IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received January 27, 2021, accepted February 14, 2021, date of publication February 24, 2021, date of current version March 4, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3061915

Light Graph Convolutional Collaborative Filtering With Multi-Aspect Information

DENGHUA MEI, NIU HUANG, AND XIN LI
School of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, China
Corresponding author: Niu Huang (cshn@mail.scut.edu.cn)

This work was supported in part by the Public Research and Capacity Building of Guangdong Province under Grant 2014B010104001, and in part by the Basic and Applied Basic Research of Guangdong Province under Grant 2015A03030 8018.

ABSTRACT The personalized recommendation has become increasingly prevalent in real-world applications, to help users in discovering items of interest. Graph Convolutional Network (GCN) has achieved great success and become a new state-of-the-art for collaborative filtering. However, most of the existing GCN based methods can only capture information about the user's purchase (or click) history, reflecting only one aspect of the user preferences and item characteristics. To provide more accurate recommendations, we need to go beyond modeling user-item interactions and take auxiliary information into consideration. In this paper, we propose a Light GCN based Aspect-level Collaborative Filtering model (LGC-ACF) to exploit multi-aspect user-item interaction information. First, we construct aspect-level user-item interaction graphs according to the interaction history and the knowledge information of items, then feed them to a delicately designed Light GCN based model to learn aspect-level representations of users and items. Finally, the representations of all aspects and all propagation layers are fused for recommendation. We apply LGC-ACF to three datasets: Movielens, Amazon, and Taobao. The experiment results show that LGC-ACF achieves average NDCG improvements of 5.31%, 4.06%, 14.9% in Movielens, Amazon, and Taobao datasets, respectively, compared with state-of-the-art baselines for recommendation.

INDEX TERMS Recommender systems, graph convolutional network, representation learning, multi-aspect information.

I. INTRODUCTION

Recommendation systems are information filtering systems designed to alleviate information overload [1], which can provide a set of items according to observed user behaviors. It has been extensively used in almost any field where information needs to be filtered, such as movies [2], music [3], micro-videos [4], news [5], [6]. As one of the most prevailing recommendation methods, the basic assumption of Collaborative Filtering (CF) is that people who share similar behaviors have similar preferences [7]–[9]. Extensive studies on CF-based recommenders have been conducted and achieved great success.

One basic task of CF is to learn the representations (a.k.a. embedding) of users and items, then conduct predictions based on the representation vectors [10], [11]. Learning informative representations of users and items is of crucial importance to improving CF. Matrix factorization (MF) factorizes the user-item interaction matrix into two low-rank user-specific and item-specific factors, then remodels user-item interaction with inner product [12]. Many subsequent studies have expanded on MF [13]–[16].

Recently, Graph Convolutional Networks (GCN) [17], [18], which have demonstrated their remarkable ability in graph representation learning, are introduced to recommender systems [19] and get great success. By treating the user-item historical behaviors as a bipartite graph with edges between users and items, CF can be transformed into the edge prediction problem in the graph [9], [19]. By iteratively stacking multiple layers of graph convolution operation (neighbor aggregation and non-linear activation operations), GCN can effectively capture collaborative signals of high-hop neighbors in the embedding propagation process [8], alleviate the data sparsity problem in collaborative filtering to some extent. These GCN-based recommender models show stronger performance compared to traditional models.

Although these methods have achieved great success, one drawback is that they typically only utilize information from

The associate editor coordinating the review of this manuscript and approving it for publication was Zhan Bu.

the user's purchase history. Existing GCN-based models have focused on learning representations of users and items solely through their interaction graph. However, people always consider multi-aspect of the item in real applications before adopting an item, e.g., item category, brand, function, appearance, and other item features. Thereby, we believe that modeling multi-aspect information of user and item can refine user preferences and item characteristics. Fig.1(a) shows a toy example. U_4 is the recommended target user. If only exploit the purchasing history of the user, it is not clear whether I_2 or I_3 should be recommended to U_4. However, if the item brand information is added to consider, we may find that I_3 is a better recommendation to U_4 because the items purchased by user U_4 belong to the same brand B_2 as I_3.

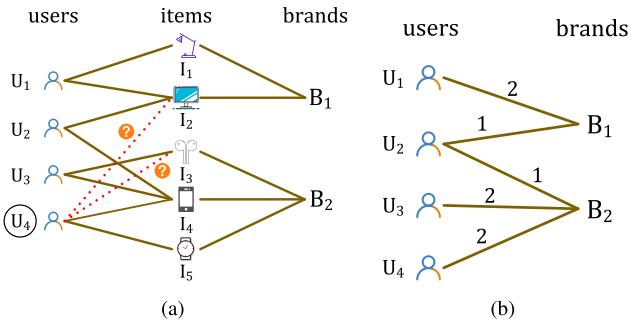


FIGURE 1. (a) A toy example of our idea. (b) User-brand interaction graph.

In this paper, we focus on exploiting information from multiple aspects of user-item interaction. Hence, we devise a novel model, Light GCN based Aspect-level Collaborative Filtering (LGC-ACF), to effectively model and combine multiple aspect-level latent factors and generate informative representations. In particular, our idea is to extract additional information of items by graph neural network, then integrate them into the final representation of user and item to obtain more informative representation. To this end, we imitate the user-item bipartite graph and construct interaction graphs of user with other item attributes, as shown in Fig.1(b), in which the weight of the edge indicates the interaction times between the user and the corresponding item attribute. Then feed these graphs into an elaborately designed multi-component model equipped with light graph convolutional network to learn different aspects and different hop embeddings of user and item. Finally, the aspect-level and layer-level embeddings are fused for recommendation. Extensive experiments on three real-world datasets demonstrate that our LGC-ACF outperforms other state-of-the-art models in recommendation effectiveness.

The major contributions of this work are as follows:

- To make better use of item attributes information that may cause user behavior, we take the lead in modeling the relationship between users and item attributes us-ing graph convolutional network in the interactive graph structure.
- We propose LGC-ACF, a novel Light GCN based approach, which can effectively model and fuse different aspect-level latent factors, which comprehensively represent the user preferences and item characteristics.
- We conduct extensive experiments on three real world datasets, to demonstrate the state-of-the-art performance of our LGC-ACF approach and the effectiveness in improving the embedding quality by fusing multi-aspect information.

II. RELATED WORK

In this section, we briefly introduce existing work on Collaborative Filtering and GCN-based method for recommendation, which are most relevant to our study.

A. COLLABORATIVE FILTERING

CF provides personalized item suggestions to users by learning user and item embeddings from their historical behavior data [7], [8], [14]. The most widely studied collaborative filtering algorithms are based on matrix factorization, which basic assumption is to factorize the interaction matrix (e.g., rating matrix) of users and items into two low-rank matrix, then conducts product between them to reconstruct interaction matrix. PMF [20] assumes that the user's rating of the item is a random variable with Gaussian noise, and optimizes the maximum likelihood function by minimizing the mean square error between the observed value and the predicted value. BiasedMF [12] improves PMF by introducing user-specific, item-specific, and global bias.

With the surge of deep learning techniques, recent studies focus on exploiting deep learning to learn nonlinear interaction features between user and item. For example, NCF [7] uses a nonlinear neural network to learn the interaction function of user and item. DMF [21] takes the interaction history of the user and item as a feature vector and inputs it to a multilayer perceptron to learn the latent expression of user and item. NeuACF [22] introduces aspect-level information on the framework of NCF and designs an attention mechanism for learning the weight of aspect-level information. It should be noted that our method is different from NeuACF, which adopts meta-path to extract similarity matrix of users and items from heterogeneous graph.

B. GCN-BASED METHODS FOR RECOMMENDATION

Different from the traditional collaborative filtering methods, GCN has become a new state-of-the-art for recommendation and has been widely used because it can capture the high-order similarity of nodes in the interaction graph.

GCMC [19] stacks a graph convolution layer followed by a dense layer to accumulate the messages that are aggregated according to different types of edges as the node representations, but it basically only considers the first order neighbors. PinSage [23] is the first application of GCN to a web-scale recommendation system. NGCF [8] iteratively propagates user and item embeddings in the graph to distill high-hop collaborative signals with graph convolutions. KGCN [24]

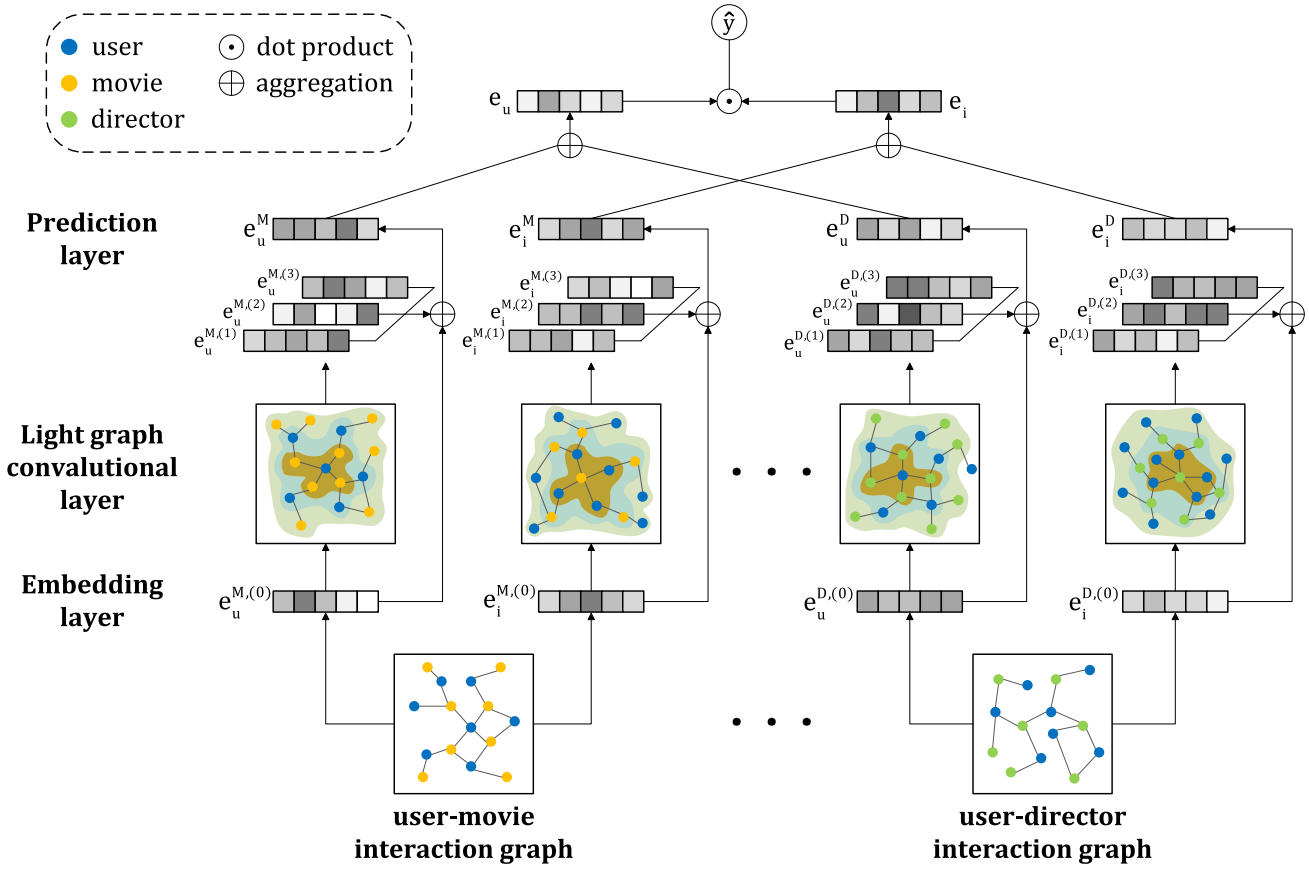


FIGURE 2. Overall architecture of LGC-ACF. It takes in the user-item interaction graphs of different aspects and predicts user-item interaction probabilities. This example is based on the Movielens dataset, the arrowed lines indicate the flow of information. For the description symbols of the embedding vector, the capital letters in the superscript are abbreviations for aspect-level information, e.g., M denotes the movie-aspect and D denotes the director-aspect; the super-index (l), where $l = 1, 2, \ldots, L$ denotes the embedding is obtained at the l-th GCN layer.

and KGAT [25] employ GCN to process heterogeneous information. The former takes the items in the knowledge graph as the center, defines a receptive field based on the number of hops, and then combines the information of the receptive field to get the embedding of the items, while the user embedding is trained separately without using knowledge information, which may limit the performance of the model; The latter encodes user behaviors and item knowledge as a unified relational graph, which is called collaborative knowledge graph. The embeddings of users and items are propagated in the collaborative knowledge graph to inject the item knowledge information. DisenGCN [26], DGCF [27], and MCCF [28] hold that treat a user-item interaction as an isolated data instance is insufficient to capture the diversity of user intents on adopting an item, may resulting in suboptimal representations. LR-GCCF [9] exploits [29] and proposes a residual network structure to tackle the over-smoothing problem. Peng and Mine [30] proposes another method to alleviate the over-smoothing problem, which randomly removes neighbor messages at each propagation layer. LightGCN [11] conducts ablation analyses on GCN and improves the model's performance and scalability by removing the feature transformation and nonlinear activation operation.

III. APPROACH

In this section, we present our model employing the Movielens dataset as an example. Firstly, we briefly describe the model architecture. Then, we introduce each component of the model in detail. Finally, we describe the loss function.

A. OVERALL STRUCTURE OF PROPOSED MODEL

Fig.2 shows the overall framework of LGC-ACF. As we can see, the model takes the user-item interaction graph of different aspects as input and then output the probability of users adopting items in the future. Firstly, an embedding layer provides and initializes the user embeddings about different aspects of the item, as well as the embeddings of different aspects of the item. More specifically, for movies recommendation, in addition to the movie ID, we also encode the genres ID, director ID, actor ID. Please see Fig.3. Note that these embeddings are trainable. Then, we stack multiple light graph convolutional layers to extract the high-hop relationship between users and items. Finally, the prediction layer aggregates the aspect-level and layer-level embeddings to generate the final representation of user and item. The inner product operation is then conducted on them to obtain their affinity score.

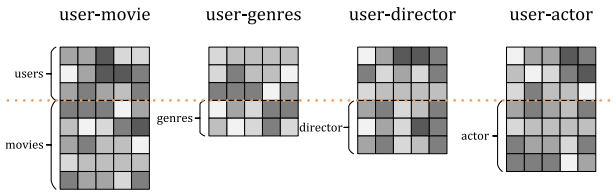


FIGURE 3. Embedding matrix of aspect-level user-item interaction graphs (an example of Movielens dataset).

B. EMBEDDING LAYER

Before generating the embeddings, we need to construct multiple aspect-level user-item interaction graphs. Most of the existing GCN-based recommendation methods explore the bipartite graph of user-item interaction history. In this work, we take advantage of item attributes information to improve the performance of the recommendation. Specifically, by imitating the bipartite graph of user and item, we construct the user-genres, user-director, and user-actor interaction graph, respectively, where the weight of the edge indicates the number of interactions between the user and the corresponding genres (or director, actor). We name these graphs as aspect-level user-item interaction graphs in this paper. For each aspect-level graph, we describe a node with an embedding vector $e \in \mathbb{R}^d$, where d indicates the embedding dimension. This can be thought of as creating a parameter matrix for each aspect-level graph. Taking the user-director graph as an example:

$$E^D = [e^D_{u_1}, \cdots, e^D_{u_N}, e^D_{i_1}, \cdots, e^D_{i_{M_D}}], \tag{1}$$

where superscript D denotes the aspect of director; N denotes the number of users; M_D denotes the number of directors. Other aspect-level embedding matrices are provided similarly. Fig.3 shows an example of Movielens dataset, where each embedding matrix represents the corresponding interaction graph. We adopt Gaussian distribution to initialize all embeddings, and they will be optimized in an end-to-end manner. Table 1 shows examples of more aspects adopted in our experimental datasets.

TABLE 1. Aspect-level information used in experiments.

Dataset	Movielens	Amazon	Taobao
Aspect	movie	product	product
	genres	brand	category
	director	category	-
	actor	-	-

C. LEARNING ASPECT-LEVEL EMBEDDING

This part will refine the aspect-level embedding by propagating them in the corresponding aspect-level interaction graph. A standard GCN layer consists of three operations: feature transformation, neighbor aggregation, and non-linear activation. The propagation rule is defined as:

$$E^{(l+1)} = \sigma(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}E^{(l)}W^{(l)}), \tag{2}$$

where the augmented adjacency matrix $\tilde{A} = A + I$, here I indicates identity matrix and is introduced to add self-connections; $\tilde{D}$ is the degree matrix, where $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$; $W^{(l)}$ denotes layer-specific feature transformation matrix; $\sigma(\cdot)$ denotes non-linear activation function.

Recently, He.*et al.* [11] find that feature transformation and non-linear activation in standard GCN contribute little to the recommendation performance. They propose LightGCN, which only retains the neighborhood aggregation operation in standard GCN. Advanced performance and model simplicity make it more scalable. Thereby we follow their work, employing LightGCN to spread and refine the aspect-level embeddings of user and item. In this work, taking the aspect of user-director for example, the propagation rule of LightGCN can be formulated as:

$$E^{D,(l+1)} = ((D^D)^{-\frac{1}{2}}A^D(D^D)^{-\frac{1}{2}})E^{D,(l)}, \tag{3}$$

where $E^{D,(l+1)} \in \mathbb{R}^{(N+M_D)\times d}$ is the embedding matrix of the user-director graph obtained after $l+1$ steps of embedding propagation; $E^{D,(0)} = E^D$ as the initial representations; A^D is the adjacency matrix of user-director graph, it is defined as:

$$A^D = \begin{pmatrix} 0 & R^D \\ (R^D)^\top & 0 \end{pmatrix} \tag{4}$$

$R^D \in \mathbb{R}^{N\times M_D}$ denotes the user-director interaction matrix, where each element represents the number of times a user has seen the movies directed by the corresponding director; 0 is all-zero matrix.

In fact, Eq.(3) with matrix form is equivalent to modeling updated embedding of each user u and each director i as:

$$e^{D,(l+1)}_u = \sum_{i\in\mathcal{N}^D_u} \frac{A^D_{ui}}{\sqrt{|\sum_j R^D_{uj}|}\sqrt{|\sum_j R^D_{ji}|}} e^{D,(l)}_i, \tag{5}$$

$$e^{D,(l+1)}_i = \sum_{u\in\mathcal{N}^D_i} \frac{A^D_{ui}}{\sqrt{|\sum_j R^D_{ji}|}\sqrt{|\sum_j R^D_{uj}|}} e^{D,(l)}_u, \tag{6}$$

where $\sum_j R^D_{uj}$ and $\sum_j R^D_{ji}$ respectively denote the degree of node u and node i in the user-director interaction graph; $\mathcal{N}^D_u$ and $\mathcal{N}^D_i$ respectively denote the neighbor set of user u and director i in the user-director graph.

We elaborately describe the node representation learning process of the user-director graph. Since the user-movie, user-genres, and user-actor aspect representation learning process is similar, we omit it for brevity.

D. RATING PREDICTION

After the previous step, we can obtain embeddings at each aspect-level module and each propagation layer of each module. Each user u can be expressed as:

$$e_u = [e^{A_0}_u, \cdots, e^{A_K}_u], \tag{7}$$

where, $e_u^{A_k} = [e_u^{A_k,(0)}, \cdots, e_u^{A_k,(L)}]$, the superscript A_k represents the corresponding aspect. We need to combine these embeddings to generate the final representation of the users and items. We divide the process into two steps: (1) Layer-level embeddings fusion. (2) Aspect-level embeddings fusion.

1) LAYER-LEVEL EMBEDDINGS FUSION

We employ the weighted average to combine the layer-level embeddings:

$$e_u^{A_k} = \frac{1}{L+1} \sum_{l=0}^{L} e_u^{A_k,(l)}, \tag{8}$$

$$e_i^{A_k} = \frac{1}{L+1} \sum_{l=0}^{L} e_i^{A_k,(l)}. \tag{9}$$

2) ASPECT-LEVEL EMBEDDINGS FUSION

We need to fuse the aspect-level embeddings further to build the final representation of user and item. For aspect-level embeddings, we still employ the weighted average for simplicity:

$$e_u = \frac{1}{K+1} \sum_{k=0}^{K} e_u^{A_k}, \tag{10}$$

$$e_i = \frac{1}{K+1} \sum_{k=0}^{K} e_i^{A_k}. \tag{11}$$

We combine the movie's embedding with the corresponding embedding of genres, director, and actor as the movie's final representation. It should be noted that we implement this process in a matrix form, while the number of genres, directors, and actors is lower than the number of movies because an actor may have participated in more than one movie, which leads to different dimensions of the embedding matrix. So before performing aspect-level embedding combination, we employ a mapping table to perform redundant operations on the rows of embedding matrix of aspects of genres, director, and actor, so that their dimensions consistent with the movie-aspect embedding matrix.

3) RATING PREDICTION

We conduct inner product operation on the final embedding of user and item to obtain their affinity scores.

$$\hat{y}(u,i) = e_u^{\top} e_i. \tag{12}$$

E. OPTIMIZATION

We employ Bayesian Personalized Ranking (BPR) loss to update model parameters that assume that observed interactions better reflect user preferences than unobserved interactions, and therefore should be assigned a higher affinity score. The objective function is formulated as follow:

$$Loss = \sum_{(u,i,j)\in T} -\ln \sigma(\hat{y}_{ui} - \hat{y}_{uj}) + \lambda \|\Theta\|^2, \tag{13}$$

where $T = \{(u,i,j)|(u,i) \in R^+, (u,j) \in R^-\}$, R^+ denotes observed interactions and R^- denotes unobserved interactions; $\Theta = \{E^{A_k,(0)}|k \in 0,\ldots,K\}$ is trainable parameters, which only including the multi-aspect embedding matrix of the 0-th layers. λ is the regularization parameter to prevent over fitting; $\sigma(\cdot)$ is the sigmoid function. We use the mini-batch Adam as the optimizer, which is able to adaptively control the learning rate with respect to the absolute value of gradient. Note that the training pairs in T is sampled randomly and we update the embeddings for all nodes in each batch.

F. TIME COMPLEXITY ANALYSES

This subsection will compare the time complexity of the three algorithms LGC-ACF (our), NGCF, and LightGCN.

For the l-th propagation layer of GCN, the computational complexity is $O(|\mathcal{R}^+|d_{l-1} + nd_l d_{l-1})$ [31], where $|\mathcal{R}^+|$ is the number of non-zero elements in the Laplacian matrix; n is number of nodes; d_l and d_{l-1} are the embedding size of the current layer and the previous layer, respectively. For the sake of comparison, we assume that the feature transformation keeps the dimension of the embedding matrix constant, that is, the node embedding dimension of each layer is d. Then the computational complexity of a GCN layer is $O(|\mathcal{R}^+|d + nd^2)$.

- NGCF [8] stacks multiple GCN layers, its time complexity is $O(L|\mathcal{R}^+|d + Lnd^2)$, where L is the layer number.
- LightGCN [11] removes the feature transformation and nonlinear activation, and only retains the neighbor aggregation operation in the standard GCN. Its time complexity is $O(L|R^+|d)$.
- LGC-ACF introduces multi-aspect information on the basis of LightGCN, which inevitably increases the computational complexity. The time complexity of LGC-ACF is $O(L\sum_{a=1}^{K}|R_a{}^+|d)$, where K denotes the number of aspect-level information. In practical applications, employing too much auxiliary information increases the computational complexity and contributes little to recommendation performance, so K is usually small (no more then 5). We introduce multi-aspect user-item interaction information, and the computational complexity remains at the same level, which is tradeoffable in practical applications.

IV. EXPERIMENTS

In this part, we evaluate our model on three real-world datasets. We introduce the experimental settings in section IV-A. Then, we compared with other methods in section IV-B. To prove the effectiveness of fusing multi-aspect information, we conduct ablation experiments in section IV-C. Finally, we conduct hyper-parameter studies in section IV-D.

A. EXPERIMENTAL SETTINGS

1) DATASETS

We conduct extensive experiments on three real-world datasets: Movielens [32], Amazon [33], Taobao [34], all of

which are accessible and vary in terms of domain, size, and sparsity. The statistics of the datasets are summarized in Table 2 and Table 3.

TABLE 2. The statistics of the datasets.

Dataset	#users	#items	ratings	Rating Density
Movielens	610	9606	100551	0.01716
Amazon	13201	14094	390581	0.00210
Taobao	69166	39406	918431	0.00034

TABLE 3. The statistics of the auxiliary information.

Movielens	#genres	21
	#directors	3942
	#actors	4034
Amazon	#brands	2771
	#categories	15
Taobao	#categories	1671

Movielens: Movielens dataset has been widely used to evaluate recommendation methods. The version we selected is **ml latest small**, which contains more than 600 users' rating information on more than 9000 movies. To get more information about movies, we crawled the genres, director, and actor information of movies from IMDB, and for a few invalid links or missing entries, we delete it from the data set. We change the explicit rating data to implicit data.

Amazon: Amazon-review is a widely used recommendation dataset. In our experiment, we select **Amazon-Electronics** for evaluation. For this dataset, we remove items that interact less than 5, while ensuring that each user interacts with no less than 20 items. We change the explicit rating data to implicit data.

Taobao: The Taobao dataset is a public online e-commerce dataset collected from taobao.com. We consider 'buy' and 'add to cart' behaviors as positive interactions, and we remove users with fewer than 10 interactions.

2) EVALUATION METRICS AND BASELINES

To evaluate the performance of top-k recommendation, we adopt two widely used ranking metrics [7], [35]: recall@K and NDCG@K, where K = 20 by default. For each user in the test set, we take all items into ranking, except the items in the training set. We compare our proposed LGC-ACF with the following methods:

- **MF** [14] is a classical matrix factorization model.
- **DMF** [21] uses the interaction matrix as input and uses a deep neural network to map users and items to a common low dimensional space.
- **GCMC** [19] originally focuses on explicit feedback, constructs multiple adjacency matrices according to the type of score, and uses different weight matrices to decode different types of edges. In this paper, we use implicit feedback to simplify it to a single-layer GCN model.
- **NGCF** [8] stacks multiple standard GCNs, aiming to inject high-order collaborative signal into the embedding process.
- **DGCF** [27] focus on user-item relationships at the finer granularity of user intents, and disentangle these intents to generate disentangled representations.
- **LR-GCCF** [9] proposes a residual network structure to deal with the over-smoothing problem.
- **LightGCN** [11] explores the embedding propagation process, eliminates the nonlinear activation function and feature transformation matrix, simplifies the model, and improves the performance.

3) IMPLEMENTATION

We implement our LGC-ACF model in Pytorch. For a fair comparison, we tune the parameter settings of each model to get the best performance. For example, in particular, for NGCF and LightGCN, we search the learning rate in {0.0001, 0.0005, 0.001, 0.005} and layer number in {1, 2, 3}, the coefficient of L2 normalization is searched in {1e-1, 1e-2, 1e-3, 1e-4}. In other baseline models, we adjust the parameters in a similar way to get the best performance. For our proposed model, we initialize the model parameters with a Gaussian distribution of mean 0 and standard deviation 0.1, and apply grid research for learning the hyperparameters. We randomly select 80% of the interaction record for training and the rest for test. All experiments are conducted on a machine with a GPU (NVIDIA RTX-2080) and a CPU (Intel i-7 9700k).

B. PERFORMANCE COMPARISON

The results of the performance comparison are shown in Table 4. We have the following observations:

- Our proposed LGC-ACF achieves the best performance over all the datasets and metrics. In particular, compared with the strongest baseline, the improvement on Movielens, Amazon, Taobao is 4.975%, 1.305%, 14.35%, respectively, in terms of recall. The significant improvement indicates that the fusion of user-item interaction information of multi-aspect is useful for distilling the representative embeddings. Note that our method can be further improved by employing a more reasonable aggregator, e.g., attention [36], LSTM [37] *etc*. Since the contribution of different aspects of information to recommendation performance is always distinct. We leaving the problem in future work.
- MF performs poorly on all three datasets, indicating that inner product is hard to fully model the complicated relationship between users and items. DMF performs consistently better than MF in all cases, which shows that non-linear projection can capture more information. The performance of GCN-based methods is better than DMF, which indicates the effectiveness of exploiting the high-order connectivity.

TABLE 4. Overall comparison. The best performance is highlighted in bold, and the second is underlined. The last row is the improvement of LGC-ACF over the best baseline.

Dataset	Movielens				Amazon				Taobao			
method	recall@20	recall@50	NDCG@20	NDCG@50	recall@20	recall@50	NDCG@20	NDCG@50	recall@20	recall@50	NDCG@20	NDCG@50
MF	0.2187	0.3513	0.2767	0.3074	0.0435	0.0806	0.0304	0.0433	0.0323	0.0575	0.0172	0.0239
DMF	0.2270	0.3552	0.2990	0.3249	0.0465	0.0834	0.0359	0.0486	0.0359	0.0613	0.0228	0.0302
GCMC	0.2265	0.3546	0.2998	0.3243	0.0468	0.0872	0.0338	0.0477	0.0339	0.0611	0.0182	0.0254
NGCF	0.2368	0.3520	0.3085	0.3283	0.0498	0.0881	0.0374	0.0507	0.0542	0.0903	0.0302	0.0397
LR-GCCF	0.2442	0.3755	0.3323	0.3543	0.0617	0.1090	0.0455	0.0619	0.0577	0.0968	0.0326	0.0429
DGCF	0.2390	0.3634	0.3133	0.3358	0.0572	0.0948	0.0426	0.0556	0.0579	0.0978	0.0322	0.0427
LightGCN	0.2459	0.3778	0.3311	0.3546	0.0579	0.1017	0.0439	0.0590	0.0585	0.0966	0.0329	0.0430
LGC-ACF	**0.2604**	**0.3931**	**0.3524**	**0.3708**	**0.0632**	**0.1092**	**0.0478**	**0.0638**	**0.0671**	**0.1115**	**0.0378**	**0.0494**
%Improv.	5.90%	4.05%	6.05%	4.57%	2.43%	0.18%	5.05%	3.07%	14.7%	14.0%	14.9%	14.9%

- We notice that the GCMC, which is also based on GCN, performs mediocre. It is at the same level as DMF on Movielens and Amazon, and lower than DMF on Taobao. We guess a possible reason is that GCMC only models the first-order relationship, and it does not integrate the initial embedding into the final representation of users and items.
- With the increasing sparsity of datasets, the gap of performance between GCN-based methods and classical methods is also widening. One possible reason is that on dense datasets, the user-item interaction is informative enough to describe user preference; on sparse datasets, due to the lack of interaction between users and items, it is necessary to explore higher-order relationships to express user preferences.
- LightGCN outperforms NGCF by a large margin in all cases, which proves once again that the feature transformation and nonlinear activation operation in standard GCN will damage the recommendation performance.

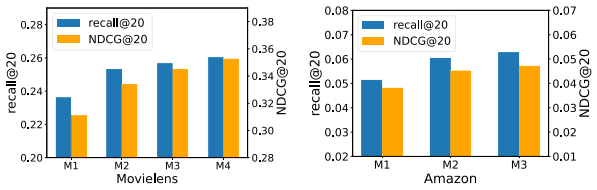


FIGURE 4. Ablation study of aspect-level information. (a)On Movielens dataset. (b)On Amazon dataset.

C. ABLATION ANALYSES

To investigate whether LGC-ACF can benefit from multi-aspect user-item interaction information, we conduct experiments with the increase of aspect to fuse more information into our model. Fig.4 shows the experimental results. It demonstrates that combining different aspects of information can improve the performance of recommendation. Specifically, Fig.4(a) shows the results on Movielens. M1 means the result of only exploiting the user-item interaction on the movie aspect. M2 means that genres aspect interaction information is added. M3 means the director aspect information is further added. M4 means we use all four aspects of information. Fig.4(b) shows the results on Amazon. We integrate the aspect-level information in the order of product, brand, and category. From the experimental results of the above two datasets, we have the following observation:

- Each aspect of user-item interaction can bring unique information, which reflects user preferences to some extent, improve the performance of recommendation.
- The performance of the model will be greatly improved after adding additional information of one aspect, but with aspect-level information is added further, the growth of the model performance becomes slower, i.e., the improvement is non-linear. One possible reason is that noise is introduced as well as information, and it will increase the training difficulty.

In fact, LightGCN is a particular case of our LGC-ACF. If only exploit purchase (or click) history, then LGC-ACF is equivalent to LightGCN. Our main contribution is to model and fuse different aspects of user-item interaction information through graph convolutional network, aiming to obtain a more accurate representation of users and items.

D. HYPER-PARAMETER STUDIES

1) IMPACT OF LAYER NUMBERS

To investigate the effect of the number of embedding propagation layers on performance, we set the number of propagation layers in range of {1, 2, 3, 4} and summarize the empirical results in Fig.5.

As we can see, on Movielens, recall@20 reaches the optimal value when the number of layers is 2, NDCG@20 reaches the optimal value when the number of layers is 3. Further stacking propagation layer on the top of the model will

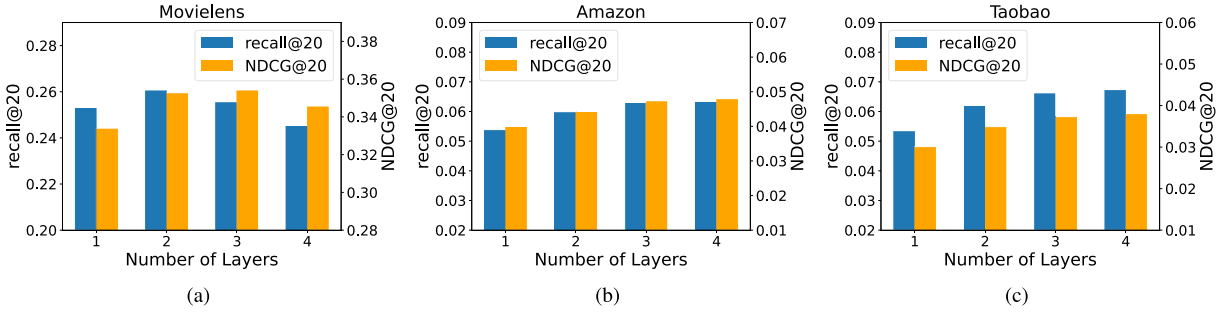


FIGURE 5. Effect of embedding propagation layer numbers.

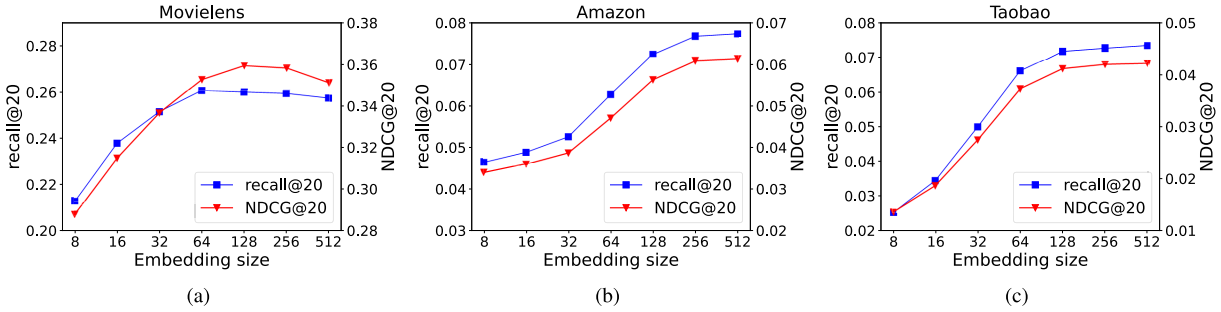


FIGURE 6. Effect of different embedding dimensions.

lead to performance degradation. Since Movielens is a dense dataset with rich interactive information, a shallow network is enough to extract high-quality expression, and stacking more layers will lead to over-fitting. On Amazon and Taobao, recall@20 and NDCG@20 keep growing with the increase of propagation layers, which proves the effectiveness of our LGC-ACF on sparse datasets.

2) IMPACT OF EMBEDDING DIMENSIONS

The dimension of embeddings d is also a crucial hyper-parameter to control the complexity and capacity of our proposed LGC-ACF. Therefore, we study the effect of the embedding dimension on recommendation performance, and we set the embedding size from 8 to 512. The experimental results are shown in Fig.6. These three subplots describe the performance curves with different numbers of dimensions of LGC-ACF on Movielens, Amazon, and Taobao, respectively. They all share a common trend: with the gradually increasing of embedding dimension, recommendation performance will first gradually increase and reach the peak, and then with the further increase of dimension, the performance will remain stable or even decline.

V. CONCLUSION

In this work, we focus on how to model additional information beyond user-item interaction history through graph convolutional networks. To this end, we propose a novel Light GCN based Aspect-level Collaborative Filtering model (LGC-ACF), which exploit user-item interaction information from different aspects through the bipartite graph structure. Distinct from existing methods that perform graph convolution operations in the knowledge graph [24], [25], we first construct aspect-level interaction graphs through interaction history and item knowledge information. Then, our model learns the aspect-level embeddings of user and item based on these interaction graphs, separately. Finally, we perform a weighted average over all aspect-level embeddings to construct the final representation of user and item prior to generate predictions. Extensive experimental results on three real-world datasets verify the effectiveness of our proposed model. In future, we will work on further improvement by studying more reasonable combination mechanism of aspect-level representation of users and items.

REFERENCES

[1] J. A. Konstan and J. Riedl, "Recommender systems: From algorithms to user experience," *User Model. User-Adapted Interact.*, vol. 22, nos. 1–2, pp. 101–123, Apr. 2012.

[2] C. A. Gomez-Uribe and N. Hunt, "The netflix recommender system: Algorithms, business value, and innovation," *ACM Trans. Manage. Inf. Syst.*, vol. 6, no. 4, pp. 1–19, Jan. 2016.

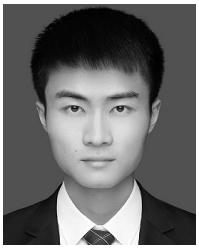
[3] S. Oramas, V. C. Ostuni, T. D. Noia, X. Serra, and E. D. Sciascio, "Sound and music recommendation with knowledge graphs," *ACM Trans. Intell. Syst. Technol.*, vol. 8, no. 2, pp. 1–21, Jan. 2017.

[4] 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 *Proc. 27th ACM Int. Conf. Multimedia*, Oct. 2019, pp. 1437–1445.

[5] X. Bai, B. Barla Cambazoglu, F. Gullo, A. Mantrach, and F. Silvestri, "Exploiting search history of users for news personalization," *Inf. Sci.*, vols. 385–386, pp. 125–137, Apr. 2017.

[6] L. Hu, C. Li, C. Shi, C. Yang, and C. Shao, "Graph neural news recommendation with long-term and short-term interest modeling," *Inf. Process. Manage.*, vol. 57, no. 2, Mar. 2020, Art. no. 102142.
[7] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural collaborative filtering," in *Proc. 26th Int. Conf. World Wide Web*, 2017, pp. 173–182.
[8] X. Wang, X. He, M. Wang, F. Feng, and T.-S. Chua, "Neural graph collaborative filtering," in *Proc. 42nd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Jul. 2019, pp. 165–174.
[9] L. Chen, L. Wu, R. Hong, K. Zhang, and M. Wang, "Revisiting graph based collaborative filtering: A linear residual graph convolutional network approach," in *Proc. AAAI Conf. Artif. Intell.*, 2020, vol. 34, no. 1, pp. 27–34.
[10] Z. Cheng, Y. Ding, L. Zhu, and M. Kankanhalli, "Aspect-aware latent factor model: Rating prediction with ratings and reviews," in *Proc. World Wide Web Conf.*, 2018, pp. 639–648.
[11] X. He, K. Deng, X. Wang, Y. Li, Y. Zhang, and M. Wang, "LightGCN: Simplifying and powering graph convolution network for recommendation," 2020, *arXiv:2002.02126*. [Online]. Available: http://arxiv.org/abs/2002.02126
[12] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, Aug. 2009.
[13] Y. Koren, "Factorization meets the neighborhood: A multifaceted collaborative filtering model," in *Proc. 14th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2008, pp. 426–434.
[14] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "BPR: Bayesian personalized ranking from implicit feedback," 2012, *arXiv:1205.2618*. [Online]. Available: http://arxiv.org/abs/1205.2618
[15] S. Sedhain, A. K. Menon, S. Sanner, and L. Xie, "AutoRec: Autoencoders meet collaborative filtering," in *Proc. 24th Int. Conf. World Wide Web*, May 2015, pp. 111–112.
[16] D. Liang, R. G. Krishnan, M. D. Hoffman, and T. Jebara, "Variational autoencoders for collaborative filtering," in *Proc. World Wide Web Conf.*, 2018, pp. 689–698.
[17] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," 2016, *arXiv:1606.09375*. [Online]. Available: http://arxiv.org/abs/1606.09375
[18] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," 2016, *arXiv:1609.02907*. [Online]. Available: http://arxiv.org/abs/1609.02907
[19] R. van den Berg, T. N. Kipf, and M. Welling, "Graph convolutional matrix completion," 2017, *arXiv:1706.02263*. [Online]. Available: http://arxiv.org/abs/1706.02263
[20] A. Mnih, R. R. Salakhutdinov, "Probabilistic matrix factorization," in *Proc. 20th Int. Conf. Neural Inf. Process. Syst.*, 2008, pp. 1257–1264.
[21] H.-J. Xue, X. Dai, J. Zhang, S. Huang, and J. Chen, "Deep matrix factorization models for recommender systems," in *Proc. 26th Int. Joint Conf. Artif. Intell.*, Aug. 2017, pp. 3203–3209.
[22] C. Shi, X. Han, S. Li, X. Wang, S. Wang, J. Du, and P. Yu, "Deep collaborative filtering with multi-aspect information in heterogeneous networks," *IEEE Trans. Knowl. Data Eng.*, early access, Sep. 17, 2019, doi: 10.1109/TKDE.2019.2941938.
[23] R. Ying, R. He, K. Chen, P. Eksombatchai, W. L. Hamilton, and J. Leskovec, "Graph convolutional neural networks for Web-scale recommender systems," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2018, pp. 974–983.
[24] H. Wang, M. Zhao, X. Xie, W. Li, and M. Guo, "Knowledge graph convolutional networks for recommender systems," in *Proc. World Wide Web Conf.*, 2019, pp. 3307–3313.
[25] X. Wang, X. He, Y. Cao, M. Liu, and T.-S. Chua, "KGAT: Knowledge graph attention network for recommendation," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2019, pp. 950–958.
[26] J. Ma, P. Cui, K. Kuang, X. Wang, and W. Zhu, "Disentangled graph convolutional networks," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 4212–4221.
[27] X. Wang, H. Jin, A. Zhang, X. He, T. Xu, and T.-S. Chua, "Disentangled graph collaborative filtering," in *Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Jul. 2020, pp. 1001–1010.
[28] X. Wang, R. Wang, C. Shi, G. Song, and Q. Li, "Multi-component graph convolutional collaborative filtering," in *Proc. AAAI Conf. Artif. Intell.*, 2020, vol. 34, no. 4, pp. 6267–6274.
[29] F. Wu, T. Zhang, A. H. de Souza, C. Fifty, T. Yu, and K. Q. Weinberger, "Simplifying graph convolutional networks," 2019, *arXiv:1902.07153*. [Online]. Available: http://arxiv.org/abs/1902.07153
[30] S. Peng and T. Mine, "A robust hierarchical graph convolutional network model for collaborative filtering," 2020, *arXiv:2004.14734*. [Online]. Available: http://arxiv.org/abs/2004.14734
[31] W.-L. Chiang, X. Liu, S. Si, Y. Li, S. Bengio, and C.-J. Hsieh, "Cluster-GCN: An efficient algorithm for training deep and large graph convolutional networks," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2019, pp. 257–266.
[32] F. M. Harper and J. A. Konstan, "The MovieLens datasets: History and context," *ACM Trans. Interact. Intell. Syst.*, vol. 5, no. 4, pp. 1–19, Jan. 2016.
[33] R. He and J. McAuley, "Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering," in *Proc. 25th Int. Conf. World Wide Web*, Apr. 2016, pp. 507–517.
[34] H. Zhu, X. Li, P. Zhang, G. Li, J. He, H. Li, and K. Gai, "Learning tree-based deep model for recommender systems," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2018, pp. 1079–1088.
[35] J.-H. Yang, C.-M. Chen, C.-J. Wang, and M.-F. Tsai, "HOP-rec: High-order proximity for implicit recommendation," in *Proc. 12th ACM Conf. Recommender Syst.*, Sep. 2018, pp. 140–144.
[36] J. Chen, H. Zhang, X. He, L. Nie, W. Liu, and T.-S. Chua, "Attentive collaborative filtering: Multimedia recommendation with Item- and component-level attention," in *Proc. 40th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Aug. 2017, pp. 335–344.
[37] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 1024–1034.

DENGHUA MEI received the B.E.E. degree from the School of Computer Science and Technology, Huazhong University of Science and Technology, and the M.S.E.E. and Ph.D. degrees from Southwest Jiaotong University. He is currently an Associate Professor with the School of Computer Science and Engineering, South China University of Technology. His research interests include deep learning and KDD.



NIU HUANG was born in Shangrao, Jiangxi, China, in 1997. He received the B.S. degree from the China University of Petroleum (East China), in 2019. He is currently pursuing the master's degree with the School of Computer Science and Engineering, South China University of Technology. His research interests include collaborative filtering and graph convolutional networks.

XIN LI is currently pursuing the master's degree with the South China University of Technology. His research interests include deep learning and recommender systems.