# Graph Heterogeneous Multi-Relational Recommendation

**Chong Chen [1], Weizhi Ma [1], Min Zhang [1]*, Zhaowei Wang [2], Xiuqiang He [2], Chenyang Wang [1], Yiqun Liu[1] and Shaoping Ma**

[1] Department of Computer Science and Technology, Institute for Artificial Intelligence,
Beijing National Research Center for Information Science and Technology, Tsinghua University
[2] Huawei Noah's Ark Lab
cc17@mails.tsinghua.edu.cn, z-m@tsinghua.edu.cn

## Abstract

Traditional studies on recommender systems usually leverage only one type of user behaviors (the optimization target, such as purchase), despite the fact that users also generate a large number of various types of interaction data (e.g., view, click, add-to-cart, etc). Generally, these heterogeneous multi-relational data provide well-structured information and can be used for high-quality recommendation. Early efforts towards leveraging these heterogeneous data fail to capture the high-hop structure of user-item interactions, which are unable to make full use of them and may only achieve constrained recommendation performance. In this work, we propose a new multi-relational recommendation model named Graph Heterogeneous Collaborative Filtering (GHCF). To explore the high-hop heterogeneous user-item interactions, we take the advantages of Graph Convolutional Network (GCN) and further improve it to jointly embed both representations of nodes (users and items) and relations for multi-relational prediction. Moreover, to fully utilize the whole heterogeneous data, we perform the advanced efficient non-sampling optimization under a multi-task learning framework. Experimental results on two public benchmarks show that GHCF significantly outperforms the state-of-the-art recommendation methods, especially for cold-start users who have few primary item interactions. Further analysis verifies the importance of the proposed embedding propagation for modelling high-hop heterogeneous user-item interactions, showing the rationality and effectiveness of GHCF. Our implementation has been released (https://github.com/chenchongthu/GHCF).

## Introduction

Recommender systems have been widely deployed in today's web platforms and applications, serving as important tools to alleviate the information overload issue and improve user experience (Ricci, Rokach, and Shapira 2011; Chen et al. 2018). To provide more accurate recommendations, it is a trending topic to take more user preference related information into account (Chen et al. 2019a, 2020b). In real-world information systems, although the systems often choose "click" or "purchase" as the optimization target, there are also various types of user behaviors, such as view,

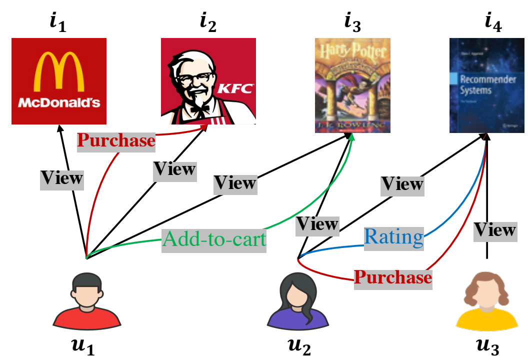


Figure 1: An example of multiple types of user feedback. High-hop connectivity contains rich semantic features that carry collaborative signals. E.g., the 3-hop heterogeneous connections between $u_1$ and $i_4$ contain $u_1 \overset{view}{\rightarrow} i_3 \overset{view}{\leftarrow} u_2 \overset{purcahse}{\rightarrow} i_4$, $u_1 \overset{cart}{\rightarrow} i_3 \overset{view}{\leftarrow} u_2 \overset{rating}{\rightarrow} i_4$, etc.

add-to-cart, etc. Figure 1 shows an example of heterogeneous user behaviors in E-commerce scenarios. Users can view an item, add an item to shopping cart, and purchase an item, etc. These heterogeneous behaviors provide valuable signals of user preference, which are helpful for building a fine-grained recommender system (Gao et al. 2019; Pan, Liu, and Ming 2016; Chen et al. 2020d; Krohn-Grimberghe et al. 2012).

To leverage these heterogeneous feedback data, several efforts on multi-relational recommender systems have been made, showing the superior performance in terms of learning user preference (Ding et al. 2018; Gao et al. 2019; Chen et al. 2020d). However, summarizing existing multi-relational recommendation methods, a common drawback can be found: these methods follow the typical Collaborative Filtering (CF) learning scheme, which lacks an explicit encoding of the high-hop graph structure of user-item heterogeneous interactions. As shown in Figure 1, high-hop connectivity also contains rich semantics that carry collaborative signals. For example, $u_1$ and $i_4$ have several 3-hop heterogeneous connections (e.g.,$u_1 \overset{view}{\rightarrow} i_3 \overset{view}{\leftarrow} u_2 \overset{purcahse}{\rightarrow} i_4$). This suggests that $u_1$ is likely to adopt $i_4$ , since his similar user $u_2$ has viewed, purchased, and rated $i_4$ before.


*Corresponding author.

However, the high-hop heterogeneous connections have not been well-utilized in previous recommendation work, which compromises the model's effectiveness. Although some recent studies have tried to introduce Graph Convolutional Network (GCN) for recommendation task (Wang et al. 2019a,b,c; He et al. 2020), they only focus on leveraging user-item homogeneous graph with only one type of user behavior. There lacks in-depth investigation of users' heterogeneous behaviors.

Motivated by the above observations, we propose to construct a unified heterogeneous graph based on multiple types of behavioral data. We further propose a novel model named Graph Heterogeneous Collaborative Filtering (GHCF), which not only seamlessly incorporates the auxiliary user behaviors into recommendation, but also considers the high-hop connectivities among the heterogeneous feedback data. Specifically, different from existing GCN applications which are either restricted to non-relational graph setting (Bruna et al. 2013; Velickovic et al. 2017) or limited to learning only node representations (Marcheggiani and Titov 2017; Schlichtkrull et al. 2018), the GCN propagation layer in GHCF is further enhanced to jointly embed both representations of nodes (user and item) and relations for multi-relational prediction. Besides, we perform multi-task learning with the advanced efficient non-sampling optimization (Chen et al. 2019b, 2020c) in model training. In contrast to sampling, non-sampling strategy computes the gradient over the whole data (including all non-observed data) and can easily converge to a better optimum in a more stable way (Xin et al. 2018; Wang et al. 2018). Through these designs, our GHCF method effectively addresses the main challenges and helps to exploit auxiliary behaviors for a better prediction on the target behavior. The main contributions of this work are as follows:

- We propose a novel neural model named GHCF for multi-relational recommendation, which uncovers the underlying relationships among heterogeneous user-item interactions and shows multi-task ability to predict various types of user behaviors using one unified model.
- We design relation-aware GCN propagation layers, which jointly embed both representations of nodes (users and items) and relations in a graph to explicitly exploit the collaborative high-hop signals.
- Extensive experiments are conducted on two benchmark datasets. The results show that GHCF consistently and significantly outperforms the state-of-the-art recommendation models, especially for cold-start users.

# Related Work

## Multi-relational Recommendation

Multi-relational (or multi-behavior) recommendation is an emerging branch in the research community of recommender systems, which aims to leverage multiple user behavior data to improve the recommendation performance on the target behavior (Gao et al. 2019; Chen et al. 2020d; Jin et al. 2020; Zhou et al. 2019). Early research naturally extends the Matrix Factorization (MF) methods to perform multiple learning of different behaviors (Tang et al. 2016; Krohn-Grimberghe et al. 2012; Singh and Gordon 2008). Another line of research addresses the problem from the perspective of learning, which considers multiple types of behaviors by changing the negative sampling strategy and enriching the training set from the auxiliary behavioral data (Ding et al. 2018; Loni et al. 2016; Qiu et al. 2018). Recently, there are also some researchers attempt to develop neural network models to capture the complicated and multi-type interactions between users and items (Gao et al. 2019; Chen et al. 2020d). For example, Chen et. al (Chen et al. 2020d) propose an Efficient Heterogeneous Collaborative Filtering model (EHCF), which correlates the prediction of each user behavior in a transfer way for multi-relational recommendation. Summarizing existing multi-relational recommendation methods, they lack an explicit encoding of the high-hop graph structure of user-item heterogeneous interactions, which is the main concern of our GHCF model.

## Graph-based Recommendation

In recent years, Graph Neural Networks (GNNs) have achieved great success due to the powerful capability on representation learning from structured data (Bruna et al. 2013; Hamilton, Ying, and Leskovec 2017; Velickovic et al. 2017). Recently, GNNs have attracted increasing attention in recommendation. For example, GC-MC (Den Berg, Kipf, and Welling 2017) applies graph convolution network on user-item graph, which employs one convolutional layer to exploit the direct connections between users and items. PinSage (Ying et al. 2018) combines random walks with multiple graph convolutional layers on the item-item graph for image recommendation. SpectralCF (Zheng et al. 2018) proposes a spectral convolution operation to discover all possible connectivity between users and items in the spectral domain. NGCF (Wang et al. 2019c) exploits high-order proximity by propagating embeddings on the user-item interaction graph. NGCF is further extended to LightGCN (He et al. 2020) by removing the non-linear activation function and feature transformation in embedding propagation layers to improve the performance of CF tasks. Besides these works on user-item interaction data, there are also GNN models for recommendation with side information, such as social-aware recommendation (Fan et al. 2019) and knowledge enhanced recommendation (Wang et al. 2019b). In this paper, we present a graph heterogeneous collaborative filtering model, which incorporates heterogeneous feedback data in graph convolutional networks for recommendation with multiple user behaviors.

# Preliminaries

## Problem Formulation

We denote the user and item sets as $\mathbf{U}$ and $\mathbf{V}$, respectively. We use $u$ to denote a user, and $v$ to denote an item. The user-item heterogeneous interactions are denoted as $\{\mathbf{Y}_{(1)}, \mathbf{Y}_{(2)}, ..., \mathbf{Y}_{(K)}\}$, where $\mathbf{Y}_{(k)} = [y_{(k)uv}]_{|\mathbf{U}|\times|\mathbf{V}|} \in \{0, 1\}$ indicates whether user $u$ has interacted with item $v$ under behavior $k$, and $K$ is the number of user behavior types. Generally, multi-relational recommendation has a target behavior to be optimized, which we denote as $\mathbf{Y}_{(K)}$. An example of the target behavior is the purchase behavior in E-commerce, and other behaviors include view, click, add-to-cart, etc. Given a target user $u$, the multi-relational recommendation task is to estimate the likelihood $\hat{y}_{(K)uv}$ that a user $u$ will interact with an item $v$ under the target behavior. The items (uninteracted under the target behavior) are ranked in descending order of $\hat{y}_{(K)uv}$ to provide the Top-N item recommendation list.

## Graph Convolutional Networks

Most existing research on graph convolutional networks (Bruna et al. 2013; Hamilton, Ying, and Leskovec 2017; Velickovic et al. 2017) are focused on learning representations of nodes in simple undirected graphs. Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ denotes the set of nodes and $\mathcal{E}$ denotes the set of edges, respectively. The node representation obtained from a single GCN layer is defined as:

$$\mathbf{E} = \sigma(\hat{\mathbf{A}}\mathbf{E}^{(0)}\mathbf{W}) \tag{1}$$

where $\hat{\mathbf{A}} = \mathbf{D}^{-\frac{1}{2}}(\mathbf{A}+\mathbf{I})\mathbf{D}^{-\frac{1}{2}}$ is the normalized adjacency matrix with added self-connections and $\mathbf{D}$ is a diagonal degree matrix, which is defined as $\mathbf{D}_{ii} = \sum_j(\mathbf{A}+\mathbf{I})_{ij}$; $\mathbf{I}$ denotes an identity matrix; $\mathbf{E}^{(0)}$ is the set $\mathbf{E}$ at the initial message-passing iteration. The model parameter is denoted as $\mathbf{W}$ and $\sigma$ is an activation function. The GCN representation $\mathbf{E}$ encodes the immediate neighborhood of each node in the graph. For capturing high-hop dependencies in the graph, several GCN layers can be stacked as:

$$\mathbf{E}^{(l)} = \sigma(\hat{\mathbf{A}}\mathbf{E}^{(l-1)}\mathbf{W}^{(l)}) \tag{2}$$

where $l$ denotes the number of layers and $\mathbf{W}^{(l)}$ is layer-specific parameter.

For a relational graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{R})$ where $\mathcal{R}$ denotes the set of relations, a commonly used GCN formulation is as follows (Marcheggiani and Titov 2017; Schlichtkrull et al. 2018):

$$\mathbf{E}^{(l)} = \sigma(\hat{\mathbf{A}}\mathbf{E}^{(l-1)}\mathbf{W}_r^{(l)}) \tag{3}$$

where $\mathbf{W}_r^{(l)}$ is the relation specific parameters of the model. However, this formulation leads to over-parameterization and embeds only nodes in the graph. Thus it need to be improved to support multi-relational recommendation.

# Graph Heterogeneous Collaborative Filtering

In this section, we present the proposed GHCF model. The overall architecture is described in Figure 2, which has three important components: 1) Embedding propagation layers, which embed both nodes and relations in heterogeneous user-item interaction data; 2) Multi-task prediction module, which predicts the likelihood that a user will interact with an item under each relation type; 3) Efficient non-sampling learning module to achieve more effective and stable model optimization.

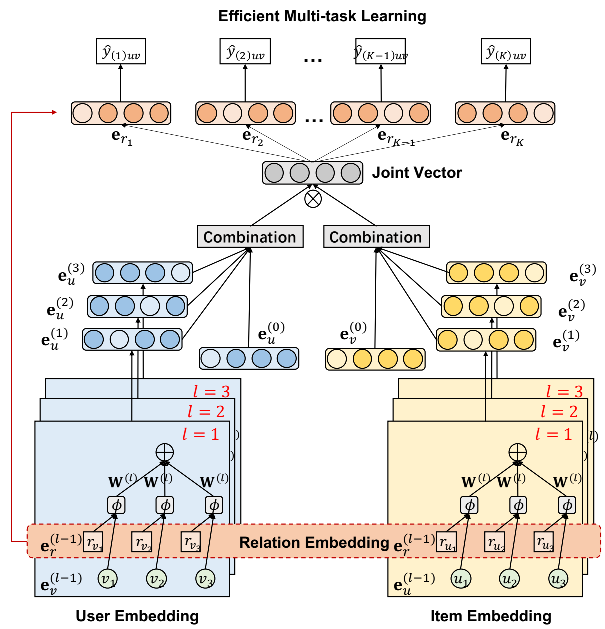


Figure 2: An illustration of GHCF model.

## Embedding Propagation Layers

The embedding propagation layers in our model are built upon the message-passing architecture of GCNs (Bruna et al. 2013; Hamilton, Ying, and Leskovec 2017; Velickovic et al. 2017) to capture the CF signals along with the graph structure of user-item heterogeneous interactions. The basic idea of GCNs is to learn representation for nodes by smoothing features over the graph. In our model, the representation of a user (or item) is modeled by accumulating the incoming messages from all the heterogeneous interacted items (or users). A general method to achieve the above target is like Eq. (3), which can be re-written as:

$$\mathbf{e}_u^{(l)} = \sigma\Big(\sum_{(v,r)\in\mathcal{N}(u)} \frac{1}{\sqrt{|\mathcal{N}_u|\,|\mathcal{N}_v|}}\mathbf{W}_r^{(l)}\mathbf{e}_v^{(l-1)}\Big) \tag{4}$$

where $\mathcal{N}(u)$ and $\mathcal{N}(v)$ are the set of immediate neighbors of $u$ and $v$, respectively; $\mathbf{W}_r^{(l)}$ is the relation specific parameters of the model; The symmetric normalization term $\frac{1}{\sqrt{|\mathcal{N}_u||\mathcal{N}_v|}}$ is used to avoid the scale of embeddings increasing with graph convolution operations. However, this formulation suffers from over-parameterization and embeds only nodes in the graph.

To address the above issues, in our model we perform composition ($\phi$) of a neighboring node $v$ with respect to its relation $r$ to model the relational user-item interactions. Inspired by entity-relation composition operations used in knowledge graph embedding approaches (Bordes et al. 2013; Vashishth et al. 2019), the message passing equation of our model is defined as:

$$\mathbf{e}_u^{(l)} = \sigma\Big(\sum_{(v,r)\in\mathcal{N}(u)} \frac{1}{\sqrt{|\mathcal{N}_u|\,|\mathcal{N}_v|}}\mathbf{W}^{(l)}\phi(\mathbf{e}_v^{(l-1)}, \mathbf{e}_r^{(l-1)})\Big) \tag{5}$$

where $\mathbf{W}^{(l)}$ is layer-specific, $\phi$ is a composition operator to incorporate relation embeddings into the GCN formulation. The activation function $\sigma$ is LeakyReLU (Maas, Hannun, and Ng 2013). Eq. (5) allows our model to be relation-aware while being linear $(O(|\mathcal{R}|d))$ in the number of feature dimensions. Specifically, in our model the composition operator is defined as:

$$\phi(\mathbf{e}_v, \mathbf{e}_r) = \mathbf{e}_v \odot \mathbf{e}_r \tag{6}$$

where $\odot$ denotes the element-wise product of vectors. Note that other composition methods like subtraction (Bordes et al. 2013) and neural network approaches (He et al. 2017; Socher et al. 2013) can also be applied, we leave it as future work.

It is worth noting that in our model, we aggregate only the connected neighbors and do not integrate the target node itself (i.e., self-connection). This is also adopted in LightGCN (He et al. 2020), which shows that through the layer combination operation (to be introduced in the next subsection), the model has already captured the same effect as self-connections in this way.

After the node embedding update defined in Eq. (5), the relation embeddings are also transformed as follows:

$$\mathbf{e}_r^{(l)} = \mathbf{W}_{rel}^{(l)} \mathbf{e}_r^{(l-1)} \tag{7}$$

where $\mathbf{W}_{rel}^{(l)}$ is a layer-specific parameter which projects all the relations to the same embedding space as nodes and allows them to be utilized in the next GCN layer.

For the first-hop propagation, $\mathbf{e}_u^{(0)}$, $\mathbf{e}_v^{(0)}$, $\mathbf{e}_r^{(0)}$ are initial features for node $u$, $v$ and relation $r$ respectively, which is generated through an ID embedding layer.

## Multi-task Prediction

After propagating with $L$ layers, we obtain multiple representations for user $u$, item $v$, and relation $r$. The representations obtained from different layers emphasize the information passed from different hops. E.g., the first layer enforces smoothness on users and items that have interactions, the second layer smooths users (items) that have overlap on interacted items (users), and higher-layers capture higher-order proximity (He et al. 2020; Wang et al. 2019c). Thus we further combine them to get the final representations:

$$\mathbf{e}_u = \sum_{l=0}^{L} \frac{1}{L+1} \mathbf{e}_u^{(l)};\ \mathbf{e}_v = \sum_{l=0}^{L} \frac{1}{L+1} \mathbf{e}_v^{(l)};\ \mathbf{e}_r = \sum_{l=0}^{L} \frac{1}{L+1} \mathbf{e}_r^{(l)} \tag{8}$$

Note that a uniform weight $1/(L+1)$ is set to each embedding layer, which leads to good performance in general. Other weighting strategies such as attention mechanisms (Vaswani et al. 2017) can also be applied, we leave it as future work.

To predict the likelihood of users' multiple behaviors on items, the learnt representation of each behavior is incorporated as a separated prediction layer. Specifically, let $\mathbf{e}_{r_k}$ denotes the learnt representation of the $k-$th behavior, the likelihood that user $u$ will perform the $k$-th behavior on item $v$ is estimated by:

$$\hat{y}_{(k)uv} = \mathbf{e}_u^T \cdot diag(\mathbf{e}_{r_k}) \cdot \mathbf{e}_v = \sum_{i}^{d} e_{u,i} e_{r_k,i} e_{v,i} \tag{9}$$

where $diag(\mathbf{e}_{r_k})$ denotes a diagonal matrix whose diagonal elements equal to $\mathbf{e}_{r_k}$ correspondingly and $d$ denotes the embedding size.

## Efficient Multi-task Learning without Sampling

To learn model parameters in a more effective and stable way, we apply the efficient non-sampling learning (Chen et al. 2020c) to optimize our GHCF model. It is a recently proposed learning method and has been shown to be superior in both effectiveness and efficiency than traditional sampling-based learning methods (Chen et al. 2020a,c,d) (e.g., Bayesian Personalized Ranking loss (Rendle et al. 2009)). Take a single $k$-th behavior as an example, for a batch of users $\mathbf{B}$ and the whole item set $\mathbf{V}$, the traditional weighted regression loss is:

$$\mathcal{L}_k(\Theta) = \sum_{u \in \mathbf{B}} \sum_{v \in \mathbf{V}} c_{uv}^k (y_{(k)uv} - \hat{y}_{(k)uv})^2 \tag{10}$$

where $c_{uv}^k$ denotes the weight of entry $y_{(k)uv}$. As can be seen, the time complexity of computing this loss is $O(|\mathbf{B}||\mathbf{V}|d)$, which is generally unaffordable in practice. Based on the derivation of previous work (Chen et al. 2020c,d), if the instance weight $c_{uv}^k$ is simplified to $c_v^k$, a more efficient form of Eq. (10) can be obtained, which is:

$$\begin{aligned}\tilde{\mathcal{L}}_k(\Theta) = &\sum_{u \in \mathbf{B}} \sum_{v \in \mathbf{V}_{(u)}^{k+}} \Big( (c_v^{k+} - c_v^{k-}) \hat{y}_{(k)uv}^2 - 2 c_v^{k+} \hat{y}_{(k)uv} \Big) \\ &+ \sum_{i=1}^{d} \sum_{j=1}^{d} \left( (e_{r_k,i} e_{r_k,j}) \left( \sum_{u \in \mathbf{B}} e_{u,i} e_{u,j} \right) \left( \sum_{v \in \mathbf{V}} c_v^{k-} e_{v,i} e_{v,j} \right) \right)\end{aligned} \tag{11}$$

where $\mathbf{V}_{(u)}^{k+}$ denotes the interacted items of user $u$ under the behavior $k$. The complexity of Eq.(11) is $O((|\mathbf{B}| + |\mathbf{V}|)d^2 + |\mathbf{V}^{k+}|d)$. Since $|\mathbf{V}^{k+}|$ is the number of positive user-item interactions under the $k$-th behavior and $|\mathbf{V}^{k+}| \ll |\mathbf{B}||\mathbf{V}|$ in practice, the complexity is reduced significantly compared with Eq. (10). The proof can be made by reformulating the expensive loss over all negative instances using a partition and a decouple operation, which largely follows from that in (Chen et al. 2020c,d) with little variations.

Multi-task learning (MTL) is a paradigm that performs joint training on the models of different but correlated tasks, so as to obtain a better model for each task (Argyriou, Evgeniou, and Pontil 2007). To better learn parameters from all the heterogeneous data, we propose a MTL objective function defined as follows:

$$\mathcal{L}(\Theta) = \sum_{k=1}^{K} \lambda_k \tilde{\mathcal{L}}_k(\Theta) + \mu \|\Theta\|_2^2 \tag{12}$$

where $K$ is the number of types of users' behavior, $\lambda_k$ is added to control the influence of the $k$-th behavior on the joint training, which is a hyper-parameter to be specified for different datasets. We additionally enforce that $\sum_{k=1}^{K} \lambda_k = 1$ to facilitate the tuning of these hyper-parameters. $L_2$ regularization parameterized by $\mu$ on $\Theta$ is conducted to prevent overfitting.

| Dataset | #User | #Item | #View | #Add-to-cart | #Purchase |
|---|---|---|---|---|---|
| ***Beibei*** | 21,716 | 7,977 | 2,412,586 | 642,622 | 304,576 |
| ***Taobao*** | 48,749 | 39,493 | 1,548,126 | 193,747 | 259,747 |

Table 1: Statistical details of the evaluation datasets.

To optimize the objective function, we use mini-batch Adam (Kingma and Ba 2014) as the optimizer. Its main advantage is that the learning rate can be self-adaptive during the training phase, which eases the pain of choosing a proper learning rate. Dropout is an effective solution to prevent neural networks from overfitting (Srivastava et al. 2014). We propose to employ two widely used dropout methods: message dropout and node dropout in our model.

# Experiments

## Experimental Settings

**Datasets** We experiment with two real-world e-commerce datasets: ***Beibei*** and ***Taobao*** [2]. The two datasets contain three types of user behaviors, including view, add-to-cart, and purchase. The target behavior of the recommendation task is purchase. The two datasets are preprocessed to filter out users and items with less than 5 purchase interactions. After that, the last purchase records of users are used as test data, the second last records are used as validation data, and the remaining records are used for training. Note that for objective comparison, in our experiments the two datasets are exactly the same as those used in (Chen et al. 2020d) [3], in which the split datasets are publicly available. The statistical details of the datasets are summarized in Table 1.

**Baselines** To demonstrate the effectiveness of our **GHCF** model, we compare it with several state-of-the-art methods. The baselines are classified into two categories based on whether they utilize single-behavior or heterogeneous data. The compared single-behavior methods include:

- **BPR** (Rendle et al. 2009), a widely used pairwise learning method for item recommendation.
- **NCF** (He et al. 2017), a state-of-the-art deep learning method which combines MF with a multilayer perceptron (MLP) model for item ranking.
- **ENMF** (Chen et al. 2020c), a state-of-the-art non-sampling recommendation method for Top-N recommendation.
- **LightGCN** (He et al. 2020), a state-of-the-art graph neural network model which simplifies the design of GNN to make it more appropriate for recommendation.

The second category that leverages heterogeneous data are as follows:

- **CMF** (Zhao et al. 2015), it decomposes the data matrices of multiple behavior types simultaneously.
- **MC-BPR** (Loni et al. 2016), it adapts the negative sampling rule in BPR for heterogeneous data.
- **NMTR** (Gao et al. 2019), a state-of-the-art method which combines the recent advances of NCF modeling and the efficacy of multi-task learning.
- **EHCF** (Chen et al. 2020d), a state-of-the-art method which correlates the prediction of each behavior in a transfer way and adopts non-sampling learning for multi-relational recommendation.

**Evaluation Methodology** All experiments are run on the same machine (Intel Xeon 8-Core CPU of 2.4 GHz and single NVIDIA GeForce GTX TITAN X GPU) for fair comparison. We apply the widely used leave-one-out technique (Gao et al. 2019; Rendle et al. 2009; Chen et al. 2020d) and then adopt two popular metrics, HR (Hit Ratio) and NDCG (Normalized Discounted Cumulative Gain), to judge the performance of the ranking list. HR is a recall-based metric, measuring whether the testing item is in the Top-N list, while NDCG is position-sensitive, which assigns higher scores to hits at higher positions. For each user, our evaluation protocol ranks all the items except the positive ones in the training set. In this way, the obtained results are more persuasive than ranking a random subset of negative items only (Krichene and Rendle 2020). For each method, we randomly initialize the model and run it five times. After that, we report the average results.

**Parameter settings** We search for the optimal parameters on validation data and evaluate the model on test data. The parameters for all baseline methods are initialized as in the corresponding papers, and are then carefully tuned to achieve optimal performances. After the tuning process, the batch size is set to 256, the size of the latent factor dimension $d$ is set to 64. The learning rate is set to 0.001. We set the negative sampling ratio as 4 for sampling-based methods, an empirical value that shows good performance. For non-sampling methods ENMF, EHCF and our GHCF, the negative weight is set to 0.01 for Beibei and 0.1 for Taobao. The number of graph layers is set to 4, and the dropout ratio was set to 0.8 for Beibei and Taobao to prevent overfitting. Our implementation has been released (https://github.com/chenchongthu/GHCF).

## Performance Comparison

The performance comparison results are presented in Table 2. To evaluate on different recommendation lengths, we set the length N = 10, 50, and 100 in our experiments. From the results, the following observations can be made:

First and foremost, our proposed GHCF achieves the best performance on the two datasets, significantly outperforming all the state-of-the-art baseline methods with p-values smaller than 0.01. The average improvement of our model to the best baseline EHCF is 16.9% on Beibei dataset and 14.2% on Taobao dataset, which verifies the effectiveness of our model. The substantial improvements can be attributed to two reasons: 1) the proposed relation-aware GCN layers, which explicitly exploit the collaborative high-hop signals; 2) the efficient non-sampling learning module, which is more effective and stable than traditional negative sampling learning strategy.

[2] https://tianchi.aliyun.com/dataset/dataDetail?dataId=649
[3] https://github.com/chenchongthu/EHCF

| *Beibei* | | HR@10 | HR@50 | HR@100 | NDCG@10 | NDCG@50 | NDCG@100 |
|---|---|---|---|---|---|---|---|
| **Single-behavior** | **BPR** | 0.0437 | 0.1246 | 0.2192 | 0.0213 | 0.0407 | 0.0539 |
| | **NCF** | 0.0441 | 0.1562 | 0.2343 | 0.0225 | 0.0445 | 0.0584 |
| | **ENMF** | 0.0464 | 0.1637 | 0.2586 | 0.0247 | 0.0484 | 0.0639 |
| | **LightGCN** | 0.0451 | 0.1613 | 0.2495 | 0.0232 | 0.0466 | 0.0611 |
| **Heterogeneous-behavior** | **CMF** | 0.0482 | 0.1582 | 0.2843 | 0.0251 | 0.0462 | 0.0661 |
| | **MC-BPR** | 0.0504 | 0.1743 | 0.2755 | 0.0254 | 0.0503 | 0.0653 |
| | **NMTR** | 0.0524 | 0.2047 | 0.3189 | 0.0285 | 0.0609 | 0.0764 |
| | **EHCF** | 0.1523 | 0.3316 | 0.4312 | 0.0817 | 0.1213 | 0.1374 |
| | **GHCF** | **0.1922**** | **0.3794**** | **0.4711**** | **0.1012**** | **0.1426**** | **0.1575**** |
| *Taobao* | | HR@10 | HR@50 | HR@100 | NDCG@10 | NDCG@50 | NDCG@100 |
| **Single-behavior** | **BPR** | 0.0376 | 0.0708 | 0.0871 | 0.0227 | 0.0269 | 0.0305 |
| | **NCF** | 0.0391 | 0.0728 | 0.0897 | 0.0233 | 0.0281 | 0.0321 |
| | **ENMF** | 0.0398 | 0.0743 | 0.0936 | 0.0244 | 0.0298 | 0.0339 |
| | **LightGCN** | 0.0415 | 0.0814 | 0.1025 | 0.0237 | 0.0325 | 0.0359 |
| **Heterogeneous-behavior** | **CMF** | 0.0483 | 0.0774 | 0.1185 | 0.0252 | 0.0293 | 0.0357 |
| | **MC-BPR** | 0.0547 | 0.0791 | 0.1264 | 0.0263 | 0.0297 | 0.0361 |
| | **NMTR** | 0.0585 | 0.0942 | 0.1368 | 0.0278 | 0.0334 | 0.0394 |
| | **EHCF** | 0.0717 | 0.1618 | 0.2211 | 0.0403 | 0.0594 | 0.0690 |
| | **GHCF** | **0.0807**** | **0.1892**** | **0.2599**** | **0.0442**** | **0.0678**** | **0.0792**** |

Table 2: Performance of different models on two datasets. ** denotes the statistical significance for $p < 0.01$ compared to the best baseline. Note that the results of EHCF are consist with those reported in (Chen et al. 2020d) since we share exactly the same data splits and experimental settings.

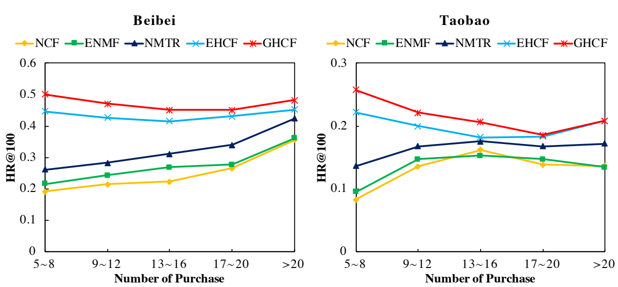


Figure 3: Performances of NCF, ENMF, NMTR, EHCF, and our GHCF on users with different number of purchase records.

Second, the methods using heterogeneous feedback data generally outperform methods that only making use of purchase behavior, which shows the complementarity of user heterogeneous feedback. Compared to the best single-behavior baseline, our GHCF exhibits remarkable average improvements of 208% on Beibei dataset and 116% on Taobao dataset, which clarifies the necessity of introducing heterogeneous feedback data.

Third, the methods with non-sampling learning strategy (ENMF, EHCF, and GHCF) generally perform better than sampling-based methods, especially for multi-relational recommendation task. This is consistent with previous work (Chen et al. 2020d; Gao et al. 2019). Although negative sampling is a widely-used learning strategy, it has been shown not suitable for learning from heterogeneous behavior data (Chen et al. 2020d). To generate a training instance, sampling-based methods (e.g., MC-BPR, NMTR) generally need to sample a negative instance for every observed interaction (regardless of the behavior type). This produces a much larger randomness in total ($K$ times than single-behavior scenario) and would inevitably lead to information loss. This explains why non-sampling methods EHCF and GHCF outperform the state-of-the-art sampling-based method NMTR substantially.

## Handling Data Sparsity Issue

Data sparsity is a big challenge in recommendation (Volkovs, Yu, and Poutanen 2017) because it is hard to establish optimal representations for inactive users with few interactions. Multi-relational recommendation which utilizes auxiliary behavior data provides a solution to alleviate the data sparsity issue. Thus we further investigate how our GHCF model performs for the users with few records of target behavior. Figure 3 illustrates the results w.r.t. HR@100 on different user groups in Beibei and Taobao. For other metrics, the observations are similar.

From the figure, we can see that our GHCF consistently outperforms other models including the state-of-the-art multi-relational methods like NMTR and EHCF, especially for the first user group with only 5-8 purchase records. Some methods have a slight descent in the middle, we think it is because of the size difference of auxiliary behavioral data. For example, on Taobao dataset the number of auxiliary behavioral records for users who have 5-8 purchase records is much more than users who have 17-20 purchase records. Typically, the data of low-level behaviors (e.g., view) is easier to collect and has a larger volume than the target behavior (e.g., purchase). The results indicate the effectiveness of leveraging auxiliary behavior to alleviate the data sparsity issue and the strong power of our GHCF model.

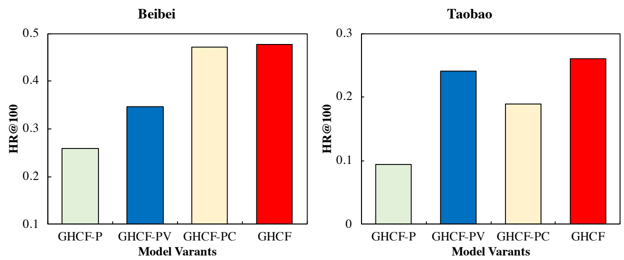


Figure 4: Effect of auxiliary behavior data.

| | *Beibei* | | *Taobao* | |
|---|---|---|---|---|
| | **HR@100** | **NDCG@100** | **HR@100** | **NDCG@100** |
| GHCF-1 | 0.4569 | 0.1494 | 0.2473 | 0.0755 |
| GHCF-2 | 0.4636 | 0.1498 | 0.2501 | 0.0778 |
| GHCF-3 | 0.4674 | 0.1551 | 0.2567 | 0.0787 |
| GHCF-4 | 0.4711 | 0.1575 | 0.2599 | 0.0792 |
| GHCF-5 | 0.4681 | 0.1554 | 0.2558 | 0.0782 |

Table 3: Effect of embedding propagation layer numbers

## Ablation Study

To understand the effectiveness of auxiliary behavior data, we conduct experiments with several variants of GHCF:

- GHCF-P: The variant model of GHCF which utilizes only purchase data.
- GHCF-PV: The variant model of GHCF which utilizes purchase data and view data.
- GHCF-PC: The variant model of GHCF which utilizes purchase data and carting data.

Figure 4 shows the performance of different variants. Due to the space limitation, we only show the results on HR@100 metrics. For other metrics, the observations are similar. As shown in the figure, both adding view data and carting data lead to better recommendation performance. When using all the three behavior data, the performance of our GHCF is further improved. This verifies the effectiveness of auxiliary behaviors for user preference modeling. Moreover, we observe that on Beibei dataset, carting behavior contributes more than that on Taobao dataset. The reason may be the size difference of auxiliary behavioral data in the two datasets.

## Effect of Layer Numbers

To investigate whether GHCF can benefit from multiple embedding propagation layers, we vary the model depth. In particular, we search the layer numbers in the range of [1, 2, 3, 4, 5]. Table 3 summarizes the experimental results, where GHCF-1 indicates the model with one embedding propagation layers, and similar notations for others. From the table, we can see that by increasing the depth of GHCF from one to four, the recommendation results on both Beibei and Taobao datasets are improved. Generally, four propagation layers are sufficient to capture the heterogeneous signals. Deeper layers might introduce noise and lead to overfitting. Moreover, when varying the number of propagation layers, GHCF is consistently superior to other methods on the two datasets. The above observations verify the effectiveness of GHCF and empirically show that explicit modeling of high-order heterogeneous connections can greatly facilitate the recommendation task.

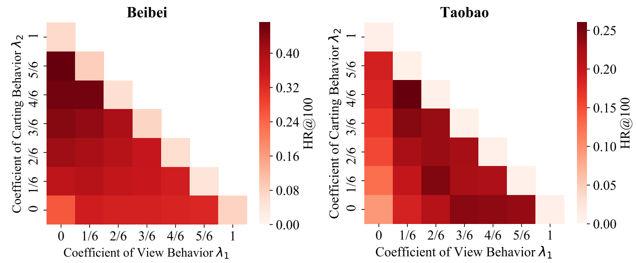


Figure 5: Performance of GHCF with different loss coefficient.

## Effect of Loss Coefficient

As the coefficient parameter $\lambda_k$ in the multi-task loss function plays a pivotal role in GHCF, we investigate its impact on the performance. There are three behavior types in Beibei and Taobao (view, add-to-cart, and purchase), which means there are three loss coefficients $\lambda_1$, $\lambda_2$, and $\lambda_3$, respectively. As $\lambda_1 + \lambda_2 + \lambda_3 = 1$, when $\lambda_1$ and $\lambda_2$ are given, the value of $\lambda_3$ is determined. We tune the three coefficients in $[0, 1/6, 2/6, 3/6, 4/6, 5/6, 1]$ and plot the results of HR@100 in Figure 5 where darker block means better performance. In the figure, outermost blocks are rather shallow since they represent a zero $\lambda_3$, which is the coefficient of the target behavior (purchase). For both datasets, the best performances of our GHCF are achieved at almost the same setting: $(1/6, 4/6, 1/6)$. On Beibei dataset, a relatively large coefficient of carting behavior outperforms that of view behavior. While on Taobao dataset, a relative large $\lambda_1$ generally performs better. We think that it is due to the size difference of auxiliary behavioral data in the two datasets.

# Conclusions

In this work, we study the problem of multi-relational recommendation that considers multiple types of user-item interactions. We propose a novel end-to-end model GHCF, which achieves the target by modelling high-order heterogeneous connectivities in the user-item integration graph. Different from most existing GCN methods, the embedding propagation layers in our model leverage a composition operator to jointly embed both representations of nodes (users and items) and relations for multi-relational prediction. Moreover, we adopt an efficient non-sampling learning module to achieve more effective and stable model optimization. Extensive experiments on two real-world datasets show that GHCF consistently and significantly outperforms the state-of-the-art recommendation models on different evaluation metrics, especially for cold-start users that have few target interactions. Future work includes exploring our GHCF model in other related tasks such as network embedding and multi-label classification.

## Acknowledgements

This work is supported by the National Key Research and Development Program of China (2018YFC0831900), Natural Science Foundation of China (Grant No. 62002191, 61672311, 61532011) and Tsinghua University Guoqiang Research Institute. This project is also funded by China Postdoctoral Science Foundation (2020M670339). Chong Chen is supported by ByteDance Scholars Program and Dr Weizhi Ma is supported by Shuimu Tsinghua Scholar Program.

## Ethical and Code Statement

Hereby, we consciously assure that:

- The paper reflects the authors' own research and analysis in a truthful and complete manner.
- No portion of this paper has been previously published.
- This paper is not being considered for publication elsewhere.
- This paper has identified and acknowledged all sources used in the creation of the paper, including any graphics, images, tables, and figures, and also including any persons who do not meet the criteria for authorship.
- We have notified AAAI of any conflicts of interest we might have with regard to the work.
- All authors have been personally and actively involved in substantial work leading to the paper, and will take public responsibility for its content.

## References

Argyriou, A.; Evgeniou, T.; and Pontil, M. 2007. Multi-task feature learning. In *Proceedings of NeurIPS*, 41–48.

Bordes, A.; Usunier, N.; Garcia-Duran, A.; Weston, J.; and Yakhnenko, O. 2013. Translating embeddings for modeling multi-relational data. In *Proceedings of NeurIPS*.

Bruna, J.; Zaremba, W.; Szlam, A.; and Lecun, Y. 2013. Spectral Networks and Locally Connected Networks on Graphs. *arXiv: Learning* .

Chen, C.; Zhang, M.; Liu, Y.; and Ma, S. 2018. Neural Attentional Rating Regression with Review-level Explanations. In *Proceedings of WWW*, 1583–1592.

Chen, C.; Zhang, M.; Liu, Y.; and Ma, S. 2019a. Social Attentional Memory Network: Modeling Aspect-and Friend-level Differences in Recommendation. In *Proceedings of WSDM*.

Chen, C.; Zhang, M.; Ma, W.; Liu, Y.; and Ma, S. 2020a. Efficient Non-Sampling Factorization Machines for Optimal Context-Aware Recommendation. In *Proceedings of WWW*, 2400–2410.

Chen, C.; Zhang, M.; Ma, W.; Liu, Y.; and Ma, S. 2020b. Jointly Non-Sampling Learning for Knowledge Graph Enhanced Recommendation. Proceedings of SIGIR.

Chen, C.; Zhang, M.; Wang, C.; Ma, W.; Li, M.; Liu, Y.; and Ma, S. 2019b. An Efficient Adaptive Transfer Neural Network for Social-aware Recommendation. In *Proceedings of SIGIR*, 225–234. ACM.

Chen, C.; Zhang, M.; Zhang, Y.; Liu, Y.; and Ma, S. 2020c. Efficient Neural Matrix Factorization without Sampling for Recommendation. *ACM Trans. Inf. Syst.* 38(2). ISSN 1046-8188.

Chen, C.; Zhang, M.; Zhang, Y.; Ma, W.; Liu, Y.; and Ma, S. 2020d. Efficient Heterogeneous Collaborative Filtering without Negative Sampling for Recommendation. In *Proceedings of AAAI*.

Den Berg, R. V.; Kipf, T.; and Welling, M. 2017. Graph Convolutional Matrix Completion. *arXiv: Machine Learning* .

Ding, J.; Yu, G.; He, X.; Quan, Y.; Li, Y.; Chua, T.-S.; Jin, D.; and Yu, J. 2018. Improving Implicit Recommender Systems with View Data. In *Proceedings of IJCAI*, 3343–3349.

Fan, W.; Ma, Y.; Li, Q.; He, Y.; Zhao, E.; Tang, J.; and Yin, D. 2019. Graph Neural Networks for Social Recommendation .

Gao, C.; He, X.; Gan, D.; Chen, X.; Feng, F.; Li, Y.; Chua, T.-S.; and Jin, D. 2019. Neural Multi-Task Recommendation from Multi-Behavior Data. In *Proceedings of ICDE*.

Hamilton, W. L.; Ying, Z.; and Leskovec, J. 2017. Inductive Representation Learning on Large Graphs 1024–1034.

He, X.; Deng, K.; Wang, X.; Li, Y.; Zhang, Y.; and Wang, M. 2020. LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation. *arXiv preprint arXiv:2002.02126* .

He, X.; Liao, L.; Zhang, H.; Nie, L.; Hu, X.; and Chua, T.-S. 2017. Neural collaborative filtering. In *Proceedings of WWW*, 173–182.

Jin, B.; Gao, C.; He, X.; Jin, D.; and Li, Y. 2020. Multi-behavior Recommendation with Graph Convolutional Networks. In *Proceedings of SIGIR*, 659–668.

Kingma, D. P.; and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* .

Krichene, W.; and Rendle, S. 2020. On Sampled Metrics for Item Recommendation. In *Proceedings of KDD*.

Krohn-Grimberghe, A.; Drumond, L.; Freudenthaler, C.; and Schmidt-Thieme, L. 2012. Multi-relational matrix factorization using bayesian personalized ranking for social network data. In *Proceedings of WSDM*, 173–182.

Loni, B.; Pagano, R.; Larson, M.; and Hanjalic, A. 2016. Bayesian personalized ranking with multi-channel user feedback. In *Proceedings of RecSys*, 361–364.

Maas, A. L.; Hannun, A. Y.; and Ng, A. Y. 2013. Rectifier nonlinearities improve neural network acoustic models. In *Proc. icml*, volume 30, 3.

Marcheggiani, D.; and Titov, I. 2017. Encoding Sentences with Graph Convolutional Networks for Semantic Role Labeling 1506–1515.

Pan, W.; Liu, M.; and Ming, Z. 2016. Transfer learning for heterogeneous one-class collaborative filtering. *IEEE Intelligent Systems* 31(4): 43–49.

Qiu, H.; Liu, Y.; Guo, G.; Sun, Z.; Zhang, J.; and Nguyen, H. T. 2018. BPRH: Bayesian personalized ranking for heterogeneous implicit feedback. *Information Sciences* 453.

Rendle, S.; Freudenthaler, C.; Gantner, Z.; and Schmidt-Thieme, L. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of UAI*, 452–461.

Ricci, F.; Rokach, L.; and Shapira, B. 2011. Introduction to recommender systems handbook. In *Recommender systems handbook*, 1–35. Springer.

Schlichtkrull, M. S.; Kipf, T.; Bloem, P.; Den Berg, R. V.; Titov, I.; and Welling, M. 2018. Modeling Relational Data with Graph Convolutional Networks 593–607.

Singh, A. P.; and Gordon, G. J. 2008. Relational learning via collective matrix factorization. In *Proceedings of KDD*.

Socher, R.; Chen, D.; Manning, C. D.; and Ng, A. 2013. Reasoning with neural tensor networks for knowledge base completion. In *Advances in neural information processing systems*, 926–934.

Srivastava, N.; Hinton, G. E.; Krizhevsky, A.; Sutskever, I.; and Salakhutdinov, R. 2014. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research* 15(1): 1929–1958.

Tang, L.; Long, B.; Chen, B.-C.; and Agarwal, D. 2016. An empirical study on recommendation with multiple types of feedback. In *Proceedings of SIGKDD*, 283–292.

Vashishth, S.; Sanyal, S.; Nitin, V.; and Talukdar, P. 2019. Composition-based Multi-Relational Graph Convolutional Networks. *arXiv: Learning* .

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *Proceedings of NeurIPS*, 5998–6008.

Velickovic, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio, P.; and Bengio, Y. 2017. Graph Attention Networks. *arXiv: Machine Learning* .

Volkovs, M.; Yu, G.; and Poutanen, T. 2017. Dropoutnet: Addressing cold start in recommender systems. In *Proceedings of NeurIPS*, 4957–4966.

Wang, H.; Zhao, M.; Xie, X.; Li, W.; and Guo, M. 2019a. Knowledge graph convolutional networks for recommender systems. In *Proceedings of WWW*, 3307–3313. ACM.

Wang, M.; Gong, M.; Zheng, X.; and Zhang, K. 2018. Modeling Dynamic Missingness of Implicit Feedback for Recommendation. In *Proceedings of NeurIPS*, 6669–6678.

Wang, X.; He, X.; Cao, Y.; Liu, M.; and Chua, T.-S. 2019b. KGAT: Knowledge Graph Attention Network for Recommendation. In *Proceedings of KDD*.

Wang, X.; He, X.; Wang, M.; Feng, F.; and Chua, T. 2019c. Neural Graph Collaborative Filtering 165–174.

Xin, X.; Yuan, F.; He, X.; and Jose, J. M. 2018. Batch IS NOT Heavy: LearningWord Representations From All Samples .

Ying, R.; He, R.; Chen, K.; Eksombatchai, P.; Hamilton, W. L.; and Leskovec, J. 2018. Graph Convolutional Neural Networks for Web-Scale Recommender Systems 974–983.

Zhao, Z.; Cheng, Z.; Hong, L.; and Chi, E. H. 2015. Improving user topic interest profiles by behavior factorization. In *Proceedings of WWW*, 1406–1416.

Zheng, L.; Lu, C.; Jiang, F.; Zhang, J.; and Yu, P. S. 2018. Spectral collaborative filtering 311–319.

Zhou, X.; Liu, D.; Lian, J.; and Xie, X. 2019. Collaborative metric learning with memory network for multi-relational recommender systems. In *Proceedings of IJCAI*, 4454–4460.