ZANCO Journal of Pure and Applied Sciences

TYPE Original article
DOI http://dx.doi.org/10.21271/ZJPAS.37.5.11

OPEN ACCESS

*Corresponding author
Shahla Havas
sh.havas@uok.ac.ir

RECEIVED : 28/12 /2024
ACCEPTED :28/05/ 2025
PUBLISHED :31/ 10/ 2025

KEYWORDS:
Recommender Systems, Sparsity Problem, GNN

Position-Aware Neural Graph Collaborative Filtering to Resolve the Sparsity Problem in Recommender Systems

Shahla Havas[1] , Alireza Abdollahpouri[1] , Nasser Yazdani[2]
[1]Department of Computer Engineering, University of Kurdistan ,Kurdistan , Iran
[2]Department of Electrical and Computer Engineering, University of Tehran, Tehran , Iran

ABSTRACT

Users' experience on the internet is positively influenced by recommendation systems, which help them in navigating through a lot of information available to them. The accuracy and relevance of recommendations can be improved only if advanced techniques like GNNs are used due to the increasing complexity associated with user-item interactions. Detailed patterns in preferences and behavior are captured by GNNs which model relationships between users and items. To this end, we present a Position-Aware Neural Graph Collaborative Filtering (PA-NGCF) model with two key differences from previous graph neural network-based recommender systems. In terms of our primary contribution, we have come up with a new way of creating node embeddings that make use of limited rating information from users to adequately capture user-item interactions. Furthermore, during message passing phase in our model, the positional information about nodes within the graph structure is explicitly included enabling richer understanding about the role and significance of each node involved in recommendation process. Two real-world datasets were used for extensive experiments to show how effective our approach was. Our findings indicate that PA-NGCF method effectively reduces the rating prediction error within recommendation systems. This error has been evaluated using the MAE and RMSE metrics. Our findings highlight the potential of the proposed method in increasing the quality of recommendations and addressing cold start problems in sparse datasets, as the reduction in prediction error leads to more accurate ranking of items adapted to user preferences.

1.Introduction

Different digital advertising networks, e-commerce platforms and social media platforms are embracing personalized recommendations in different contexts. In other words, personalized recommendation seeks to estimate the likelihood that a consumer will take on a particular product based on their previous engagements such as clicks, ratings and purchases. Collaborative filtering (CF) has solved this problem because it assumes that users who have similar behavior would also have similar preferences for items. To implement this assumption, one could parametrically represent users and items that would be followed by predicting user preference from these parametric reconstructions of interactions in the past (Koren et al., 2021).

Simply put, learnable CF models can be split into two main parts: embeddings which map users and items into numerical representations; and interaction modeling which can regenerate historical interactions from learnt embeddings. For example; among the most commonly used methods in matrix factorization is direct embedding of user and item IDs as vectors where user-to-item interactions are represented by inner product (Koren et al., 2009). Collaborative deep learning enhances traditional matrix factorization by incorporating deep representations derived from additional item information(Xu et al., 2015). Neural collaborative filtering models take a different approach by replacing the simple dot product used in MF with more sophisticated neural networks (Xiao et al., 2017), while translation-based CF models use Euclidean distance as a measure of interaction.(Liang et al., 2018).

Although the said methods could lead to successful embedding techniques, they are however effective embedding techniques for collaborative filtering. The main reason is that the embedding function does not have an explicit encoding of the collaborative signal that is so critical. That signal in this case is hidden in user-item interactions and reflects the similarity between different users or items with respect to their behavior patterns. It follows that embeddings of that nature and formed using any of the mentioned methods may not provide the full contextual range necessary for productive collaborative filtering with such a significant collaborative signal's information encoded elsewhere (Wu et al., 2020). In addition current techniques rely on an amount of user item interactions. The data related to these interactions is usually scarce and does not offer enough details, for training the model effectively (Yang et al., 2018). Consequently if the embeddings fail to capture filtering cues these methods must heavily depend on the interaction function to make up for the lack of collaborative information, in the embeddings.

Overall, the entire model faces the challenge of interaction data sparsity. However, this issue becomes even more noticeable for users with very few interactions, who are faced with the cold start problem. During the message passing process in graph neural networks, these users have a limited number of neighbors from whom they can aggregate information. As a result, the model’s ability to learn meaningful representations for such users is weakened. This limitation in neighborhood information restricts the model’s capacity to effectively update and generalize user features, ultimately reducing recommendation accuracy for these users.

To address this limitation, we suggest the usage of node spatial positions in the graph. Our approach therefore aims at capturing more intricate relationships beyond direct user-item interactions by analyzing node proximities. More so, these higher-order interactions will be added during model training thereby leading to a richer representation of user preferences and item characteristics. As a result, such an approach not only improves the quality of learned embeddings but also makes it possible for the model to better reflect collaborative signals underlying data. So doing, we aim at developing an innovative method that can enhance recommendation systems’ performance while providing users with more accurate and personalized suggestions.

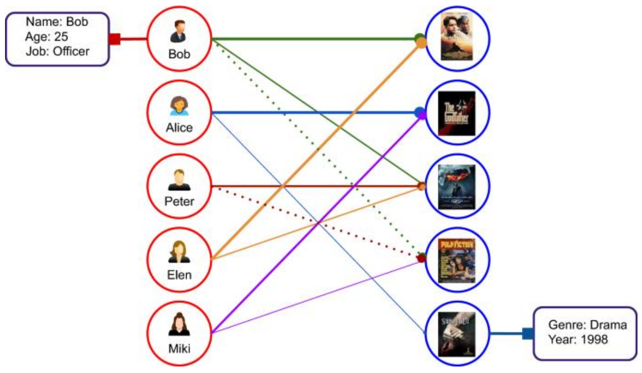

Figure 1: Bipartite Graphs in recommendation

Contribution

In comparison to prior graph neural network-based recommender systems, our PA-NGCF introduces two key innivations that distinguish it from prior works:

1- Unlike traditional graph neural networks that aggregate information solely based on node connectivity, PA-NGCF incorporates the relative positions of nodes within the graph to enhance the message passing process. Our intention in using position-based neighborhood selection is to enrich the aggregated information, especially for users with cold start problems, by providing more informative embeddings.

2- The model is trained to predict user preferences by computing the inner product of user and item embeddings, which directly corresponds to predicted ratings. We use the Mean Squared Error (MSE) loss function to train the model to directly estimate the rating a user would assign to each item.

We test our proposed method on two real-world datasets extensively to demonstrate its performance capabilities. The results show that our approach is effective for generating accurate recommendations, outperforming existing baselines and demonstrating the potential of graph-based techniques in recommendation systems.

2.Notations and Preliminaries

2.1 Recommendation Systems and Bipartite Graphs

In the modern world that is so full of information, recommendation systems are also becoming increasingly important in getting users through lots and lots of content as well as enabling them to discover relevant items. The involvement of these systems plays a key role in increasing user satisfaction, raising engagement levels, and fostering business growth. In this case, the bipartite graphs are employed to suggest items that may be of interest to users. This is accomplished by dividing the vertices into two sets and connecting vertices from different sets with edges. Such a representation is common in recommender systems, where one set represents users, the other items and edges imply interactions or preferences. As shown in Figure 1, the bipartite graph consists of two types of nodes: users and items. The edges in this graph connect user nodes to item nodes. Each user node represents an individual user in the system, while each item node corresponds to a product, service, or other entities available in the system. The edges between user and item nodes indicate interactions such as purchases, likes, ratings, and other forms of user engagement with the items. Consequently, employing bipartite graphs when modeling user-item relationships enable recommendation systems to make personalized suggestions that enhance the end-user experience and system engagement.

Table 1 The key notations used in the proposed method

Notation	Description
U	**A set of user nodes in the graph**
I	**A set of item nodes in the graph**
V	$U \cup I$
$\mathbf{p}_u$	**The position vector associated with user node u**
$\mathbf{p}_i$	**The position vector associated with item node i**
e_u	**The embedding vector of user u**
e_i	**The embedding vector of item i**
$N(p_u)$	**The position-aware neighborhood of user u**
$N(p_i)$	**The position-aware neighborhood**

	of item i
h_u^l	**The hidden representation of user u at the l^{th} convolutional layer**
h_i^l	**The hidden representation of item i at the l^{th} convolutional layer**
R	The set of user-item pairs with known ratings in the training set
$r_{u,i}$	The known rating given by user u to item i
W^l	The weight matrix of the l^{th} convolutional layer
b^l	The bias vector of the l^{th} convolutional layer
d	The dimensionality of the node embedding vectors
ε	A threshold parameter such that two nodes are treated as neighbors if the distance between their position vectors is less than ε.

2.2 Notations

Let $G = (V, E)$ be an undirected graph, where $V = \{v_1, v_2, ..., v_n\}$ is the set of N nodes, E is the set of edges describing the relations between. The key notations used in the proposed method are summarized in the Table1. In Section 4, each variable and its application are explicitly discussed.

3.Related work

This section discusses research papers that explore the use of graph network (GNN) based recommender systems and review driven recommendation models.

Graph Neural Networks (GNNs) are used tools for providing recommendations. Numerous research works have employed GNNs to handle quantities of data (Hamsheen and Flah, 2023, Anwer et al., 2024). The inductive approach has become the prevailing method for utilizing GNNs in recommendation systems (Wu et al., 2022). This approach is effective as it learns from data segments (subgraphs) (Ying et al., 2018, Hamilton et al., 2017) enabling predictions on information and managing extremely large datasets. Graph Neural Collaborative Filtering (NGCF) (Wang et al., 2019) introduces a framework that utilizes graph networks to improve collaborative filtering by adeptly modeling user item interactions within a graph structure. In contrast, to Graph Convolutional Networks (GCNs) which often face challenges of computational expenses and intricacy LightGCN (He et al., 2020) simplifies the graph convolution operation while retaining the capacity to capture crucial user item interactions. By employing a lightweight architecture, LightGCN reduces the number of parameters and computational overhead, making it feasible to process large graphs in real-time.

GraphSAGE (Hamilton et al., 2017) a known approach developed by Hamilton and colleagues in 2017 is widely used for exploring information, about nodes within a network. It examines clusters of interacting nodes (known as subgraphs) surrounding each node to comprehend the networks structure and gain insights into node attributes. Several new methods have emerged following GraphSAGE, such as PinSAGE introduced by (Ying et al., 2018, Afoudi et al., 2023). PinSAGE builds upon GraphSAGEs principles. Focuses on analyzing data types like images and text sourced from Pinterest combining information from both modalities to describe individual nodes.

EGI, which stands for Edge Group Inference is a method that learns about groups of interconnected nodes surrounding a node without requiring labels. These groups are processed using Graph Neural Networks (GNNs) to generate representations for the node. EGI has demonstrated its effectiveness across networks with structures as shown (Zhu et al., 2021). Furthermore the concept of EGI gap was introduced as a metric to evaluate the similarity between two networks suggesting limitations in transferring knowledge, between GNNs based on their underlying architecture similarities.

Heterogeneous graph neural networks (HGNNs), on the other hand, are useful in scenarios when there are different types of nodes and different types of connections. Compared to normal GNNs, these methods are more appropriate for non-simple networks. HGT (Hu et al., 2020) is one of the most frequently used HGNNs. Its aim is to increase efficiency in the processing of data whereby the relevant portion of the network is divided into small components in accordance with the node location. Additionally, it differs in its treatment of some nodes and edges to some node and edge types(Yang et al., 2023) also offers another HGNN called SeHGNN which is able to simplify the process in this case through quickly gathering information from its neighbors. Its intention is to appreciate the significance of the network by studying larger equations of the network, consolidating reconstructive evidence from different sources.

This direction has great potential as recent studies have shown that combining heterogeneous graph neural networks (HGNNs) with contrastive learning can address challenges like data sparsity and noise in recommendation systems. For example, (Lin et al., 2022) have demonstrated the effectiveness of considering both structural and semantic relationships between nodes when creating contrasting pairs. However, SimGCL (Yu et al., 2022) and XSimGCL (Yu et al., 2023) are examples of this which have made the process of contrastive learning easier by adding noise to the embedding area hence improving its efficiency and accuracy. On the other hand, HGCL improves recommendation results in practical datasets through what is called “meta-networks” that enable it to make personalized comparisons. In summary, HGCL exploits diverse relationships within the data using contrastive learning for more accurate user and item representations.

Despite the significant advancements in graph-based recommender systems, a critical gap remains in the explicit utilization of node positional information within the graph structure. Most existing methods focus on leveraging structural, semantic, or heterogeneous relationships to learn user and item embeddings, yet they often overlook the potential value embedded in the relative positions of nodes. Node position can capture unique contextual and relational patterns-such as proximity to influential nodes or specific communities-that are not fully represented by traditional features. In this work, we address this gap by incorporating positional information of nodes directly into the embedding process. By doing so, our approach aims to construct richer and more informative representations for users and items.

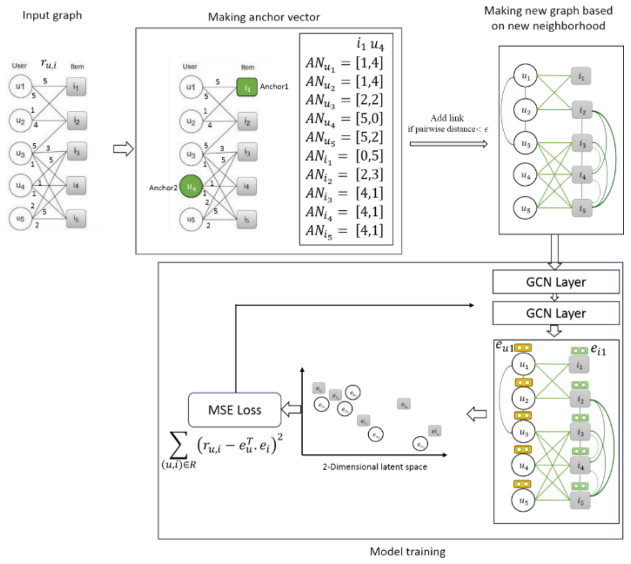

Figure 2: Architecture of the proposed method

4.Proposed Method

In this section, we present our Position-Aware Neural Graph Collaborative Filtering (PA-NGCF) framework, a novel GNN-based recommender system that leverages the inherent homophily and structural properties of the user-item bipartite graph to improve recommendation performance. As shown in Figure 2, our proposed method consists of two key components: (1) position aware neighbors’ selecting, and (2) training the model to predict user preferences based on the enhanced graph representation.

4.1 position aware neighbors’ selecting

In the first stage of our proposed PA-NGCF model, we incorporate the relative positions of nodes within the graph structure into the message passing process of the graph neural network. Traditional graph neural networks rely solely on the connectivity patterns between nodes, aggregating information from neighboring nodes during message passing. However, the position of a node in the graph can provide important contextual information about its role and importance within the recommendation system. To capture the positional information of nodes, we first randomly select a set of anchor nodes in the graph to capture the positional information of nodes. These anchor nodes serve as reference points for determining the positions of all other nodes. We then compute the distance between each node and the anchor nodes, measured by the number of hops or steps required to reach the anchor nodes. This process assigns a position vector to each node, where the elements of the vector represent the distances to the anchor nodes.

Let u be a node in the graph, and let p_u be the position vector associated with node u, where $\mathrm{p}_u = [\mathrm{d}_{u1}, \mathrm{d}_{u2}, \dots, \mathrm{d}_{uk}]$ and d_uk represents the distance between node u and the k-th anchor node. For example, in Figure 2, nodes i_1 and u_4 are selected as anchor nodes. Therefore, the distances from other nodes to these two anchor nodes are calculated and included in the position vectors. As shown, for node u_5, the distance to anchor i_1 is 4, and the distance to anchor u_4 is 2. Consequently, the position vector for node u_5 is given by [4,2].

4.2 Training Model

Based on the graph G, a GNN-based recommender system generates user/item embeddings by graph neural networks to predict user's preference.

4.2.1 Embedding Node Vectors

In the first step of training PA_NGCF model, we learn the initial embedding vectors for the nodes (users and items) in the enhanced graph. Let u and i represent a user and an item, respectively. We denote the embedding vectors of u and i as e_u and e_i, respectively.

4.2.2 Position-Aware Convolution Layer

In the convolutional layers of our PA-NGCF model, we introduce a novel approach to defining the neighborhood of each node during the message passing process. Instead of relying solely on the connectivity patterns between nodes, we leverage the position vectors computed in the previous stage of our model to determine the relevant neighbors for each node. We define the position-aware neighborhood $N(p_u)$ of user u and $N(p_i)$ of item i as the sets of nodes whose position vectors are within a certain distance threshold ε from the position vectors of u and i, respectively. Specifically, two nodes are considered neighbors if the distance between their position vectors is less than ε. This criterion is used to construct the new neighborhood for each node as described in Equation 1.

$$N(p_u) = \{v \in V \mid (p_u - p_v)^2 < \varepsilon\} \qquad (1)$$

$$N(p_i) = \{v \in V \mid (p_i - p_v)^2 < \varepsilon\}$$

During the convolution operation, user node u aggregates the previous layer representations of the nodes in its position-aware neighborhood $N(p_u)$, and item node i aggregates the features of the nodes in its position-aware neighborhood $N(p_i)$, rather than the entire sets of neighboring nodes based on the graph connectivity. The convolution operation at layer l is then defined as:

$$h_u^l = \sigma\Big(\sum_{v \epsilon N(p_u) \cup \{u\}} \frac{1}{\sqrt{|N(p_u)||N(p_v)|}} \times W^l \times h_v^{l-1} \Big)$$

$$h_i^l = \sigma\Big(\sum_{v \epsilon N(p_i) \cup \{i\}} \frac{1}{\sqrt{|N(p_i)||N(p_v)|}} \times W^l \times h_v^{l-1} \Big) \qquad (2)$$

Where W^l and b^l are the weight matrix and bias vector of the l_{th}convolutional layer, respectively, and σ is a non-linear activation function.

4.2.3 Loss Function

To train the PA-NGCF model, we minimize the Mean Squared Error (MSE) loss between the predicted ratings and the ground-truth ratings. Let R be the set of user-item pairs with known ratings in the training set. The loss function is defined as:

$$MSE = \sum_{(u,i) \in R} (r_{u,i} - e_u^T . e_i)^2 \qquad (3)$$

Where $r_{u,i}$ is the known rating given by user u to item i, and $e_u^T . e_i$ is the dot product between the final embedding vectors of u and i, representing the predicted rating. By minimizing this loss function, the model learns to predict ratings that are close to the true ratings, leveraging the structural information captured by the convolutional layers and the homophilic connections in the enhanced user-item graph.

5.Experimental Results

This section outlines the experimental setting and findings of our analysis. Here, we contrast the proposed PA_NGCF with existing rating prediction models. Experiments were performed using two datasets to assess the effectiveness of the proposed method. The key statistics for these datasets are presented in Table2.

Table 2: Basic statistics of 2 datasets

Dataset	Movielens	Epinions
#Users	943	5654
#Items	1650	4380
#Interaction	80000	25442

5.1 Baseline

To compare to some ranking prediction techniques that exist in various ways, two different methods were selected as benchmarks: matrix factorization and a GNN-based approach. These Matrix Factorization methods include NMF (Lee and Seung, 2000) and SVD++ (Koren, 2008), whereas these GNN-based techniques comprise SIMGCL (Yu et al., 2022), XSIMGCL (Yu et al., 2023), and NGCF (Wang, 2019 #10). The evaluation metric was root mean square error (RMSE) and Mean absolute Error(MAE).

5.2 Experimental settings

Our PA_NGCF model was developed using PyTorch (Paszke et al., 2019). The datasets were divided into three parts: 80% for training, 10% for validation and the remaining 10% for testing. We used two GCN layers in this study. The evaluation metrics used in our experiments are Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE). MAE measures the average absolute difference between the predicted ratings and the actual ratings, providing an overall indication of prediction accuracy. RMSE, on the other hand, calculates the square root of the average squared differences between predicted and actual ratings, thereby penalizing larger errors more heavily.

In Section 5.2.1 we provided the results for the proposed method with the variations of number of anchors used during computation. Distance threshold ε is set to be equal to 2 in experiments. Also, we analyze our proposed method without considering positions of nodes also known as C_NGCF (see Section 4.2.1). This permits us to assess how node positions affect our model's performance as discussed in Section 5.2.2.

5.2.1 Impact of Anchor Count on Model Performance

In this section, we examine the impact of varying anchor counts on the performance of our proposed model. Specifically, we selected different percentages of nodes from the graph to serve as anchors, including 2%, 5%, 10%, 15%, 20%, and 25%. The results obtained for each of these percentages are illustrated in Figure 3.

As illustrated in Figure 3, it is evident that both very low and very high anchor counts (specifically, 2% and 25%) lead to suboptimal model performance. In contrast, an anchor count of 5% yielded the best results for certain datasets. However, the optimal anchor percentage varies depending on the structure of the dataset, particularly in terms of clustering and sparsity. For instance, the Epinions dataset, which is sparser and has a more dispersed graph structure, requires a higher number of anchors to achieve optimal performance. The best results for this dataset were obtained when there were 10% of anchors. For low values of anchor number, the model's ability to correctly locate nodes on the graph may be compromised. However, too many anchors can cause all node positions to become more similar thereby leading to over-smoothing. This effect of over-smoothing leads to the inability of the model in distinguishing between nodes hence reducing its performance.

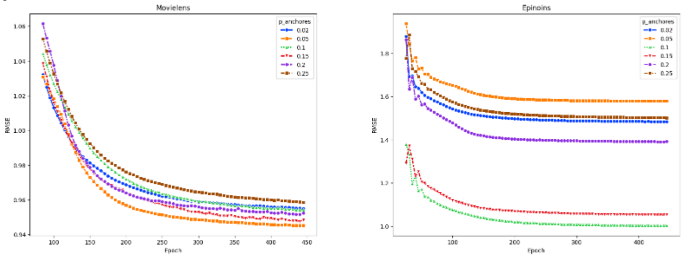

Figure 3: Impact of Anchor Count on PA_NGCF Performance

5.2.2 Impact of Position Awareness on the Effectiveness of the Proposed Approach

Findings on how position awareness affects our newly proposed approach are displayed in Figure 4 that shows that incorporating position awareness markedly improved performance. By accounting for a node's location during message passing, our PA_NGCF model can better embed the structural properties of user-item graphs and form better informed representation for nodes which are robust to changes in data.

The position-aware convolution operation enables the model to prioritize messages from nodes that are in similar positions within the graph, thus potentially leading to more accurate recommendations. This strategy has been particularly helpful for the Epinions dataset since it allows nodes with fewer neighbors to receive messages from nodes in similar positions, thereby partially compensating for lack of neighboring nodes. The sparse Epinions data set has been largely compensated for by higher order neighborhoods that consider cold start node neighbors using PA_NGCF model.

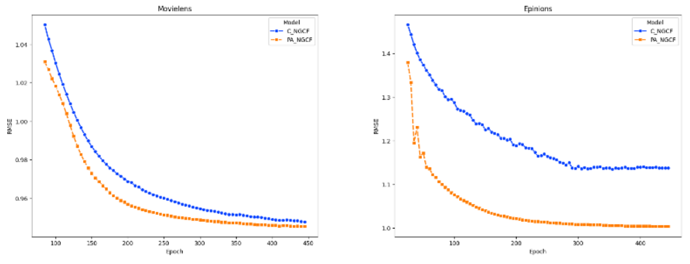

Figure 4: Impact of Position Awareness on proposed aproach

5.3 Performance comparison to baselines

Table3 shows a performance comparison among different recommendation models on two datasets. For example, our positional information incorporated message passing approach is the most important innovation achieved in our PA-NGCF model. It enables our graph neural network to learn more comprehensive representations of users and items by considering both the connectivity patterns and structural position of nodes within a graph.So PA-NGCF is giving the best results with respect to MAE as well as RMSE across both datasets, demonstrating its ability for making precise recommendations. C-NGCF also does well right behind PA_NGCF. In sparse graphs, where user-item interactions are limited, leveraging positional information can compensate for the lack of direct connections by capturing higher-order relationships and global context. This helps the model infer user preferences more effectively, even when explicit interaction data is scarce. In sparse graphs, where user-item interactions are limited, leveraging positional information can compensate for the lack of direct connections by capturing higher-order relationships and global context. This helps the model infer user preferences more effectively, even when explicit interaction data is scarce.

Table 3: MAE and RMSE performance of PA_NGCF and other baseline models

Model \Dataset	Measure	Movielens	Epinions
NMF	MAE	0.9612	1.0510
	RMSE	1.1050	1.2284
SVD++	MAE	0.9756	0.9845
	RMSE	1.0250	1.1913
SIMGCL	MAE	0.9321	0.9725
	RMSE	0.9689	1.1920
XSIMGCL	MAE	0.9134	0.9845
	RMSE	0.9899	1.2420
C-NGCF	MAE	0.9012	0.9725
	RMSE	0.9561	1.1312
PA-NGCF	MAE	**0.8951**	**0.9645**
	RMSE	**0.9412**	**1.0510**

5.4 Time Complexity Analysis

In this section, we focus particularly on the time complexity analysis of our PA-NGCF framework. As we concentrate on the completion of the rating matrix in the offline phase, this factor of time complexity is not of great importance. Nonetheless, it is important to know the computational requirements of our method to justify its scalability and efficiency.

Assuming that we have k anchor nodes, the distance from each node to each one of these anchor nodes must be computed in order to achieve the anchor vectors. This operation incurs a time complexity of O(N·K), where N refers to the total number of nodes in the graph (users and items). For each node, we find the distances to all K anchor nodes which costs O(K) operations. If this is done to all N nodes, the aggregate time complexity would be O(N·K). After this step, each node has a anchor vector which indicates its distances to the anchor nodes. In the course of the message passing process and while defining the new neighborhood concept, it becomes necessary to find out the position vectors, and more specifically the distances between them, since they represent the relevant neighbors for each node. The distance computation of all the node pairs results in a time complexity of O(N^2). If the training process is repeated over I iterations then the overall time complexity of our procedure is O(I(N^2 +(NK)) Overall, it is important to understand that although our method shows polynomial complexity, it has to be interpreted with respect to our problem. Given that these operations are offline, the scalability aspect of the PA-NGCF framework still remains under control. This enables us to exploit the rich structural information contained in the user-item graph and improve recommendation performance without worrying too much about running time efficiency lately.

Conclusion

We have thus developed the Position-Aware Neural Graph Collaborative Filtering (PA-NGCF) model that effectively addresses recommendation system issues, particularly on sparse datasets. By utilizing limited rating information supplied by users and considering positions in message passing, our model has been able to successfully capture complex user-item interactions leading to improved quality of recommendations. Results from extensive experiments show that PA-NGCF outperforms existing benchmark models, validating the efficacy of incorporating higher-order relations as a means to mitigate the cold start problem for better recommendation accuracy.

There are many promising future directions for this work. One possibility would be to include other aspects about users and items in addition to contextual information like auxiliary data and time-dependent behavior. This can help us in integrating multiple sources of data, which will help improve robustness of representations through learning while enhancing recommendations' accuracy. Moreover, advanced techniques in GNNs including different graph structures might also yield more insights into optimizing recommender systems. In summary, these findings highlight the potential application of graph-based approaches in solving future.

Conflict of interest

The authors declare that there is no conflict of interest regarding the publication of this article.

References

Afoudi, Y., Lazaar, M. & Hmaidi, S. 2023. An enhanced recommender system based on heterogeneous graph link prediction. *Engineering Applications of Artificial Intelligence,* 124**,** 106553.

Anwer, M. A., Qattan, G. A. & Ali, A. M. 2024. Ocular disease classification using different kinds of machine learning algorithms. *Zanco Journal of Pure and Applied Sciences,* 36**,** 25-34.

Hamilton, W., Ying, Z. & Leskovec, J. 2017. Inductive representation learning on large graphs. *Advances in neural information processing systems,* 30.

Hamsheen, E. S. & Flah, L. R. 2023. Improvement performance by using Machine learning algorithms for fake news detection. *Zanco Journal of Pure and Applied Sciences,* 35**,** 48-57.

He, X., Deng, K., Wang, X., Li, Y., Zhang, Y. & Wang, M. Lightgcn: Simplifying and powering graph convolution network for recommendation. Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, 2020. 639-648.

Hu, Z., Dong, Y., Wang, K. & Sun, Y. Heterogeneous graph transformer. Proceedings of the web conference 2020, 2020. 2704-2710.

Koren, Y. Factorization meets the neighborhood: a multifaceted collaborative filtering model. Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining, 2008. 426-434.

Koren, Y., Bell, R. & Volinsky, C. 2009. Matrix factorization techniques for recommender systems. *Computer,* 42**,** 30-37.

Koren, Y., Rendle, S. & Bell, R. 2021. Advances in collaborative filtering. *Recommender systems handbook***,** 91-142.

Lee, D. & Seung, H. S. 2000. Algorithms for non-negative matrix factorization. *Advances in neural information processing systems,* 13.

Liang, S., Sun, R., Lee, J. D. & Srikant, R. 2018. Adding one neuron can eliminate all bad local minima. *Advances in Neural Information Processing Systems,* 31.

Lin, Z., Tian, C., Hou, Y. & Zhao, W. X. Improving graph collaborative filtering with neighborhood-enriched contrastive learning. Proceedings of the ACM web conference 2022, 2022. 2320-2329.

Wang, X., He, X., Wang, M., Feng, F. & Chua, T.-S. Neural graph collaborative filtering. Proceedings of the 42nd international ACM SIGIR conference on Research and development in Information Retrieval, 2019. 165-174.

Wu, B., Wen, W., Hao, Z. & Cai, R. 2020. Multi-context aware user–item embedding for recommendation. *Neural Networks,* 124**,** 86-94.

Wu, S., Sun, F., Zhang, W., Xie, X. & Cui, B. 2022. Graph neural networks in recommender systems: a survey. *ACM Computing Surveys,* 55**,** 1-37.

Xiao, J., Ye, H., He, X., Zhang, H., Wu, F. & Chua, T.-S. 2017. Attentional factorization machines: Learning the weight of feature interactions via attention networks. *arXiv preprint arXiv:1708.04617*.

Xu, B., Wang, N., Chen, T. & Li, M. 2015. Empirical evaluation of rectified activations in convolutional network. *arXiv preprint arXiv:1505.00853*.

Yang, D., Chen, L., Liang, J., Xiao, Y. & Wang, W. Social tag embedding for the recommendation with sparse user-item interactions. 2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM), 2018. IEEE, 127-134.

Yang, X., Yan, M., Pan, S., Ye, X. & Fan, D. Simple and efficient heterogeneous graph neural network. Proceedings of the AAAI conference on artificial intelligence, 2023. 10816-10824.

Ying, R., He, R., Chen, K., Eksombatchai, P., Hamilton, W. L. & Leskovec, J. Graph convolutional neural networks for web-scale recommender systems. Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining, 2018. 974-983.

Yu, J., Xia, X., Chen, T., Cui, L., Hung, N. Q. V. & Yin, H. 2023. XSimGCL: Towards extremely simple graph contrastive learning for recommendation. *IEEE Transactions on Knowledge and Data Engineering,* 36**,** 913-926.

Yu, J., Yin, H., Xia, X., Chen, T., Cui, L. & Nguyen, Q. V. H. Are graph augmentations necessary? simple graph contrastive learning for recommendation. Proceedings of the 45th international ACM SIGIR conference on research and development in information retrieval, 2022. 1294-1303.

Zhu, Q., Yang, C., Xu, Y., Wang, H., Zhang, C. & Han, J. 2021. Transfer learning of graph neural networks with ego-graph information maximization. *Advances in Neural Information Processing Systems,* 34**,** 1766-1779.