, explainability allows businesses to identify and address biases, errors, or unintended consequences in the recommendation process [8].

This paper proposes an explainable GNN-based recommender system called **Exp**lainable **N**eural **G**raph **C**ollaborative **F**iltering (ExpNGCF) that integrates user-item interactions and user reviews into its process. To provide explainability, we leverage a Temporal Convolutional Network (TCN)[9] to encode user reviews into vector representations to capture contextual information in text-based reviews. Next, we extract opinion-aspect pairs, which include specific features of products mentioned in the reviews, along with users' opinions or sentiments regarding these aspects. To integrate them with the GNN process, we create two graphs that leverage the interactions between users and opinion-aspects and between items and opinion-aspects. In these graphs, the nodes represent users/items and opinion-aspects, with edges indicating their interactions. The model output consists of representation vectors for users, items, and opinion aspects. When an item is recommended to a user, a set of opinion aspects is also ranked, with the most relevant ones serving as explanations for the recommendation. This is achieved by computing the similarity between the corresponding vectors of users, items, and opinion aspects. Extensive experiments conducted on four real-world datasets demonstrate the effectiveness of our ExpNGCF model, which achieves significant performance improvements over the baseline models across various evaluation metrics.

II. RELATED WORKS

A. GNN-based recommenders

The core idea of GNNs is to utilize a message-passing mechanism, where nodes aggregate and transform information from their neighbors through multiple layers to generate rich node embeddings[10]. Several GNN-based recommender systems have been proposed in the literature. PinSage[4] uses graph convolution layers on the item-item graph for image recommendation on Pinterest. NGCF [5] leverages the power of GNNs to effectively model user-item interactions within a graph structure. Unlike traditional methods that rely solely on user-item interactions, NGCF incorporates the direct connections between users and items and higher-order connectivity patterns in the graph. While NGCF and traditional GNN-based models incorporate multiple layers of message passing, LightGCN[6] simplifies the GNN architecture by removing layer-specific parameters and directly aggregating user-item interaction information through a single layer. Recently, some methods such as Self-supervised Graph Learning (SGL) [11] and LightGCL[7] have introduced self-supervised learning approaches to enhance node representation learning with less supervised information. Interested readers are referred to a survey for more details on GNN-based recommenders [1].

B. Explainable recommender systems

Explainable recommender systems aim to justify their recommendations to users by providing reasons for suggesting a particular item. In recent years, textual information from user reviews has been explored to enhance explainability in recommendation models[12]. These reviews include users' experiences using the product or service and help users make better decisions[13]. Some studies, such as

(*) Corresponding author

[14], have enhanced explainability at the aspect level, where in recommender systems, aspects are considered explicit features of item nodes that provide useful information. TriRank[15] extracts aspects using users' opinions and models the user-item-aspect triad relationships in a heterogeneous graph. In [16], the relationship between users and aspects is modeled as a bipartite graph, where users' preferences are inferred based on location-specific aspects. In [14], recommendations are explained by finding a map between items and aspects. In [9], a pre-trained language model was used to extract semantic relationships in explanations by considering triple relationships. In [17], the TCN model is used to discover semantic relationships in user reviews. Some studies, such as [18], extract tags and attributes from user reviews and consider them as explanations. The method proposed in [19] learns the representations of user-item-explanation relationships in a heterogeneous graph and simultaneously performs item recommendations and explanations.

III. PROPOSED METHOD

In this paper, we propose an explainable GNN-based recommender system called ExpNGCF that integrates user-item interactions and user reviews into its process. The method includes three steps: (1) User review processing, (2) Learning representations, and (3) Model optimization. In the first step, user reviews are processed to extract opinion-aspect pairs. The TCN model is then applied to generate their embeddings. Two bipartite graphs are generated to model the interactions between users and opinion-aspects, as well as between items and opinion-aspects. The second step aims to create representations of users, items, and aspects. To this end, we input three bipartite graphs, user-item, user-review, and item-review graphs, into the GCN models to extract their representations. In the final step, a supervised loss function is used to optimize model parameters. Figure 1 shows the overall architecture of the proposed method.

A. User review processing

To integrate user reviews, we first extract opinion-aspects from user reviews. This involves identifying specific aspects or features of products mentioned in the reviews, along with the corresponding opinions or sentiments expressed by users about these aspects. For instance, consider a review stating, “The camera quality of this phone is excellent, but the screen resolution is poor”. This review extracts two opinion-aspect pairs denoted as “camera-excellent” and “screen-poor”. Here, "camera" serves as the aspect, representing the specific feature under evaluation, while “excellent” encapsulates the sentiment or evaluation expressed regarding that aspect. We proceed by constructing two bipartite graphs that leverage the interactions between users and opinion aspects (i.e. G^{UR}), as well as between items and opinion aspects (i.e. G^{IR}). In these graphs, the nodes represent both users/items and opinion-aspects, with edges indicating their interactions. In this step we also employ the TCN as a language model which captures the semantic information present in the reviews and encode them into vector representations.

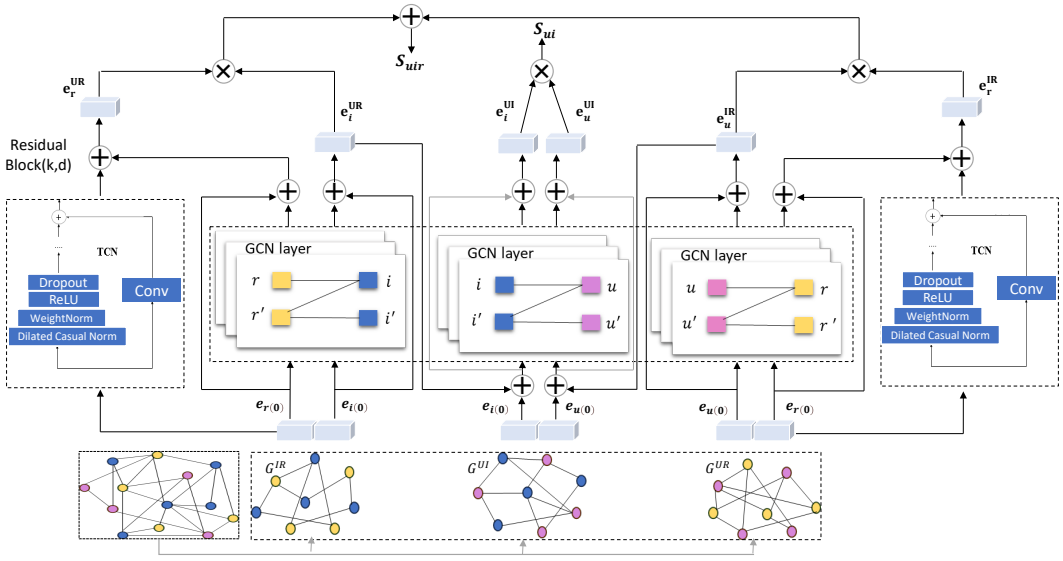

Figure 1. The overall architecture of the proposed method.

Our model includes three bipartite graphs extracted from user reviews (i.e. user-review: G^{UR} and item-review: G^{IR}) along with user-item interactions (i.e. G^{UI}). In each bipartite graph, edge weights indicate different interaction behaviors, defined as follows:

$$W_{uir}^{(k)} = \sum_{j \in J^{(K)}} Y_{uir}^{(k)} \quad (1)$$

where $J^{(K)}$, $W_{uir}^{(k)}$ and $Y_{uir}^{(k)}$, represent the set of nodes, interaction weights, and number of occurrences between user u, item i, and review r in the subgraph G^k, respectively.

B. Representation Learning

We employ two distinct architectures to enhance node representation in subgraphs: Graph Convolutional Network (GCN) and TCN. GCN aims to grasp intricate relational structures in graph data, while TCN specializes in sequence modeling, using temporal convolutional structures to capture semantic patterns and dependencies in sequences

effectively. Utilizing both models enables our method to efficiently represent complex relationships and sequential nuances within the varied graph.

Updating node embeddings: In our method, we use the GCN to generate node embeddings for each bipartite graph. GCNs work by aggregating features from a node's local neighborhood, effectively capturing structural information and complex relationships within the graph. The update equation is $E^{(l+1)} = \left(D^{-\frac{1}{2}}AD^{-\frac{1}{2}}\right)E^{(l)}$, where D is degree matix, A is adjacency matrix, and $E^{(0)}\epsilon R^{|d||n|}$ denotes the node embedding matrix for all nodes at initial layer. Using a high number of convolutional layers in GNNs may result in over-smoothing of node features in the final layer. To address this issue, one approach is to obtain the final node representation by aggregating the node representations from all layers. This aggregation step involves calculating the average of the node representations across each layer. Therefore, the l-layer graph convolution can be represented as:

$$C^l\left(A, E^{(0)}\right) = \frac{1}{L+1}\sum_{l=1}^{L}\left(D^{-\frac{1}{2}}AD^{-\frac{1}{2}}\right)^l E^{(0)} \quad (2)$$

User review representation: The goal in this step is to generate informative vector representations of user reviews using TCN. The review text is processed as a sequence of word embeddings $X = \{x_1, x_2, \dots, x_T\}$, where each $x_t \in R^d$ represents the embedding of the t-th word in a review of length T. The TCN applies 1D causal convolutions, where the output at time t is computed as:

$$h_t^{(l)} = \sum_{i=1}^{k-1} W_i^l \, x_{t-d*i}^{(l)} + b^{(l)} \quad (3)$$

with $W_i^{(l)}$ being the learnable convolution filters, $b^{(l)}$ the bias term, d the dilation factor, and k the kernel size. These convolutions ensure that the output at each step only depends on the current and previous embeddings, preserving temporal order. The network proceeds through multiple layers, each producing hidden representations $h_t^{(l)}$. After passing through all layers, the final sequence of embeddings from the last layer, $h^{(L)} = \{h_1^{(L)}, h_2^{(L)}, \dots, h_T^{(L)}\}$, captures both local and global dependencies in the text. To final embedding for the review text is $e_r = \frac{1}{T}\sum_{t=1}^{T} h_t^{(L)}$. The output of this step is embeddings for elements of user-review and item-review bipartite graphs (i.e. e_r^{UR} , e_r^{IR}).

Score opinion-aspect explanations: In this step, we introduce a mechanism to rank opinion-aspects for each recommended item. Initially, we obtain embeddings for both users and items. Subsequently, we employ a similarity function to determine the degree of resemblance between the user and item embeddings, gauging their compatibility. We apply a similarity function to see if an item is similar to a user as $S_{Rec} = f(e_u^{UI}, e_i^{UI})$, where f is a kernel function aims to compute the similarity between two embeddings. For the current recommendation task, our objective to provide explainability to this recommendation is to rank opinion-aspects and identify the top k relevant aspects. This process involves comparing the embeddings of users and items with representations of opinion-aspects, choosing the most similar ones. We use the following function to score relevancy of the opinion-aspects to a current recommendation:

$$S_{Exp} = (e_u^{UR}, e_r^{UR}) + (e_i^{IR}, e_r^{IR}) \quad (4)$$

For instance, in the scenario where a smartphone is recommended to a user, the system might prioritize opinion-aspects such as "*battery life - excellent*" or "*screen size - good*" based on their similarity to the user's preferences and the item's features.

C. Model optimisation

To train the model, we utilize a combination of two distinct loss functions. For obtaining embeddings of opinion aspects from reviews, we apply a Graph Convolutional Network (GCN) to bipartite graphs representing user-review and item-review interactions. Furthermore, applying GCNs to the user-item graph provides embeddings for both users and items. Consequently, each user, item, and opinion-aspect is represented by two distinct embeddings. To ensure consistency among these representations, we employ a self-supervised loss function, denoted as $L_{BPR-Exp}$, which serves as a measure of similarity between embeddings. This loss function is formulated as follows:

$$\begin{aligned} L_{BPR-Exp} &= -\sum_{u=1}^{N}\sum_{a\in E_{ui}}\sum_{b\notin E_{ui}} ln\ \sigma\Big(S_{Exp}(e_u^{UR}, e_i^{IR}, e_r^{IR}) \\ &- S_{Exp}(e_u^{UR}, e_a^{IR}, e_b^{IR})\Big) \end{aligned} \quad (5)$$

where, $\sigma(\cdot)$, P_u and E_{ui} represent the sigmoid function, the collection of items the user u have, and the set of reviews which user u mentioned for item i, respectively. We also employ the Bayesian Personalized Ranking (BPR) loss function that get the relative ranking of items and computes the loss as:

$$\begin{aligned} L_{\text{BPR-Rec}} &= -\sum_{u=1}^{N}\sum_{i\in P_u}\sum_{j\notin P_u} \ln\ \sigma\Big(S_{\text{Rec}}(e_u^{UI}, e_i^{UI}) \\ &- S_{\text{Rcc}}\big(e_u^{UI}, e_j^{UI}\big)\Big) \end{aligned} \quad (6)$$

This loss function ensures that the model learns to prioritize items based on their relevance to indivi dual users, thereby enhancing the quality and relevance of recommendations. To optimize the overall model, we combine $L_{\text{BPR-Exp}}$ as a self-supervised loss function with $L_{BPR-Rec}$ as:

$$L_{Total} = \alpha \cdot L_{BPR-Rec} + (1-\alpha)\cdot L_{BPR-Exp} \quad (7)$$

The parameter α allows fine-tuning the balance between the importance of recommendation and explanation scores.

IV. Experiments and Results

The performance of the proposed method was evaluated through several experiments. The experiments were conducted in the Google Colab environment utilizing GPU acceleration. The implementation leveraged the RecBole framework [20] for building recommendation system models and PyTorch as the underlying deep learning framework. Four real-world datasets including

AmazonMTV[1], HotelRec[2], TripAdvisor[3], and Yelp[4], were employed in the experiments. In each dataset, 80% of the data used as training sets and the remaining 20% as test sets in each iteration. These datasets are publicly available, and their specifications are provided in Table 1.

TABLE I. DATASET DETAILS AND SPECIFICATIONS

Dataset	Users	Items	Reviews	Ratings	Sparsity
AmazonMTV	7621	6985	1811	92824	99.82
TripAdvisor	31793	44838	6478	482272	99.96
Yelp	37033	42498	2524	356140	99.97
HotelRec	80386	68173	10878	681988	99.98

A. Evaluation measures

In this study, we utilize both Recall@k and NDCG@k as popular metrics to evaluate the performance of our proposed method [21, 22]. Recall@k measures the proportion of relevant items successfully retrieved within the top k recommendations, reflecting the system's ability to capture relevant items as is defined as:

$$Recall@k = \frac{|\{R_i\} \cap \{Top\_k_i\}|}{|\{R_i\}|} \quad (8)$$

where $|\{R_i\}|$, is the number of relevant items, and $|\{R_i\} \cap \{Top\_k_i\}|$ is the number of relevant items that are in the top-k recommendations. Moreover, NDCG@k evaluates the ranking quality by considering the position of relevant items in the recommended list, with higher ranks contributing more to the final score. It is defined as:

$$NDCG@k = \frac{DCG@k}{IDCG@k} \quad (9)$$

Here $DCG@k$ is computed as $DCG@k = \sum_{i=1}^{k} \frac{2^{rs_i}-1}{\log_2(i+1)}$,

where rs_i , is the relevance score of the item at position i and k is the number of ranked items. Moreover, $IDCG@k$ (Ideal DCG) is the maximum possible $DCG@k$, serving as a normalization factor. This normalization ensures that the NDCG score ranges from 0 to 1, with 1 representing the ideal ranking. NDCG is particularly useful for assessing the performance of recommendation systems and information retrieval algorithms, as it captures both the relevance and the ranking order of the recommended items. Furthermore, for the overall evaluation of the system, we use the geometric mean (GM) as a comprehensive metric. It provides a balanced measure, calculating joint performance in both proposal ranking and item explanation tasks [19]. The GM is denoted as $GM = \sqrt{S_R S_E}$ where, S_R and S_E are the item recommendation score and explanation ranking score, respectively.

B. Competitors

The proposed method is compared with a set of state-of-the-art methods include AMCF [14], A2-GCN[18], A2-GCN-light, which simplifies A2-GCN by using the idea of LightGCN [6], and ExpGCN [19]. Models such as A2-GCN and A2-GCNlight, which are based on GNNs, need to be specifically optimized for the explanation ranking task. Therefore, to compare these models with the proposed model, we utilize the learned node representations to generate explanation and obtain a preference score for explanations [19]. These competitors were selected based on their relevance and prominence in the field. To ensure a fair and comprehensive evaluation, the parameters and configurations of each method is set according to their established best practices. These configurations specified an embedding size of 128, a learning rate of 0.001, and a maximum of 170 training epochs. Additionally, an early stopping strategy was implemented with a stopping threshold set at 10.

C. Results

We performed several experiments to evaluate the results for item recommendation. The data was randomly divided into a training, validation, and test set in the 0.7, 0.15, and 0.15 ratios, respectively. For this task, all items that have not interacted with a user are treated as candidates, and the results are calculated by averaging the metrics of all users. These tables report the results based on Recall@10, Recall@20, NDCG@10, and NDCG@20. The outcomes of the item recommendation task are reported in Tables II-V for the AmazonMTV, Yelp, TripAdvisor, and HotelRec datasets, respectively. In these tables, R@10 and R@20 refer to Recall@10 and Recall@20, respectively. In the experiments, the embedding size for our method is set to 128. The best results are shown in bold in each table, and the average improvements are shown in the last row. Table 2 shows that the proposed method outperforms the other methods across all metrics. Specifically, the proposed method achieves an NDCG@20 of 0.0379, NDCG@10 of 0.0297, Recall@20 of 0.0861, and Recall@10 of 0.0552. Compared to the other methods, the proposed method shows an improvement of 4.12% in NDCG@20, 6.07% in NDCG@10, 3.86% in Recall@20, and 5.14% in Recall@10. These results indicate that the proposed method is highly effective in generating relevant and accurate recommendations. The results highlight the capability of the proposed method to capture complex interactions and provide personalized recommendations that align closely with user preferences.

The evaluation results across the Yelp, TripAdvisor, and HotelRec datasets exhibit patterns similar to those observed in the AmazonMTV, indicating the consistency and efficacy of the proposed method across diverse scenarios. For instance, the proposed method consistently outperforms other baselines across all evaluation metrics in the Yelp dataset. Notably, it achieves the highest NDCG@20 and NDCG@10 scores of 0.0322 and 0.0245, respectively, indicating its strong ability to rank relevant items higher in the recommendation list. Similarly, in the TripAdvisor and HotelRec datasets, the proposed method consistently exhibits competitive performance, with significant improvements in NDCG and Recall metrics compared to existing methods. These results underscore the effectiveness of the proposed method in enhancing recommendation quality and user satisfaction across various recommendation scenarios.

[1] https://cseweb.ucsd.edu/~jmcauley/datasets.html#amazon_reviews
[2] https://github.com/Diego999/HotelRec
[3] https://www.tripadvisor.com/
[4] https://www.yelp.com/dataset/challenge

TABLE II. RECOMMENDATION RESULTS ON AMAZONMTV

	R@10	R@20	NDCG@10	NDCG@20
AMCF	0.0317	0.0563	0.0176	0.0241
A2GCN	0.0392	0.0658	0.0214	0.0285
A2GCNlight	0.0350	0.0567	0.0183	0.0241
ExpGCN	0.0525	0.0829	0.0280	0.0364
ExpNGCF	**0.0552**	**0.0861**	**0.0297**	**0.0379**
%Improvement	5.14%	3.86%	6.07%	4.12%

TABLE III. RECOMMENDATION RESULTS ON YELP

	R@10	R@20	NDCG@10	NDCG@20
AMCF	0.0376	0.0633	0.0202	0.0270
A2GCN	0.0339	0.0568	0.0184	0.0245
A2GCNlight	0.0362	0.0625	0.0193	0.0263
ExpGCN	0.0454	0.0745	0.0242	0.0319
ExpNGCF	**0.0460**	**0.0751**	**0.0245**	**0.0322**
%Improvement	1.33%	0.91%	1.14%	0.97%

TABLE IV. RECOMMENDATION RESULTS ON TRIPADVISOR

	R@10	R@20	NDCG@10	NDCG@20
AMCF	0.0220	0.0373	0.0123	0.0166
A2GCN	0.0192	0.0326	0.0111	0.0150
A2GCNlight	0.0244	0.0389	0.0136	0.0177
ExpGCN	0.0289	**0.0470**	0.0172	0.0223
ExpNGCF	**0.0293**	0.0469	**0.0173**	**0.0224**
%Improvement	1.13%	-0.19%	0.68%	0.48%

TABLE V. RECOMMENDATION RESULTS ON HOTELREC

	R@10	R@20	NDCG@10	NDCG@20
AMCF	0.0359	0.0592	0.0192	0.0254
A2GCN	0.0310	0.0523	0.0167	0.0223
A2GCNlight	0.0302	0.0513	0.0160	0.0216
ExpGCN	0.0401	0.0664	**0.0215**	0.0284
ExpNGCF	**0.0412**	**0.0677**	0.0214	**0.0289**
%Improvement	2.57%	1.87%	-0.4%	1.78%

To evaluate the quality of these explanations, we use metrics such as Recall@k and NDCG@k to compute the overlap between the opinion aspects suggested by our method and those derived from the user's actual reviews. This enables us to assess how well our method aligns with the user's preferences and provides meaningful explanations for the recommendations. The results of these experiments are reported in Tables VI-IX. These results show that the proposed method stands out among the evaluated methods with the highest scores across all metrics. Specifically, Table VI indicates that our method achieves an NDCG@20 of 0.3229 and an NDCG@10 of 0.3028, indicating that the recommended opinion aspects are highly relevant and useful to users. Additionally, the Recall scores (Recall@20 and Recall@10) for the proposed method are 0.4798 and 0.4067, respectively, highlighting its ability to capture a substantial portion of the relevant opinion-aspects. Compared to alternative methods such as AMCF, A2GCN, A2GCNlight, and ExpGCN, the proposed method demonstrates improvements ranging from 0.39% to 0.55% across the evaluated metrics.

Tables VII-IX show similar results for Yelp, TripAdvisor, and HotelRec datasets. These results demonstrate consistent trends across all datasets. The results show that the proposed method consistently outperforms alternative methods. Notably, the proposed method achieves improvements ranging from 0.18% to 1.77% across the various metrics and datasets.

We also compared the GM metric results and reported them as radar charts in Figure 2. The results demonstrate the proposed method's robustness and versatility across diverse recommendation scenarios. A set of experiments also performed to analyze the impact of different learning rates on our model's performance. Figure 3 presents the results of the proposed method for various learning rates ranging from 0.001 to 0.5 on the AmazonMTV. The results demonstrate that as the learning rate increases from 0.001 to 0.01, there's a gradual decline in both Recall@10 and NDCG@10 values. As the learning rate increases, there's a gradual decrease in both measures. However, the decline in performance is more pronounced for higher learning rates, particularly at 0.1 and 0.5. Figure 4 presents an analysis of the results of the proposed method across various embedding sizes ranging from 8 to 128. The results show that as the embedding size increases, performance is generally improved across all datasets.

TABLE VI. EVALUATION OF EXPLANATION ON AMAZONMTV

	R@10	R@20	NDCG@10	NDCG@20
AMCF	0.3137	0.4036	0.1806	0.2054
A2GCN	0.3338	0.4078	0.2385	0.2590
A2GCNlight	0.3793	0.4613	0.2666	0.2893
ExpGCN	0.4051	0.4773	0.3013	0.3211
ExpNGCF	**0.4067**	**0.4798**	**0.3028**	**0.3229**
%Improvement	0.39%	0.51%	0.48%	0.55%

TABLE VII. EVALUATION OF EXPLANATION ON YELP

	R@10	R@20	NDCG@10	NDCG@20
AMCF	0.0936	0.1530	0.0514	0.0678
A2GCN	0.1291	0.1845	0.0787	0.0939
A2GCNlight	0.1212	0.1752	0.0729	0.0877
ExpGCN	0.1754	0.2432	0.1094	0.1281
ExpNGCF	**0.1757**	**0.2442**	**0.1098**	**0.1286**
%Improvement	0.18%	0.39%	0.38%	0.40%

TABLE VIII. EVALUATION OF EXPLANATIONON TRIPADVISOR

	R@10	R@20	NDCG@10	NDCG@20
AMCF	0.1037	0.1538	0.0566	0.0715
A2GCN	0.1225	0.1690	0.0849	0.0985
A2GCNlight	0.1134	0.1568	0.0766	0.0893
ExpGCN	0.1566	0.2121	0.1081	0.1243
ExpNGCF	**0.1579**	**0.2126**	**0.1100**	**0.1251**
%Improvement	0.84%	0.25%	1.77%	0.58%

TABLE IX. EVALUATION OF EXPLANATION ON HOTELREC

	R@10	R@20	NDCG@10	NDCG@20
AMCF	0.0488	0.1041	0.0247	0.0403
A2GCN	0.0975	0.1419	0.0595	0.0718
A2GCNlight	0.0790	0.1178	0.0479	0.0591
ExpGCN	0.1200	0.1714	0.0749	0.0894
ExpNGCF	**0.1219**	**0.1730**	**0.0759**	**0.0904**
%Improvement	0.39%	0.51%	0.48%	0.55%

V. CONCLUSION

In this paper, we proposed an explainable GNN-based recommender system called ExpNGCF. This approach integrates user reviews into the recommendation process. It uses the TCN to encode user reviews into vector representations, capturing temporal dynamics and contextual information. Additionally, we extract opinion-aspect pairs from user reviews, enabling the system to understand specific features and sentiments expressed by users. These aspects are then used to construct bipartite graphs, capturing interactions between users/items and opinions. Finally, a self-supervised learning strategy combines bipartite graphs with TCN-generated review embeddings. Performed experiments on multiple datasets, including AmazonMTV, Yelp, TripAdvisor, and HotelRec, demonstrate the superiority of the proposed ExpNGCF method over existing approaches.

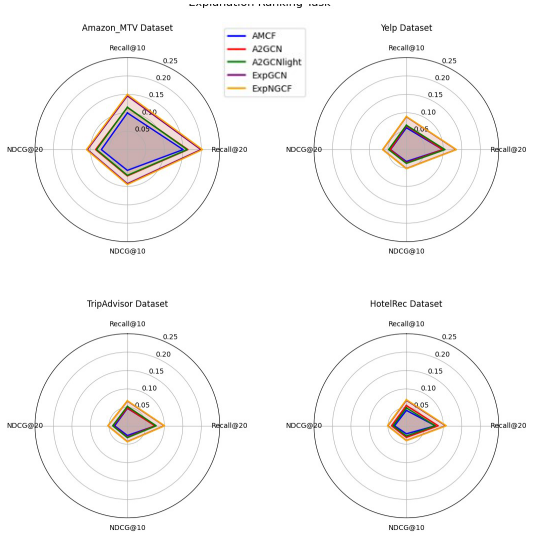

Figure 2. Comparing the performance of methods in terms of GM.

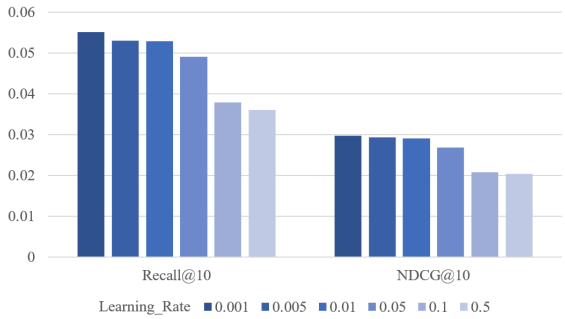

Figure 3. The proposed method using various learning rates on AmazonMTV.

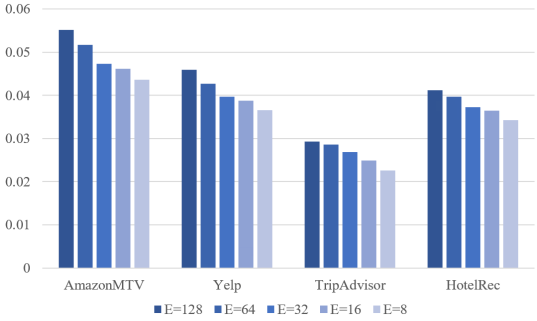

Figure 4. Recall@10 of the proposed method for various embedding sizes.

REFERENCES

[1] C. Gao et al., "A Survey of Graph Neural Networks for Recommender Systems: Challenges, Methods, and Directions," ACM Transactions on Recommender Systems., vol. 1, no. 1, pp. 1-51, 2023,

[2] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph Attention Networks," presented at the International Conference on Learning Representations, 2018.

[3] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive Representation Learning on Large Graphs," presented at the Proceedings of the 31st International Conference on Neural Information Processing Systems, pp. 1025–1035, 2017.

[4] R. Ying, R. He, K. Chen, P. Eksombatchai, W. L. Hamilton, and J. Leskovec, "Graph convolutional neural networks for web-scale recommender systems," in Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2018, pp. 974-983.

[5] X. Wang, X. He, M. Wang, F. Feng, and T.-S. Chua, "Neural graph collaborative filtering," in Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2019, pp. 165-174.

[6] X. He, K. Deng, X. Wang, Y. Li, Y. Zhang, and M. Wang, "Lightgcn: Simplifying and powering graph convolution network for recommendation," in Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2020, pp. 639-648.

[7] C. Xuheng, H. Chao, X. Lianghao, and R. Xubin, "Light GCL: Simple Yet Effective Graph Contrastive Learning for Recommendation," presented at the The Eleventh International Conference on Learning Representations, 2023.

[8] X. Wang, D. Wang, C. Xu, X. He, Y. Cao, and T.-S. Chua, "Explainable Reasoning over Knowledge Graphs for Recommendation," Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, no. 01, pp. 5329-5336, 2019,

[9] L. Li, Y. Zhang, and L. Chen, "On the relationship between explanation and recommendation: Learning to rank explanations for improved performance," ACM Transactions on Intelligent Systems and Technology, vol. 14, no. 2, pp. 1-24, 2023.

[10] S. Wu, F. Sun, W. Zhang, X. Xie, and B. Cui, "Graph neural networks in recommender systems: a survey," ACM Computing Surveys, vol. 55, no. 5, pp. 1-37, 2022.

[11] J. Wu et al., "Self-supervised graph learning for recommendation," in Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2021, pp. 726-735.

[12] S. Pan, D. Li, H. Gu, T. Lu, X. Luo, and N. Gu, "Accurate and explainable recommendation via review rationalization," in Proceedings of the ACM Web Conference 2022, pp. 3092-3101.

[13] L. Li, Y. Zhang, and L. Chen, "Personalized transformer for explainable recommendation," presented at the Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, pp. 4947-4957, 2021.

[14] D. Pan, X. Li, X. Li, and D. Zhu, "Explainable recommendation via interpretable feature mapping and evaluation of explainability," presented at the Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence Article No.: 373, Pages 2690 - 269

[15] X. He, T. Chen, M.-Y. Kan, and X. Chen, "Trirank: Review-aware explainable recommendation by modeling aspects," in Proceedings of the 24th ACM International Conference on Information and Knowledge Management, pp. 1661-1670, 2015.

[16] R. Baral, X. Zhu, S. Iyengar, and T. Li, "Reel: Review the aware explanation of location recommendation," in Proceedings of the 26th Conference on User Modeling, Adaptation, and Personalization, pp. 23-32, 2018.

[17] N. Heidari, P. Moradi, A. Koochari, An attention-based deep learning method for solving the cold-start and sparsity issues of recommender systems, Knowledge-Based Systems, Volume 256, 28 November 2022, 109835

[18] F. Liu, Z. Cheng, L. Zhu, C. Liu, and L. Nie, "An attribute-aware attentive GCN model for attribute missing in recommendation," IEEE Transactions on Knowledge and Data Engineering, vol. 34, no. 9, pp. 4077-4088, 2020.

[19] T. Wei, T. W. Chow, J. Ma, and M. Zhao, "Expgcn: Review-aware graph convolution network for explainable recommendation," Neural Networks, vol. 157, pp. 202-215, 2023.

[20] W. X. Zhao et al., "Recbole: Towards a unified, comprehensive and efficient framework for recommendation algorithms," in Proceedings of the 30th ACM International Conference on Information & Knowledge Management, 2021, pp. 4653-4664.

[21] Y.-M. Tamm, R. Damdinov, and A. Vasilev, "Quality metrics in recommender systems: Do we calculate metrics consistently?," in Proceedings of the 15th ACM Conference on Recommender Systems, 2021, pp. 708-713.

[22] M. Jalili, S. Ahmadian, M. Izadi, P. Moradi, and M. Salehi, "Evaluating Collaborative Filtering Recommender Algorithms: A Survey," IEEE Access, vol. 6, pp. 74003-74024, 2018.