A U-shaped Hierarchical Recommender by Multi-resolution Collaborative Signal Modeling

Peng Yi[1], Xiongcai Cai[1,2] ✉, and Ziteng Li[1]

[1] UNSW Sydney, NSW 2052, Australia
x.cai@unsw.edu.au
[2] Techcul Research, Sydney, Australia

Abstract. Items (users) in a recommender system inherently exhibit hierarchical structures with respect to interactions. Although explicit hierarchical structures are often missing in real-world recommendation scenarios, recent research shows that exploring implicit hierarchical structures for items (users) would largely benefit recommender systems. In this paper, we model user (item) implicit hierarchical structures to capture user-item relationships at various resolution scales resulting in better preferences customization. Specifically, we propose a U-shaped Graph Convolutional Network-based recommender system, namely UGCN, that adopts a hierarchical encoding-decoding process with a message-passing mechanism to construct user (item) implicit hierarchical structures and capture multi-resolution relationships simultaneously. To verify the effectiveness of the UGCN recommender, we conduct experiments on three public datasets. Results have confirmed that the UGCN recommender achieves overall prediction improvements over state-of-the-art models, simultaneously demonstrating a higher recommendation coverage ratio and better-personalized results.

Keywords: Recommender Systems · Hierarchical Model · Embedding Learning.

1 Introduction

Recommender systems implemented by a wide range of online businesses are critical in alleviating the information overload problem [30]. The basic point of building a recommender system is to model the user-item relationship based on previous interactions. As one of the most widely used recommendation techniques, model-based collaborative filtering (CF) algorithms extract useful information about user-item connectivity by projecting users and items into a shared latent space and representing them with corresponding low-dimensional embeddings [9] [15]. In other words, model-based CF describes user preferences for items through the inner product of the projected embeddings in the shared space [22]. Therefore, the algorithm for projecting users/items into certain embeddings plays a key role in the success of these model-based CF recommenders.

There are many approaches to learn user/item embeddings, including classical matrix factorization (MF) [14], neural network-based models, and graph

convolutional networks (GCNs)-based models [26] [8]. Although these models present competitive performance, they also have the obvious drawback of neglecting the inherent hierarchy of items/users. More specifically, most existing embedding learning models only focus on modeling individual items or users independently, which cannot fully capture the items' (or users') hierarchies information and hence fail to precisely model personalized preference. Take the movie Frozen as an example, it belongs to the subgenre "Family Animation" and can be further categorized into the genre "Animation", demonstrating a hierarchical structure of "individual → sub-genre → genre". Similarly, users may present a similar hierarchy of "individual → occupation → age". Since items of the same subgenre (or genre) are likely to have similar attributes, they are likely to acquire similar preferences [25]. Thus, hierarchical information about items or users can very likely improve the preference modeling process of recommender systems.

It is worth pointing out that in real-world recommendation scenarios, explicit hierarchies are often not applicable [24]. For this reason, some researchers have used MF-based models to learn implicit item/user hierarchies [25]. Specifically, the implicit item/user hierarchy can be easily obtained by decomposing the original item/user embedding matrix into several smaller and more compressed item/user matrices, respectively. The success of these models confirms the validity of exploring implicit hierarchies [2]. However, there are still two limitations that limit the performance of these models, which can be summarized as the limited representation based on a simple MF model and the neglect of diverse collaboration signals in different hierarchical levels. More specifically, recent studies have argued that the basic MF-based models simply adopt the interaction between users and items as the ultimate objective function, but ignore the potential similarity signals stored in the interaction [1]. From this perspective, models based on graphical convolutional networks (GCNs), which can naturally model the user-item relationship in the interaction graph structure, propose a more reasonable and efficient way to build recommender systems. In addition to the limitations imposed by simple models, existing models that focus solely on building two separate multilayer architectures are problematic. More directly, exploring parallel user/item hierarchies only shows the multi-level relationships within a user or item. However, the core information contained in the hierarchy is the multi-resolution collaborative signals stored in different levels. For example, given the interaction that a user "Alice" like a movie "Frozen", exploring the structure of "Alice → Second Grade Nursery Student → Kids" and "Frozen → Family Animation → Animation" merely displays the inner association for a user or an item. Indeed, this interaction not only reveals "Alice's" unique preference for "Frozen", but also represents a shared interest that "Second Grade Nursery Students" may prefer "Family Animation", or announces a more general signal that "Kids" prefer "Animation". Clearly, these multi-resolution collaborative signals will help generate better recommendation predictions.

Based on the above analysis, we propose an implicit user/item hierarchical exploration model, i.e., UGCN, which utilizes a u-shaped hierarchical graph con-

volution model and is able to capture user-item collaboration signals in different resolution scales. Specifically, the u-shaped structure uses a pooling operation to adaptively compress the original user-item interaction graph into smaller compressed graphs. In other words, by merging similar nodes in the fine-grained graph into a new group node in the coarse-grained graph, collaboration signals at different resolution scales can be better captured by graph convolution operations on different compressed graphs. Then, the captured collaborative signals are gradually projected back to the original nodes through symmetric unpooling operations. With the multi-resolution collaborative signal captured in final item/user embeddings, better personalized recommendations can be ultimately generated.

In summary, this work has the following main contributions.

- We propose the UGCN recommender, a U-shaped hierarchical recommender based on graph convolutional networks, which captures diverse collaborative signal from a stacked multilayer graph architecture.
- We conducted an empirical study on three million-level datasets of recommender systems. The experimental results show that the proposed model achieves the best recommendation performance, obtains significant improvements in recommendation coverage, and at the same time obtains more personalized recommendation results.

2 Related Work

There are many state-of-the-art model-based recommender systems. The most representative one is Matrix Factorization (MF). MF-based models such as PMF [19] and SVD [15] utilize straightforward procedures to generate low-dimensional user (item) embeddings directly from the user (item) one-hot vector. Although these models could achieve reasonable results on some experimental datasets, they are still insufficient for real-world recommender systems due to their simple structure and limited model expressions. With the superior power of representation learning, utilizing neural architecture to learn embeddings has become popular for model-based recommender systems. Zhuang et al. [31] proposed a model named REAP, which applies an autoencoder to generate the latent factors for users and items from the user-item matrix. Xue et al. [28] proposed a Deep Matrix Factorization Model (DMF) by utilizing two parallel neural networks to map the user and item into low-dimensional space from both explicit rating and implicit behavior. Han et al. [7] indicated that using one aspect representations is insufficient and proposed modeling the complicated relationship in recommender systems through Heterogeneous Information Network (HIN). Ebesu et al. [5] believed the user/item neighborhood information can help generate better results and utilized a memory network to model the neighborhood information and generate neighbor-based embeddings. Moreover, Jiang et al. [11] proposed a convolutional neural network-based Gaussian model to represent the users (items) with uncertainty and create better recommendations.

Compared with the traditional collaborative filtering [22] and neural collaborative filtering recommender systems, the GCN-based recommendation methods demonstrated competitive performances [8] and have attracted much research attention. Wang et al. [26] proposed Neural Graph Collaborative Filtering (NGCF), which utilizes a bipartite graph structure to exploit the high-order user-item connectivity and achieve promising recommendation performance. He et al. [8] further simplified the design of GCN to make it more concise and appropriate for recommendation. Moreover, Sun et al. [23] pointed out that directly applying GCNs to process the bipartite graph is sub-optimal since this method neglects the intrinsic differences between user nodes and item nodes. To this end, they proposed NIA-GCN, a new framework that explicitly models relationships between neighboring nodes and exploits the heterogeneous nature of the user-item bipartite graph. Liu et al. [17] mentioned that existing GCN-based recommenders often suffer from the over-smoothing problem. To alleviate that problem, they presented a recommendation model, namely IMP-GCN, which performs high-order graph convolution inside subgraphs and hence limits the neighbor exploration process to similar users (items). Furthermore, Wu et al. [27] explored self-supervised learning on GCN-based recommenders and targeted improving their recommendation accuracy and robustness. Although these models present promising recommendation results, the neglect of item (user) hierarchical structures still prevents them from generating superior predictions.

In fact, item (user) hierarchical structures have been widely explored and utilized in recommendation scenarios [29] [20]. For example, Lu et al. [18] utilized item hierarchies stored in side-information and demonstrated the effectiveness of incorporating explicit hierarchical structure in recommender systems. However, real-world recommendation data may not contain detailed external side information or can not directly provide explicit item (user) hierarchy structures. Therefore, exploring and using the implicit hierarchical structures of user-item interaction information has become a common method to solve this problem. Wang et al. [25] proposed a framework, namely IHSR, which can construct implicit item (user) hierarchies by gradually decomposing the item (user) characteristic matrix. Analogously, Li [16] proposed a Hidden Hierarchical Matrix Factorization (HHMF), which learns the hidden hierarchical structures from the user-item scoring record without prior knowledge of the hierarchy. Even though these models achieve competitive results compared to basic model-based recommenders, the straightforward MF-based strategies still limit the model expression and result in sub-optimal predictions.

In summary, most existing embedding learning models neglected the inherent item (user) hierarchical structure and did not fully capture the valuable information stored in user-item interactions. Meanwhile, the explicit hierarchical structures are not often provided in real-world recommendation scenarios, and existing implicit hierarchical models are merely based on simple matrix factorization, which is insufficient for handling increasingly sparse and complex recommendation scenarios. Hence, a better hierarchical model for the recommendation problem is needed.

3 Methodology

3.1 The Basic Collaborative Filtering Model

The basic idea of traditional model-based collaborative filtering algorithms can be summarized as utilizing a model to project an individual user/item into a shared latent space and modeling a user's preference given an item by the inner product of the corresponding user/item embeddings. To make it more clear, we adopt e_u and e_i to represent the learned personalized embedding for user u and item i respectively. Moreover, as classic models merely rely on interactions and neglects high-level hierarchical context information, we formulate e_u and e_i as e_{u-c0} and e_{i-c0}, where $c0$ indicates the context extracted in the $0th$ hierarchy. Then, the preference of user u given item i can be computed as follows:

$$y_{ui} = e_u \cdot (e_i)^T = e_{u-c0} \cdot (e_{i-c0})^T \tag{1}$$

3.2 Modeling Implicit Hierarchies

In the basic model, the interaction is often considered as the unique preference of a single user given a specific item. However, as a member of our modern society, a user's behavior is inevitably influenced by implicit contextual information and would demonstrate some implicit context-based preferences. When it comes to interaction modeling recommender systems, we interpret the implicit contextual information as the group-level collaborative signal. More specifically, according to the example in Section 1, a user's interaction is not only demonstrating an individual's personalized interest but also indicates several informative group-level tendencies. As these group-level tendencies (multi-resolution collaborative signal) are useful for understanding the user's behavior and would be helpful for generating customized recommendations, a hierarchical model becomes a more reasonable way to construct general recommender systems. To be more specific, we decompose user embedding e_u and item embedding e_i into the summation of different representations, which corresponds to the preference information learned among different levels of hierarchical architectures. Take a n-levels hierarchical model as an example; the final user embedding e_u and item embedding e_i can be calculated through:

$$e_u = e_{u-c0} + e_{u-c1} + e_{u-c2} + ... + e_{u-cn} \tag{2}$$

$$e_i = e_{i-c0} + e_{i-c1} + e_{i-c2} + ... + e_{i-cn} \tag{3}$$

, where e_{u-c0} and e_{i-c0} represent the personalized embedding for individual user u and item i respectively. While, $e_{u-cx}, x \in (1, n)$ and $e_{i-cy}, y \in (1, n)$ demonstrate the diverse group-level collaborative signals. It is worth noticing that $e_{u-cx}, x \in (1, n)$ and $e_{i-cy}, y \in (1, n)$ would be shared among the group members within the same group at the same hierarchical level. Moreover, the greater x value in $e_{u-cx}, x \in (1, n)$ corresponds to a more compressed group

representation, which indicates e_{u-cx} would be shared by a larger number of group members. With the defined e_u and e_i in hierarchical architecture, the user u's preference given item i can be estimated through:

$$y_{ui} = e_u \cdot (e_i)^T = (e_{u-c0} + e_{u-c1} + ... + e_{u-cn}) \cdot (e_{i-c0} + e_{i-c1} + ... + e_{i-cn})^T \quad (4)$$

3.3 UGCN Recommender

As discussed above, the essential point for constructing a hierarchical architecture is to extract multi-resolution collaborative signals and encode these informative signals into corresponding embeddings. To tackle this issue, we adopt the idea of utilizing a hierarchical graph structure to enable generating diverse group-level embeddings on different graph scales. In other words, we aim to gradually compress the original user-item interaction graph into several group-level interaction graphs. Hence, a node in the compressed graph could represent a group of similar users/items, and an edge in the compressed graph would indicate the shared behavior of group members. Taking a step further, the node embedding learned on diverse graph architectures of different scales can naturally capture the corresponding diverse group-level collaborative signals shared among certain group members. Following the analysis, we display the whole hierarchical recommender construction in three parts: item/user hierarchies extraction (i.e., graph architecture compressing), graph node embedding learning (i.e., multi-resolution collaborative signal modeling), and diverse embedding integrating.

Item/user Hierarchies Extraction To reasonably explore item/user hierarchies and properly compress the original user-item interaction graph, we follow the core point of collaborative filtering that similar users/items might exhibit similar preferences. In other words, by merging the similar nodes in the finer scaled graph into a group node in the coarser scaled graph, the generated diverse graph architectures could successfully reveal the user/item hierarchies relationships. More importantly, the group node in a coarser scaled graph can automatically exhibit the shared interest of those similar nodes in a finer scaled graph. To simplify the discussion, we adopt the common pooling operation to represent the process of compressing a finer graph into a coarser one, which can be defined as:

$$G'(U', I', E') = Pooling(G(U, I, E)) \quad (5)$$

, where $G(U, I, E)$ and $G'(U', I', E')$ is the finer scaled graph and coarser scaled graph respectively. Expressly, to implement the pooling operation, we utilize the following two steps: 1) group formation: assembling similar neighbors based on the similarity calculation on 1-hop connections; 2) node merging: generating a new group node based on the assembled members, where the edge of a new group node is mainly determined by the common connections of similar nodes. Meanwhile, a predefined ratio α is utilized to control the percentage of the nodes selected for grouping, enabling the construction of more flexible hierarchical architectures.

Graph Node Embedding Learning After obtaining the graphs of diverse scales, capturing the diverse collaborative signal in different graphs and learning informative embeddings for each node in each structure become the essential issue. To this end, we apply a graph convolution operation on graph architecture to encode informative collaborative signals into node embeddings. Specifically, the neighborhood aggregation proposed by LightGCN [8] is adopted to simplify the whole convolutional calculation process, which can be represented as:

$$e_u^{(k+1)} = \sum_{i \in \mathcal{N}_u} \frac{1}{\sqrt{|\mathcal{N}_u|}\sqrt{|\mathcal{N}_i|}} e_i^{(k)} \tag{6}$$

$$e_i^{(k+1)} = \sum_{u \in \mathcal{N}_i} \frac{1}{\sqrt{|\mathcal{N}_i|}\sqrt{|\mathcal{N}_u|}} e_u^{(k)} \tag{7}$$

, where $e_u^{(k)}$ and $e_i^{(k)}$ respectively denote the embedding of user u and item i generated after k convolutional operations. Meanwhile, $\mathcal{N}_u$ and $\mathcal{N}_i$ denote the set of items that are interacted by the user u and the set of users that interact with the item i, respectively. The symmetric normalization term $1/(\sqrt{|\mathcal{N}_i|}\sqrt{|\mathcal{N}_u|})$ follows the design of standard GCN [13] to avoid the scale of embeddings increasing with graph convolution operations. The final embeddings after operating m convolutional operations can be computed as:

$$e_u = \frac{1}{m}\sum_{k=0}^{m} e_u^{(k)} \quad ; \quad e_i = \frac{1}{m}\sum_{k=0}^{m} e_i^{(k)} \tag{8}$$

It is worth noticing that the graph convolution operations will be independently implemented on each graph structure. The initialization for graph nodes will only be applied to the original user-item interaction graph. Moreover, the initial node embeddings in the coarser graph can be easily generated by the mean embeddings of similar nodes in the finer graph that are selected and merged into the corresponding group nodes.

Diverse Embedding Aggregation Following the multi-resolution collaborative signal extraction process, properly propagating the extracted information back to final embeddings is also crucial. Symmetric to pooling operation, we utilize un-pooling operation to gradually restore the coarser scaled architecture back to the finer ones:

$$G(U, I, E) = UnPooling(G'(U', I', E')) \tag{9}$$

As a group-level node in a coarser scaled graph represents all of its group members' collaborative interaction patterns, the collaborative information contained in the group node is directly copied to its group members in the finer scaled graph. In this case, the group-level collaborative signal obtained in a coarser scaled graph can successfully be propagated back to the corresponding group members.

UGCN Architecture and Model Training To better exhibit the proposed UGCN architecture, we develop a 3-levels UGCN recommender depicted in Figure 1.

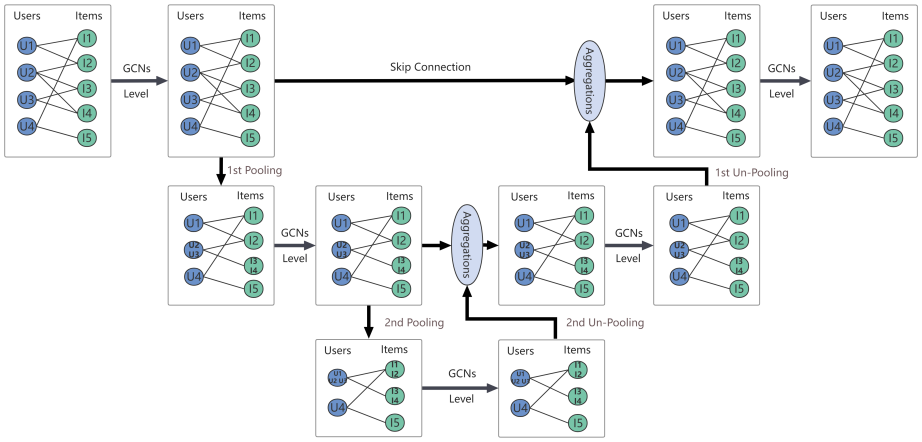


Fig. 1. An illustration of a 3-levels UGCN Recommender. In this example, we hierarchically stack the original user-item interaction graph with two compact graphs generated by Pooling operations. Then, simplified GCN layers are implemented to extract the multi-resolution collaborative signals at three graphs of different scales. The Un-pooling operations would gradually restore the finer graph architecture and integrate the informative signals into final user/item embeddings.

To properly train the UGCN recommender, we employ the BPR loss, a pairwise loss that aims to enlarge the prediction differences among positive and negative samples. The final loss function for UGCN is presented below:

$$\hat{x}_{uij} = y_{ui} - y_{uj} \tag{10}$$

$$L_{BPR} = -\sum_{u=1}^{M} \sum_{i \in \mathcal{N}_u} \sum_{j \notin \mathcal{N}_u} \ln \sigma(\hat{x}_{uij}) + \lambda ||\Theta|| \tag{11}$$

, where σ denotes the activation function, λ controls the L2 regularization strength, and Θ represents the parameter vector of the UGCN model. As the only trainable parameter is the initial embedding of the user-item interaction graph, the Θ in Equation (11) equals the embedding matrix $\mathbf{E}$.

3.4 Theoretical analysis for UGCN Recommender

In this section, we offer in-depth analyses of the UGCN Recommender, aiming to answer the question: how does this hierarchical recommender benefit real-world

recommendation problems? To answer this question, we utilize a 3-levels UGCN recommender depicted in Fig. 1 as an example.

Given a user u with his/her initial embedding e_{u0}, the final embedding e_u produced by a 3-levels UGCN model can be represented as:

$$e_u = GCN_0^2(GCN_0^2(e_{u0}) + GCN_1^2(GCN_1^2(e_{u1}) + GCN_2^2(e_{u2}))) \tag{12}$$

$$e_{u1} = Pooling(GCN_0^2(e_{u0})) \tag{13}$$

$$e_{u2} = Pooling(GCN_1^2(e_{u1})) \tag{14}$$

, where GCN_n^m represents operating m graph convolutional layers on the graph structure at $(n+1)-th$ level of the hierarchical graph model. For example, GCN_0^2 indicates operating two graph convolution layers on the original user-item interaction graph. Meanwhile, e_{u1} and e_{u2} represent the computed initial embedding of the merged group node in graph architectures at $2-nd$ and $3-rd$ level of hierarchical models, respectively. Moreover, the Un-pooling operation is discarded in the whole calculation as the Un-pooling operation simply copies the learned group node embedding back to its corresponding group members. By properly expanding the equation (12), we can finally obtain:

$$e_u = \underbrace{GCN_0^4(e_{u0})}_{e_{u-c0}} + \underbrace{GCN_0^2(GCN_1^4(e_{u1}))}_{e_{u-c1}} + \underbrace{GCN_0^2(GCN_1^2(GCN_2^2(e_{u2})))}_{e_{u-c2}} \tag{15}$$

As the final embedding generated by the UGCN recommender can be interpreted as the hierarchical structure discussed in Section 3.2, we hence adopt the simplified expressions in our following discussion.

Formally, for user u, the gradient of the BPR loss *w.r.t.* the trainable embedding e_{u0} is as follows:

$$\frac{\partial BPR}{\partial e_{u0}} \propto \sum_{i \in \mathcal{N}_u} \sum_{j \notin \mathcal{N}_u} \frac{-e^{-\hat{x}_{uij}}}{1 + e^{-\hat{x}_{uij}}} \cdot \frac{\partial}{\partial e_{u0}} \hat{x}_{uij} - \lambda \tag{16}$$

According to the equation (10), $\hat{x}_{uij}$ can be further expanded as follows:

$$\hat{x}_{uij} = (e_{u-c0} + e_{u-c1} + e_{u-c2}) \cdot [(e_{i-c0} + e_{i-c1} + e_{i-c2}) - (e_{j-c0} + e_{j-c1} + e_{j-c2})] \tag{17}$$

As presented above, the initial group embeddings e_{u1} and e_{u2} in equation (15) can be calculated through a linear model contains e_{u0}, we can finally obtain:

$$\frac{\partial}{\partial e_{u0}} \hat{x}_{uij} = \alpha \cdot [(e_{i-c0} - e_{j-c0}) + (e_{i-c1} - e_{j-c1}) + (e_{i-c2} - e_{j-c2})] \tag{18}$$

, where α is the constant number computed through the partial derivatives of $(e_{u-c0} + e_{u-c1} + e_{u-c2})$ with respect to e_{u0}. Based on the equation (15) and (18), it is apparent that the U-shape hierarchical model can contain valuable group-level differences information during the model training process. Hence, we believe the UGCN model can produce more informative embeddings and generate better predictions.

4 EXPERIMENTS

We conduct experiments on three public million-size datasets to answer the following research questions.

- **Q1**: How does the UGCN recommender perform compared to state-of-the-art models? Does UGCN achieve higher accuracy?
- **Q2**: Are multi-resolution collaborative signals helpful for generating more personalized embedding? Can our model produce better recommendations?
- **Q3**: How do different hyper-parameter settings (e.g., the number of model's hierarchical levels) affect the recommendation performance?

4.1 Experimental Settings

Datasets We conduct experiments on three public million-size datasets, namely Movielens 1M (Ml1m), Gowalla, and Yelp2018. The detailed dataset statistics are shown in Table 1. For each dataset, we randomly split 80% of each user's interactions to construct the training set, while the remainder forms the test set. For each training set, we randomly select 10% of interactions as a validation set to tune hyperparameters. Moreover, according to our pair-wise BPR training strategy, each interaction of an individual user in the training set is treated as a positive sample, and the corresponding negative one will be randomly sampled from the non-interacted items of the same user.

Table 1. Statistics of experimental datasets.

Statistics	Users	Items	Interactions	Density
Ml1m	6022	3043	995, 154	0.05431
Gowalla	29, 858	40, 981	1, 027, 370	0.00084
Yelp2018	31, 668	38, 048	1, 561, 406	0.00130

Evaluation Metrics To evaluate different aspects of Top@K recommendation results for the proposed model and existing ones, we first adopt **Recall**@K and **NDCG**@K [10] to compare the overall prediction accuracy. Then, we apply an additional publicly accepted method (i.e., Coverage [3]) to assess the personalization performance. As coverage demonstrates the percentage of items that have been recommended to users, we adopt the following equations to compute both recommendation coverage $R-Coverage$ and hit coverage $H-Coverage$:

$$\begin{aligned} R-Coverage(K) &= N_{recommended}/N \\ H-Coverage(K) &= N_{Hit}/N \end{aligned} \tag{19}$$

, where $N_{recommended}$ and N_{Hit} imply the number of different items that have been recommended and hit in one or more users' top@K list, respectively; N is

the number of total items. According to common sense, users are always pleased with mixed recommendation results rather than merely recommending the same items[3]. Therefore, we aim to pursue a higher Coverage value.

Comparison Methods We compare UGCN with several state-of-the-art recommendation models.

- **MF-BPR** [21]: This model presented a generic optimization criterion BPR-Opt for personalized ranking.
- **NeuMF** [9]: This is a state-of-the-art neural Collaborative Filtering model that used implicit user-item interaction.
- **HHMF** [16]: This is a recent proposed hidden hierarchical recommender which learned from the user-item scoring record and does not need prior knowledge.
- **NGCF** [26]: This is a state-of-the-art GCN-based model which integrated the user-item interaction into the embedding process through the graph convolution operation.
- **NIA-GCN** [23]: This is a newly proposed GCN-based model which explicitly exploits the user-user and item-item relationships through a pairwise neighborhood aggregator.
- **LR-GCCF** [4]: This is a state-of-art GCN-based model which utilizes a residual network structure and linear graph convolution operations to alleviate the over-smoothing problem.
- **LightGCN** [8]: This is a newly proposed GCN-based model that simplifies the design of graph convolution operation and only includes neighbor aggregation in convolution operations.
- **SGL** [27]: This is a newly proposed Graph Training Strategy which implements on the LightGCN model and achieves competitive performance.

Parameter Settings The embedding size is fixed to 64 for all models, and the embedding parameters are initialized with the Xavier method [6]. The default learning rate is 0.002, and the default mini-batch size is 2048. The depth of hierarchical architecture is tested in the range of $[2, 3, 4]$. The number of graph convolution operations is tested from 1 to 3, and the predefined ratio α is validated in the range of [0.01 0.99]. The early stopping and validation strategies are the same as LightGCN. The Adam [12] optimizer is also employed and used in a mini-batch manner. Moreover, we also adopt dropout mechanisms on every GCN layer to mitigate over-fittingm and the default dropout rate is 0.6.

Implementation Details In the group formation step, we adopt the cosine function to derive the similarity among a pair of users (or items). For node merging in pooling operation, we mainly retain the common interactions of similar users in the same group and randomly delete some individual behaviors. Meanwhile, it is worth pointing out that the pooling operation would run separately and steply on user and item nodes. Moreover, the only trainable parameters of

the UGCN recommender are the initial user/item embeddings of the original user-item interaction graph. For the implementation of the baseline models, the default training strategies and hyperparameters settings from the corresponding referenced papers are followed.

4.2 Prediction Accuracy Comparison (QR1)

To evaluate the overall prediction accuracy, we test Recall@20 and NDCG@20 among all different models. The final results are presented in Table 2.

Table 2. The comparison of overall performance among UGCN and baseline methods.

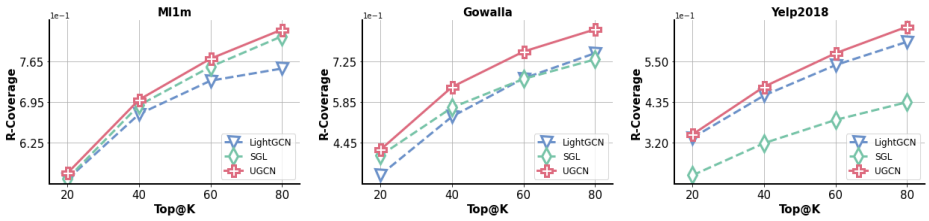
Data-	**Ml1m**		**Gowalla**		**Yelp2018**	
sets	Recall@20	NDCG@20	Recall@20	NDCG@20	Recall@20	NDCG@20
MF-BPR	0.2101	0.1787	0.1291	0.1109	0.0433	0.0354
NeuMF	0.2297	0.1886	0.1399	0.1212	0.0451	0.0363
HHMF	0.2311	0.2025	0.1477	0.1283	0.0498	0.0384
NGCF	0.2513	0.2511	0.1570	0.1327	0.0579	0.0477
NIA-GCN	0.2359	0.2243	0.1359	0.1358	0.0599	0.0491
LR-GCCF	0.2231	0.2124	0.1519	0.1358	0.0561	0.0343
LightGCN	0.2576	0.2427	<u>0.1830</u>	<u>0.1554</u>	0.0649	0.0530
SGL	<u>0.2700</u>	<u>0.2547</u>	0.1781	0.1501	<u>0.0674</u>	<u>0.0553</u>
UGCN	**0.2774**	**0.2633**	**0.1876**	**0.1587**	**0.0689**	**0.0561**
%Improv.	*+2.74%*	*+3.38%*	*+2.51%*	*+2.12%*	*+2.23%*	*+1.45%*

Our UGCN recommender consistently outperforms the baseline methods on all datasets. In particular, UGCN achieves 2.74%, 2.51%, and 2.23% improvement of recall@20 over the best baseline on Ml1m, Gowalla, and Yelp2018, respectively. Meanwhile, the performance of UGCN on NDCG@20 is also outstanding, presenting 3.38%, 2.51%, and 1.45% enhancements. Moreover, compared to MF-based and neural-based models, the significant improvements among GCN-based models reveal the superiority of GCNs in handling embedding learning tasks, which is consistent with the discussion in Section 2. The results also demonstrate that the HHMF model and UGCN recommender perform better than basic-MF and existing GCN-based models, respectively, confirming the effectiveness of exploring users/items hierarchies in recommender systems. In contrast to the HHMF model, the better performance of UGCN demonstrates that modeling the diverse resolution collaborative signal would be a better strategy than merely focusing on exploring user/item hierarchies separately. Moreover, since a denser dataset would result in more accurate user/item grouping and lead to more precise group-level signal extraction, this would explain why the most significant improvement is achieved in the ML1m dataset.

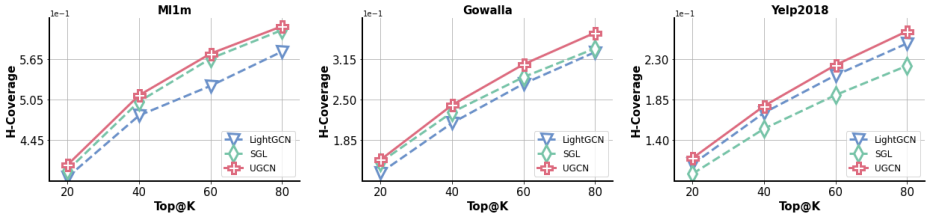
All the analyses above exhibit that our proposed UGCN recommender outperforms all the state-of-the-art models and achieves the highest recommendation accuracy.

4.3 Personalization Comparison (QR2)

As discussed above, we adopt Coverage [3] to demonstrate whether integrating diverse group-level collaborative signals can lead to better recommendations. Specifically, we adopt the state-of-the-art models LightGCN [8] and SGL [27] as our compared baseline owing to their competitive results displayed in Table 2. Moreover, we test three top@K recommendation cases, and the comparison results of Coverage among three models are presented in Fig. 2.



(a) R-Coverage Comparison Results among LightGCN, SGL, and UGCN



(b) H-Coverage Comparison Results among LightGCN, SGL, and UGCN

Fig. 2. Coverage Comparison Results

Based on the results in Fig.2, UGCN achieves higher $R-Coverage$ and $H-Coverage$ in all datasets. Taking the worst-case (top@20) as an example, $R-Coverage$ for UGCN are 57.34%, 42.20%, and 34.39%, which indicate the corresponding 1.60%, 4.95% and 2.81% improvement on three datasets compared to the best baseline. Similarly, $H-Coverage$ results also achieve relatively significant increases. Meanwhile, it is worth noticing that the $Coverage$ result of SGL on the Yelp2018 dataset is under-performed by those of LightGCN, even though SGL obtains a higher recommendation accuracy according to Table 2. These results demonstrate that higher recommendation accuracy might not reveal better recommendations. In comparison, the proposed UGCN recommender obtains higher recommendation accuracy and higher coverage simultaneously,

indicating the UGCN's superiority. Considering the difference between UGCN and those baseline models, we could answer Q2 by concluding that integrating multi-resolution collaborative allows recommender systems to precisely capture user/item personalized preferences and leads to better recommendations.

4.4 Hyper Parameter Analysis (QR3)

The above discussion indicates integrating diverse collaborative signals will benefit the recommender system. Nevertheless, it is still unclear how the number of model's hierarchical levels impact the recommendation performance? To this end, we conducted the comparison experiments on Ml1m and Gowalla. Owing to the limited space, we discard the results on Yelp2018, but the discussion would be applicable for other datasets. The results are presented in Fig. 3.

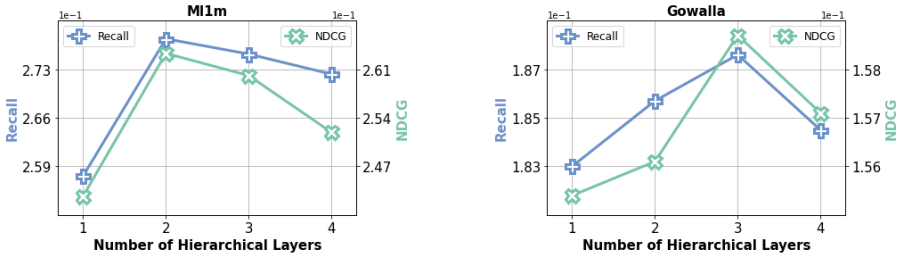


Fig. 3. Hyper-parameters Comparison Results

As shown in Fig. 3, the best performance is achieved when the number of the model's hierarchical levels are set to two and three on Ml1m and Gowalla, respectively. Compared to the one-level UGCN model (i.e., original LightGCN, which does not contain hierarchical architecture), the huge improvement confirms the necessity of modeling diverse collaborative signals. Meanwhile, the different optimal results on different datasets reveal that a fair number of UGCN's hierarchical levels are always needed to be optimized on specific datasets during training.

5 Conclusion

This paper proposes a graph convolutional network-based hierarchical recommender, namely UGCN, on a bipartite graph to capture diverse group-level collaborative signals and produce better recommendation predictions. Precisely, the proposed model utilizes a U-shaped architecture to gradually compress the original user-item interaction graph into smaller graphs by merging similar nodes together. Then, simplified graph convolutional operations can easily extract diverse group-level collaborative signals from compressed graphs. After that, those

learned informative signals would be projected back to the final user/item embeddings adaptively. Extensive and comprehensive experiments on three public datasets demonstrate that the UGCN recommender achieves significant overall prediction improvements over state-of-the-art models with better recommendations simultaneously.

6 Acknowledgment

The first author is funded by the China Scholarship Council (CSC) from the Ministry of Education of P.R. China.

References

1. Alhijawi, B., Awajan, A., Fraihat, S.: Survey on the objectives of recommender system: Measures, solutions, evaluation methodology, and new perspectives. ACM Computing Surveys (CSUR) (2022)
2. Baral, R., Iyengar, S.S., Zhu, X., Li, T., Sniatala, P.: Hirecs: A hierarchical contextual location recommendation system. IEEE Transactions on Computational Social Systems **6**(5), 1020–1037 (2019)
3. Cai, X., Hu, Z., Chen, J.: A many-objective optimization recommendation algorithm based on knowledge mining. Information Sciences **537**, 148–161 (2020)
4. Chen, L., Wu, L., Hong, R., Zhang, K., Wang, M.: Revisiting graph based collaborative filtering: A linear residual graph convolutional network approach. In: Proceedings of the AAAI. vol. 34, pp. 27–34 (2020)
5. Ebesu, T., Shen, B., Fang, Y.: Collaborative memory network for recommendation systems. In: The 41st international ACM SIGIR conference on research & development in information retrieval. pp. 515–524 (2018)
6. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: Proceedings of the thirteenth international conference on artificial intelligence and statistics. pp. 249–256. JMLR Workshop and Conference Proceedings (2010)
7. Han, Z., Xu, F., Shi, J., Shang, Y., Ma, H., Hui, P., Li, Y.: Genetic meta-structure search for recommendation on heterogeneous information network. In: Proceedings of the 29th ACM International Conference on Information & Knowledge Management. pp. 455–464 (2020)
8. He, X., Deng, K., Wang, X., Li, Y., Zhang, Y., Wang, M.: Lightgcn: Simplifying and powering graph convolution network for recommendation. In: Proceedings of the 43rd ACM SIGIR. pp. 639–648 (2020)
9. He, X., Liao, L., Zhang, H., Nie, L., Hu, X., Chua, T.S.: Neural collaborative filtering. In: Proceedings of the 26th international conference on world wide web. pp. 173–182. International World Wide Web Conferences Steering Committee (2017)
10. Isinkaye, F.O., Folajimi, Y.O., Ojokoh, B.A.: Recommendation systems: Principles, methods and evaluation. Egyptian informatics journal **16**(3), 261–273 (2015)
11. Jiang, J., Yang, D., Xiao, Y., Shen, C.: Convolutional gaussian embeddings for personalized recommendation with uncertainty. arXiv preprint arXiv:2006.10932 (2020)
12. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)

13. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)
14. Koren, Y.: Factorization meets the neighborhood: a multifaceted collaborative filtering model. In: Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 426–434. ACM (2008)
15. Koren, Y., Bell, R., Volinsky, C.: Matrix factorization techniques for recommender systems. Computer **42**(8), 30–37 (2009)
16. Li, H., Liu, Y., Qian, Y., Mamoulis, N., Tu, W., Cheung, D.W.: Hhmf: hidden hierarchical matrix factorization for recommender systems. Data Mining and Knowledge Discovery **33**(6), 1548–1582 (2019)
17. Liu, F., Cheng, Z., Zhu, L., Gao, Z., Nie, L.: Interest-aware message-passing gcn for recommendation. In: Proceedings of the Web Conference 2021. pp. 1296–1305 (2021)
18. Lu, K., Zhang, G., Li, R., Zhang, S., Wang, B.: Exploiting and exploring hierarchical structure in music recommendation. In: Asia Information Retrieval Symposium. pp. 211–225. Springer (2012)
19. Mnih, A., Salakhutdinov, R.R.: Probabilistic matrix factorization. Advances in neural information processing systems **20** (2007)
20. Nikolakopoulos, A.N., Kouneli, M.A., Garofalakis, J.D.: Hierarchical itemspace rank: Exploiting hierarchy to alleviate sparsity in ranking-based recommendation. Neurocomputing **163**, 126–136 (2015)
21. Rendle, S., Freudenthaler, C., Gantner, Z., Schmidt-Thieme, L.: Bpr: Bayesian personalized ranking from implicit feedback. arXiv preprint arXiv:1205.2618 (2012)
22. Schafer, J.B., Frankowski, D., Herlocker, J., Sen, S.: Collaborative filtering recommender systems. In: The adaptive web, pp. 291–324. Springer (2007)
23. Sun, J., Zhang, Y., Guo, W., Guo, H., Tang, R., He, X., Ma, C., Coates, M.: Neighbor interaction aware graph convolution networks for recommendation. In: Proceedings of the 43rd ACM SIGIR. pp. 1289–1298 (2020)
24. Wang, S., Tang, J., Wang, Y., Liu, H.: Exploring implicit hierarchical structures for recommender systems. In: Twenty-Fourth International Joint Conference on Artificial Intelligence (2015)
25. Wang, S., Tang, J., Wang, Y., Liu, H.: Exploring hierarchical structures for recommender systems. IEEE Transactions on Knowledge and Data Engineering **30**(6), 1022–1035 (2018)
26. Wang, X., He, X., Wang, M., Feng, F., Chua, T.S.: Neural graph collaborative filtering. In: Proceedings of the 42nd ACM SIGIR. pp. 165–174 (2019)
27. Wu, J., Wang, X., Feng, F., He, X., Chen, L., Lian, J., Xie, X.: Self-supervised graph learning for recommendation. In: Proceedings of the 44th ACM SIGIR. pp. 726–735 (2021)
28. Xue, H.J., Dai, X., Zhang, J., Huang, S., Chen, J.: Deep matrix factorization models for recommender systems. In: IJCAI. pp. 3203–3209 (2017)
29. Yang, J., Sun, Z., Bozzon, A., Zhang, J.: Learning hierarchical feature influence for recommendation by recursive regularization. In: Proceedings of the 10th ACM Conference on Recommender Systems. pp. 51–58 (2016)
30. Zhang, S., Yao, L., Sun, A., Tay, Y.: Deep learning based recommender system: A survey and new perspectives. ACM Computing Surveys (CSUR) **52**(1), 1–38 (2019)
31. Zhuang, F., Luo, D., Yuan, N.J., Xie, X., He, Q.: Representation learning with pair-wise constraints for collaborative ranking. In: Proceedings of the tenth ACM international conference on web search and data mining. pp. 567–575 (2017)