Interest-aware Collaborative filtering Recommendation Model based on Graph Neural Networks

Rohan Joshi (RohanJoshi9105.bfac@hotmail.com)
University of Bradford
Shreya Patel
University of Bradford
Sybilla Smith
University of Bradford
Humphrey St.Dawson
University of Bradford

Research Article

Keywords: Recommendation Systems, Collaborative Filtering, Knowledge Graphs, Graph Convolutional Neural Networks, Infromation.

Posted Date: April 26th, 2023

DOI: https://doi.org/10.21203/rs.3.rs-2859276/v1

Additional Declarations:
Competing interests: The authors declare no competing interests.

Interest-aware Collaborative filtering Recommendation Model based on Graph Neural Networks

Rohan Joshi, Shreya Patel, Sybilla Smith,Humphrey St.Dawson,

Abstract—Recommendation systems have become an integral part of various online platforms, providing personalized recommendations to users while reducing information overload. Collaborative filtering-based models have been effective in learning user and item representations by modeling historical user-item interactions. However, these models often suffer from sparsity and cold-start problems, which limit their performance. To alleviate these issues, researchers have designed sophisticated algorithms that incorporate user and item attributes, which have proven to be effective in modeling. Recent studies have utilized item attributes and knowledge graphs (KGs) to represent interconnected attributes, providing rich auxiliary information for recommendation systems. Graph Convolution Neural Network (GCN) based models have become the most popular research method in recommendation systems. These models have the ability to learn embedding representations on graph structures, which makes them effective in modeling recommendation systems. However, the GCN model suffers from the oversmoothing problem, which makes node representations indistinguishable after multiple layers of graph convolution. Researchers have proposed models such as LR-GCN, Light-GCN, DropEdge, and IMP-GCN to mitigate this problem. Despite the significant advancements in GCN-based models, they still do not take full advantage of higher-order neighbors' covariance signals. Additionally, they do not consider the influence of users' different opinions on the recommendation results when propagating and aggregating synergistic signals. This paper proposes a graph convolutional neural network recommendation model with interest-aware message delivery that utilizes 2nd-order neighbors' cooperative signals and users' views to improve accuracy in learning node embeddings. Our model separates subgraphs for users and products, allowing for intergroup crossover in the classification. The proposed model addresses the shortcomings of current GCN-based models, and it is evaluated on several datasets. The results show that our model outperforms state-of-the-art models in terms of prediction accuracy, making it a promising recommendation model for various applications.

Index Terms—Recommendation Systems, Collaborative Filtering, Knowledge Graphs, Graph Convolutional Neural Networks, Infromation.

1 Introduction

With the advancement of Internet technology, people have access to a large amount of online content, such as news, movies, and various products. However, the explosion of online information often overwhelms users, and recommendation systems are an effective information filtering tool that reduces information overload while providing a good user experience. Recommendation systems have become one of the key technologies in various online platforms whose purpose is to predict whether a user will interact with a project or not [1]. Among them, collaborative filtering-based models have made effective progress in learning user and item representations by modeling historical user-item interactions [2, 3]. Collaborative filtering algorithm is a classical recommendation technique that represents users as well as items in vector form and models their interactions by specific operations (inner product, neural networks [4], etc.). However, collaborative filtering algorithms usually suffer from sparsity and cold-start problems. To alleviate the sparsity and cold-start problems of collaborative filtering-based recommender systems, researchers usually collect user and item attributes and design sophisticated algorithms to exploit this additional information [5].

Several recent studies [6, 7, 8, 9, 10] have used item attributes for modeling, and the results of these studies show that attributes are not isolated but are interconnected, one form of which is the Knowledge Graph (KG). KG is a typical heterogeneous network in which each node represents an entity, which can be either an item or an attribute, and two related entities are connected by an edge. KG has rich semantic association information, which can provide rich auxiliary information for recommendation systems. GCN (Graph Convolution Neural Net work) [11] based models have been very effective in recommendation due to their powerful ability to learn embedding representations on graph structures. The GCN model is the most popular research method in recommendation systems today [12, 13, 14, 15, 16]. The main function of the GCN model is to improve the embedding representation of users and items by iteratively aggregating feature information from neighbors using additional information extracted from the connectivity of the graph. NGCF [12] exploits the higher-order connectivity of graphs to alleviate the sparsity problem in recommender systems, however, GCN suffers from the over-smoothing problem because the graph convolution

*Rohan Joshi is the corresponding author.

- *Rohan Joshi, Shreya Patel, Sybilla Smith, and Humphrey St.Dawson are with University of Bradford, UK. (e-mail: RohanJoshi9105.bfac@hotmail.com).*

operation can actually be considered as a special kind of graph Laplacian smoothing, which makes the node representation indistinguishable after multiple layers of graph convolution [17]. To address these problems, Chen proposed the LR-GCN model to mitigate the oversmoothing effect by adding residual operations. He et al. [13] further pointed out that the feature transformation and nonlinear activation in the original GCN model have little positive impact on the final performance, and even more negative impact with training. Based on this, the proposed Light-GCN model only retains the neighborhood aggregation operation and removes the "self-loop" in the aggregation operation to alleviate the over-smoothing problem. The proposed Light-GCN model only retains the neighborhood aggregation operation and removes the "self-loop" from the aggregation operation to mitigate the over-smoothing problem. To reduce the propagation of negative information, DropEdge [18] randomly removes a certain number of edges from the input graph in each training cycle, and theoretically demonstrates that DropEdge either reduces the convergence speed of oversmoothing or mitigates the information loss caused by oversmoothing. The IMP-GCN [19] model proposed in the same period considers that not all information from higher-order neighbors is positive in reality, and by randomly dividing users into two subgraphs for learning, the associated edges of users between the two subgraphs are cut off, thus filtering out the propagation of negative information in higher-order graph convolution operations.

Although graph convolutional neural networks have been recognized to have significant advantages in recommendation tasks, the proposed GCN-based recommendation models usually ignore the some problems. The graph convolution layers of most models do not take full advantage of the covariance signals of higher-order neighbors. The GraphRec model proposed in 2019 directly aggregates the synergy signals of friends using social relationships [20], but this approach does not exploit the higher-order signals of items and requires additional auxiliary information. The graphical convolution layer of the existing model pays little attention to the influence of users' different opinions on the recommendation results when propagating and aggregating synergistic signals. In the recommendation scenario, the ratings represent the users' opinions. However, most of the existing models ignore the difference in user's viewpoints when aggregating 1st-order synoptic signals. We note that the models proposed by GCMC and the literature [11] try to aggregate the synergistic signals at different rating levels separately, but their methods still do not take into account the differences in user perspectives when combining signals from multiple levels. To address these shortcomings, this paper proposes a graph convolutional neural network recommendation model with interest-aware message delivery for the rating prediction task in the recommendation system, with separate subgraphs for users and products, and intergroup crossover in the classification. The study uses the 2nd-order neighbors' cooperative signals and users' views to improve the accuracy of the graph convolutional neural network in learning node embeddings, which is used to update the node embeddings in the graph convolutional layer. The performance advantage is validated on several recommendation datasets. The performance advantage is validated on several recommendation datasets.

2 Related Work

2.1 Graph neural networks

Graph neural networks aim to apply neural networks to data in non-Euclidean spaces for feature learning. The graph convolutional neural network (GCN) was proposed by Kpif in 2017, which provides a novel idea for processing graph-structured data by applying convolutional neural networks (CCNs) to graph data. Graph neural networks have been widely used in recent years to handle graph-structured data, such as social networks, citation networks, and knowledge graphs. One of the key challenges in these applications is to learn the representation of each node that captures both its structural and semantic information.

In early studies based on graph data, researchers conducted learning by iteratively propagating neighbor information through recurrent neural networks. However, this process requires a large computational effort, which is a major bottleneck for large-scale graph data analysis. To address this issue, many researchers have proposed various techniques to alleviate the computational burden of graph neural networks. One of the most promising approaches is to use network embeddings [20, 21, 22], which project each node in a graph into a low-dimensional vector space while preserving the network topology and node information. This produces a low-dimensional vector representation of the nodes [23], which can be used for various graph data analysis tasks such as classification, recommendation, and clustering. Network embedding algorithms can be broadly classified into three categories: matrix decomposition, random wandering, and deep learning methods [24, 25]. Matrix decomposition methods, such as matrix factorization and singular value decomposition, aim to factorize the adjacency matrix of a graph into low-rank matrices that can be used as node embeddings. Random walking methods, such as DeepWalk and node2vec, generate node embeddings by simulating random walks on a graph and treating the sequence of visited nodes as a sentence for word embedding learning. Deep learning methods, such as graph convolutional networks and graph attention networks, directly learn node embeddings by propagating information through the graph structure. In recent years, deep learning methods for network embedding have gained increasing attention due to their superior performance and scalability. These methods can be further categorized into two groups: graph self-encoders and graph convolutional neural networks with unsupervised learning. Graph self-encoders, such as SDNE and DNGR, use autoencoder architectures to learn the non-linear mappings between the high-dimensional input space and the low-dimensional embedding space. Graph convolutional neural networks, such as GraphSage and GAT, apply convolutional operations on the graph structure to aggregate the information from the node's local neighborhood and learn the node embedding in a supervised or unsupervised manner. Apart from network

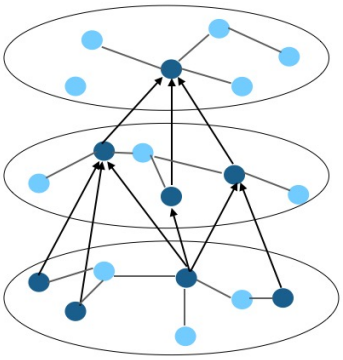

Fig. 1: Neighbor node feature fusion.

embedding, another promising approach for graph data analysis is matrix completion, which aims to predict missing values in a partially observed matrix. In recommender systems, matrix completion can be used to predict user-item ratings based on the observed ratings. GCMC [11] is an example of a graph-based auto-encoder framework for matrix completion, which utilizes message passing on a bipartite interaction graph to produce latent features of user and item nodes. These latent representations are then used to reconstruct rating links through a bilinear decoder. This approach is particularly advantageous when external structured information such as social networks is available, as it can help alleviate the cold start problem and improve performance.

2.2 Knowledge graph based Recommendation

The use of knowledge graphs (KGs) in recommendation systems has gained considerable attention in recent years due to their ability to provide rich semantic information about items and users. One of the most common approaches to incorporating KGs into recommendation systems is through graph embedding methods, which aim to learn low-dimensional representations of entities in the KG while preserving their structural and semantic properties. However, these methods are typically based on two separate stages: KG embedding and recommendation, which can limit the effectiveness of the final recommendation results. To address this limitation, the authors propose an end-to-end graph neural network recommendation model called MKR [26], which integrates the KG embedding and recommendation stages into a single framework. MKR consists of two main modules: a recommendation system module and a knowledge graph entity feature learning module. These two modules are bridged by a cross-compression module that automatically learns the higher-order feature interactions of items and entities in both tasks and shares feature information by sequential training. By optimizing the graph representation learning based on the feedback of the recommendation results, MKR is able to achieve better performance than existing methods that rely on separate KG embedding and recommendation stages.

Another recent approach to KG-aware recommendation systems is RippleNet [27], which aims to capture the user's interests and preferences through the propagation of information in the KG. The core idea of RippleNet is interest propagation, where user history is used as a seed, and then user interests are iteratively extended outward through connections in KG to obtain a user vector for the downstream prediction task. This approach enables RippleNet to effectively capture the complex relationships between users and items in the KG, and has been shown to outperform other state-of-the-art methods on several benchmark datasets. The use of KGs in recommendation systems is a promising research direction, and the development of end-to-end graph neural network models such as MKR and interest propagation methods like RippleNet are likely to have a significant impact on the field in the coming years.

3 Methodology

3.1 Problem definition

In a typical recommendation scenario, there will be a set $U = \{u_1, u_2, \ldots, u_M\}$ containing M users and a set $V = \{v_1, v_2, \ldots, v_N\}$ containing N items. The user-object interaction matrix is $Y \in R^{M \times N}$, which reflects the implicit feedback from the user, when $y = 1$ means that user u has a connection with object v, such as clicking, browsing or purchasing, and vice versa $y_{uv} = 0$. In addition, there is a knowledge graph G, which consists of the entity-relationship-entity triple (h, r, t). Here $h \in E, r \in R$ and $t \in E$ denote the head, relationship and tail of the triad, and E and R are the sets of entities and relationships in the knowledge graph, respectively. Some GCN-based recommendation models implicitly transmit higher-order covariance signals by stacking multiple graph convolution layers, but it is difficult to control the strength of the higher-order signals in this way and is susceptible to noise interference. For example, suppose there is a connected path $u_0 \rightarrow v_1 \rightarrow u_2 \rightarrow v_3 \rightarrow u_4$ on the bipartite graph, where user u_2 rates item v_3 by chance, and although it is not similar to the preference of target user u_0, this rating operation will still result in the implicit transmission of u_4's synergistic signal to u0 along the stacked four graph convolution layers, instead of being suppressed or eliminated as a noisy signal.

3.2 Information propagation

The general idea is to randomly divide the users into n groups, and the items associated with those users are also part of this subgraph. Then, for the original input, the items are randomly divided into n groups, and the users associated with those items also belong to this subgraph. Given that the direct neighbors between users and items provide the most interesting and important information for users, in particular, all first-order neighbor nodes are used for first-order propagation. Using $e_u^{(0)}$ to denote the embedding of user u at user level 0 and $e_i^{(0)}$ to denote the embedding of item i at user classification level 0, the convolution operation is as follows.

$$e_u^{(1)} = \sum_{i \in N_u} \frac{1}{\sqrt{|N_u|}\sqrt{|N_i|}} e_i^{(0)} \tag{1}$$

$$e_i^{(1)} = \sum_{u \in N_i} \frac{1}{\sqrt{|N_i|}\sqrt{|N_u|}} e_u^{(0)} \tag{2}$$

where $e_u^{(1)}$ denotes the node representation of user u at user classification level 1 denotes the node representation of user

u at level 1 of the user classification, and $e_i^{(1)}$ denotes the node embedding of items at level 1 of the user classification.

For higher-order graph convolution, in order to reduce the effect of noise, nodes in a subgraph can only use the information of nodes in that subgraph, because all direct nodes corresponding to a user belong to the same subgraph at the time of classification, so the user node can receive the information of all accessible nodes. In contrast, all direct nodes corresponding to an item may be in different subgraphs, and the item node can only receive information about the reachable nodes in all subgraphs. Using $e_i^{(k)}$ to represent the embedding representation of item i in the subgraph G_{sU} after k layers of convolution in the user classification approach, the higher-order propagation can be defined as follows.

$$e_u^{(k)} = \sum_{i \in N_u} \frac{1}{\sqrt{|N_u|}\sqrt{|N_i|}} e_i^{(k-1)} \tag{3}$$

$$e_i^{(k)} = \sum_{u \in N_i} \frac{1}{\sqrt{|N_i|}\sqrt{|N_u|}} e_u^{(k-1)} \tag{4}$$

The n embedding representations are obtained by learning, and the embedding representations obtained at each layer are combined to form the final embedding representation of user u and item i. For the user classification approach e_{is} can be viewed as an in-group feature embedding learned from the subgraph G_{Us}.

$$e_i^{(k)} = \sum_{s \in S_u} e_u^{(k)} \tag{5}$$

where S_u is the subgraph to which item i belongs.

The model propagation algorithm is implemented using a matrix propagation mechanism, which allows updating all user and item embedding representations simultaneously. e(0) is the user and item node embedding representation matrix, e(k) represents the node representation after k iterations, and EsU(k) and EsI(k) represent the item representation of the user classification-item classification subgraph after k iterations. The first layer of embedding propagation can be described as follows.

$$E^{(1)} = LE^{(0)} \tag{6}$$

where L is the Laplace matrix of the user-project relationship matrix. When the propagation of subgraphs is involved in the higher-order graph convolution layer, the subgraph embedding propagation process can be described in a user classification manner as follows.

$$E_{sU}^{(k-1)} = L_{sU} E_{sU}^{(k-2)} \tag{7}$$

where $k \geq 2$ and L represents the Laplacian matrix of the subgraph E. Then continue to propagate the embedding of $(k-1)$ layers to form the k-layer embedding.

$$E_{sU}^{(k)} = LE_{sU}^{(k-1)} \tag{8}$$

All k-layer embeddings involving different subgraphs are aggregated to form the final k-layer embedding, and the items are classified in the same way.

$$E_U^{(k)} = \sum_{s \in G_{sU}} E_{sU}^k \tag{9}$$

$$E_I^{(k)} = \sum_{s \in G_{sI}} E_{sI}^k \tag{10}$$

3.3 Constructing collaborative signals

Aggregating higher-order co-signals in the graph convolution layer is beneficial for learning the node embedding representation. However, the model in this paper only uses the co-signals of the 2nd-order neighbors among all the higher-order neighbors. The 2nd-order neighbors are more similar to the target entity in terms of preferences than higher-order neighbors, and the more interacting entities they have in common with the target entity, the higher the similarity. We give a method to compute the 2-nd order collabrative signal and its strength so that u_j is a 2-nd order neighbor of the target user u_i on graph $\mathbf{G}$ and $u_i \rightarrow v_k \rightarrow u_j$ is an arbitrary connectivity path of length 2 between them, then the 2-nd order collabrative signal from u_j to u_i can be defined as follows.

$$s_{i \rightarrow j}^l = \sum_{k \in N_i \cap N_j} p_{i,k,j}(e_{u,j}^{(l-1)} + (e_{u,i}^{(l-1)} \odot (e_{u,j}^{(l-1)}) \tag{11}$$

where,$e_{u,j}^{(l-1)}$ is the embedding of user u_j at layer l-1. $\odot$ is the Hadamard product by element. $N_i \cap N_j$ denotes the set of items jointly intersected by u_i and u_j. $p_{i,k,j}$ is the 2nd order covariance signal strength coefficient of the graph convolutional nerve network, which is related to the node degree on the path. Suppose that user u_j and target user u_i do not have similar preferences, but become the second-order neighbors of u_i by a chance evaluation operation. However, the signal enhancement of the Hadamard product term is weak due to the dissimilarity of their node embeddings, and given the very small number of their common interaction items $N_i \cap N_j$, the noise synergistic signal propagated by u_j to the target user u_i is also very limited. Let $E^{(l-1)} \in R^{(m+n)\times h}$ be the matrix of all embedded vectors in layer l - 1. $S^{l(2)} \in R^{(m+n)\times h}$ be the 2nd order co-signal matrix of all nodes in layer l, whose row vectors are composed of $s_{u,i}^{l(2)}$ or $s_{v,i}^{l(2)}$.

$$S^{l(2)} = \overline{L}^{(2)} E^{(l-1)} + \overline{L}^{l(2)} E^{(l-1)} \odot E^{(l-1)} \tag{12}$$

where $\overline{L}^{(2)}$ is the de-diagonalized canonical 2nd order adjacency matrix. And $L^{(2)}$ is the normalized 2nd order adjacency matrix, which measures the 2nd order signal strength based on the node degree, and is the signal strength coefficient $p_{i,k,j}$ matrix form, that is:

$$L^{(2)} = (D^{-\frac{1}{2}} \cdot A \cdot D^{-\frac{1}{2}})^2 \tag{13}$$

3.4 Node embedding update

We pool the 1st and 2nd order collaborative signals and the nodes' previous layer embedding, and update the node embedding vector in the lth layer, as follows.

$$E^l = ReLU((E^{(l-1)} \oplus S^{l(1)} \oplus S^{l(2)})W_2^l) \tag{14}$$

where $ReLU()$ is the excitation function and operator $\oplus$ indicates the matrix concatenation operation.

3.5 Prediction

The nodes of the final learning of the graphical convolutional layer are embedded in e_u^L and e_v^L and fed

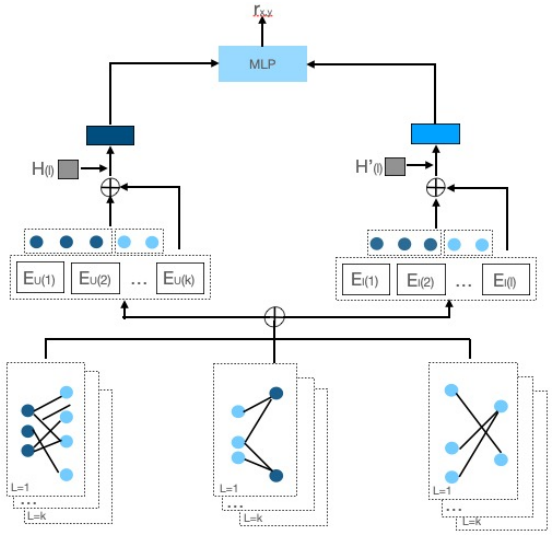

Fig. 2: Schematic diagram of our model.

into a MLP network to predict the unknown ratings $\hat{r}_{u,v}$ between user u and item v.

$$\hat{r}_{u,v} = W_4(ReLU(W_3(e_u^L \oplus e_v^L))) \tag{15}$$

where $W_3 \in R^{d\times 1}$, $W_4 \in R^{d\times 1}$ is the MLP weight matrix to be learned. d is the number of neurons in first MLP layer.

4 EXPERIMENTS

4.1 Dataset

ML-100K,ML-1M,ML-10M are movie recommendation datasets of different sizes from MovieLens website. Table 1 summarizes the data sets used for the experiments. In the preprocessing stage, we encode the original entity features of users and items directly on the bipartite graph using auxiliary information and graph structure features . The entity features of the project consist of three parts: node degree, scoring feature, and semantic feature.

4.2 Experiment setting

The number of convolutional layers $L = 3$, the regular coefficients λ are selected by cross-validation in the range of $\{0.01, 0.05, 0.1, 0.5, 1, 5, 10\}$, and the hidden layer parameter $d = 16$ for MLP. The learning rate of Adam's optimizer is set to 10-3 during training, the maximum number of iterations is 1000, the size of each batch of training data is set to 1024, and the node dropout probability $\eta = 0.3$. The parameters of the comparison algorithm are basically set according to the original text. In addition, the data set is divided into training set, validation set and test set in the ratio of 0.8, 0.1 and 0.1. The model parameters are selected in the validation set and the root mean square error (RMSE) is used to evaluate the prediction error of each model in the test set.

In our experiments, we select four mainstream recommendation models as comparison algorithms, including BPR, GCMC, GraphRec, and NGCF.

BPR: We use Bayesian Personalized Ranking based Matrix Factorization. BPR introduces a pair-wise loss into the Matrix Factorization to be optimized for ranking.

NGCF: Neural Collaborative Filtering fuses matrix factorization and Multi-Layer Perceptron (MLP) to learn from user-item interactions. The MLP endows NCF with the ability of modelling non-linearities between users and items.

GCMC: Graph Convolutional Matrix Completion utilizes a graph auto-encoder to learn the connectivity information of a bipartite interaction graph for latent factors of users and items.

GraphRec: this model uses GNNs to integrate node information and topology for consistent modeling of user-project graphs and user-user social graphs.

4.3 Result Analysis

The prediction error of the four recommendation models based on graph convolutional neural networks is significantly better than that of the NeuCF model based on the moment matrix decomposition. This means that the former can learn the embedded representation of nodes more accurately and obtain better prediction results by integrating the solid features of nodes and graph structure features on the topological graph. GCMC combines the cooperative signals of 1st-order neighbors and has insufficient learning ability for node embedding, so it performs the worst among the four GCN-based recommendation models. Both GraphRec and NGCF models utilize higher-order covariance signals and achieve better prediction results than GCMC. The difference is that GraphRec directly aggregates the covariance signals of social friends, which are equivalent to similar higher-order neighbors in a bipartite graph. NGCF, on the other hand, implicitly propagates and aggregates higher-order synergistic signals by stacking multiple graph convolution layers. This shows that aggregating higher-order collaborative signals can help improve the performance of the recommendation model. The model in this paper achieves the lowest prediction error for most of the data sets. The experimental results of this model significantly outperform other models with 95% confidence in most cases. The superior performance of the present model can be attributed to its increased use of higher-order synergistic signals in the graph convolution layer and the introduction of user perspectives into the signal aggregation process. In contrast, GraphRec only obtains the higher-order synergistic signals of users directly from the social relationships, without considering the higher-order signals of the items. The NGCF model does not explicitly utilize higher-order synergy signals within the graph convolution layer and does not consider the influence of user perspectives.

5 CONCLUSION AND FUTURE WORK

The GCN model is the most popular research method in recommendation systems today. The main function of the GCN model is to improve the embedding representation of users and items by iteratively aggregating feature information from neighbors using additional information extracted from the connectivity of the graph. NGCF exploits the higher-order connectivity of graphs to alleviate the

TABLE 1: Statistics of Experimental Datasets.

Dataset	Users	Items	Rating	Rating density	User feature	Item feature
ML-100K	943	1628	100000	0.0630	329	326
ML-1M	6040	3706	1000209	0.0447	329	326
ML-10M	69878	10677	10000054	0.0134	306	327

TABLE 2: Comparison of performance of diiferent model.

Model	ML-100K	ML-1M	ML-10M
BPR	0.913	0.854	0.768
GCMC	0.903	0.846	0.757
GraphRec	0.887	0.831	0.742
NGCF	0.875	0.827	0.736
OURS	0.866	0.821	0.724

sparsity problem in recommender systems, however, GCN suffers from the over-smoothing problem because the graph convolution operation can actually be considered as a special kind of graph Laplacian smoothing, which makes the node representation indistinguishable after multiple layers of graph convolution. To address these problems, Chen proposed the LR-GCN model to mitigate the oversmoothing effect by adding residual operations. He et al. further pointed out that the feature transformation and nonlinear activation in the original GCN model have little positive impact on the final performance, and even more negative impact with training. Based on this, the proposed Light-GCN model only retains the neighborhood aggregation operation and removes the "self-loop" in the aggregation operation to alleviate the over-smoothing problem. The proposed Light-GCN model only retains the neighborhood aggregation operation and removes the "self-loop" from the aggregation operation to mitigate the over-smoothing problem. To reduce the propagation of negative information, DropEdge randomly removes a certain number of edges from the input graph in each training cycle, and theoretically demonstrates that DropEdge either reduces the convergence speed of oversmoothing or mitigates the information loss caused by oversmoothing.

Although graph convolutional neural networks have been recognized to have significant advantages in recommendation tasks, the proposed GCN-based recommendation models usually ignore the some problems. The graph convolution layers of most models do not take full advantage of the covariance signals of higher-order neighbors. The GraphRec model proposed in 2019 directly aggregates the synergy signals of friends using social relationships [20], but this approach does not exploit the higher-order signals of items and requires additional auxiliary information. The graphical convolution layer of the existing model pays little attention to the influence of users' different opinions on the recommendation results when propagating and aggregating synergistic signals. In the recommendation scenario, the ratings represent the users' opinions. However, most of the existing models ignore the difference in user's viewpoints when aggregating 1st-order synoptic signals. We note that the models proposed by GCMC and the literature try to aggregate the synergistic signals at different rating levels separately, but their methods still do not take into account the differences in user perspectives when combining signals from multiple levels. To address these shortcomings, this paper proposes a graph convolutional neural network recommendation model with interest-aware message delivery for the rating prediction task in the recommendation system, with separate subgraphs for users and products, and intergroup crossover in the classification. The study uses the 2nd-order neighbors' cooperative signals and users' views to improve the accuracy of the graph convolutional neural network in learning node embeddings, which is used to update the node embeddings in the graph convolutional layer. The performance advantage is validated on several recommendation datasets. The performance advantage is validated on several recommendation datasets.

REFERENCES

[1] S. Rendle, Z. Gantner, C. Freudenthaler, and L. Schmidt-Thieme, "Fast context-aware recommendations with factorization machines," in *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, 2011, pp. 635–644.

[2] P. Covington, J. Adams, and E. Sargin, "Deep neural networks for youtube recommendations," in *Proceedings of the 10th ACM conference on recommender systems*, 2016, pp. 191–198.

[3] 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.

[4] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural collaborative filtering," in *Proceedings of the 26th international conference on world wide web*, 2017, pp. 173–182.

[5] H.-T. Cheng, L. Koc, J. Harmsen, T. Shaked, T. Chandra, H. Aradhye, G. Anderson, G. Corrado, W. Chai, M. Ispir *et al.*, "Wide & deep learning for recommender systems," in *Proceedings of the 1st workshop on deep learning for recommender systems*, 2016, pp. 7–10.

[6] J. Huang, W. X. Zhao, H. Dou, J.-R. Wen, and E. Y. Chang, "Improving sequential recommendation with knowledge-enhanced memory networks," in *The 41st international ACM SIGIR conference on research & development in information retrieval*, 2018, pp. 505–514.

[7] H. Wang, F. Zhang, X. Xie, and M. Guo, "Dkn: Deep knowledge-aware network for news recommendation," in *Proceedings of the 2018 world wide web conference*, 2018, pp. 1835–1844.

[8] F. Zhang, N. J. Yuan, D. Lian, X. Xie, and W.-Y. Ma, "Collaborative knowledge base embedding for recommender systems," in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 2016, pp. 353–362.

[9] X. Yu, X. Ren, Y. Sun, Q. Gu, B. Sturt, U. Khandelwal, B. Norick, and J. Han, "Personalized entity recommendation: A heterogeneous information network approach," in *Proceedings of the 7th ACM international conference on Web search and data mining*, 2014, pp. 283–292.

[10] Q. Wang, Y. Wei, J. Yin, J. Wu, X. Song, and L. Nie, "Dualgnn: Dual graph neural network for multimedia recommendation," *IEEE Transactions on Multimedia*, 2021.

[11] R. v. d. Berg, T. N. Kipf, and M. Welling, "Graph convolutional matrix completion," *arXiv preprint arXiv:1706.02263*, 2017.

[12] 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.

[13] L. Chen, L. Wu, R. Hong, K. Zhang, and M. Wang, "Revisiting graph based collaborative filtering: A linear residual graph convolutional network approach," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, no. 01, 2020, pp. 27–34.

[14] 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.
[15] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
[16] Y. Wei, X. Wang, L. Nie, X. He, R. Hong, and T.-S. Chua, "Mmgcn: Multi-modal graph convolution network for personalized recommendation of micro-video," in *Proceedings of the 27th ACM International Conference on Multimedia*, 2019, pp. 1437–1445.
[17] Y. Wei, X. Wang, L. Nie, X. He, and T.-S. Chua, "Graph-refined convolutional network for multimedia recommendation with implicit feedback," in *Proceedings of the 28th ACM International Conference on Multimedia*, 2020, pp. 3541–3549.
[18] Y. Rong, W. Huang, T. Xu, and J. Huang, "The truly deep graph convolutional networks for node classification," *arXiv preprint arXiv:1907.10903*, vol. 5, 2019.
[19] F. Liu, Z. Cheng, L. Zhu, Z. Gao, and L. Nie, "Interest-aware message-passing gcn for recommendation," in *Proceedings of the Web Conference 2021*, 2021, pp. 1296–1305.
[20] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, 2009.
[21] H.-J. Xue, X. Dai, J. Zhang, S. Huang, and J. Chen, "Deep matrix factorization models for recommender systems." in *IJCAI*, vol. 17. Melbourne, Australia, 2017, pp. 3203–3209.
[22] Y. Wei, X. Wang, W. Guan, L. Nie, Z. Lin, and B. Chen, "Neural multimodal cooperative learning toward micro-video understanding," *IEEE Transactions on Image Processing*, vol. 29, pp. 1–14, 2019.
[23] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: Online learning of social representations," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2014, pp. 701–710.
[24] D. Wang, P. Cui, and W. Zhu, "Structural deep network embedding," in *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, 2016, pp. 1225–1234.
[25] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," *Advances in neural information processing systems*, vol. 30, 2017.
[26] H. Wang, F. Zhang, M. Zhao, W. Li, X. Xie, and M. Guo, "Multi-task feature learning for knowledge graph enhanced recommendation," in *The world wide web conference*, 2019, pp. 2000–2010.
[27] H. Wang, F. Zhang, J. Wang, M. Zhao, W. Li, X. Xie, and M. Guo, "Ripplenet: Propagating user preferences on the knowledge graph for recommender systems," in *Proceedings of the 27th ACM international conference on information and knowledge management*, 2018, pp. 417–426.