Neural Graph Filtering for Context-aware Recommendation

Chuanyan Zhang CHUANYAN_ZHANG@SINA.CN
School of Computer Science and Technology, Shandong University, Jinan, China, 250101

Xiaoguang Hong HXG@SDU.EDU.CN
Qilu Software College of Shandong University, Jinan, China, 250101

Editors: Vineeth N Balasubramanian and Ivor Tsang

Abstract

With the rapid development of web services, various kinds of context data become available in recommender systems to handler the data sparsity problem, called context-aware recommendation (CAR). It is challenging to develop effective approaches to model and exploit these various and heterogeneous data. Recently, heterogeneous information network (HIN) has been adopted to model the context data due to its flexibility in modelling data heterogeneity. However, most of the HIN-based methods, which rely on meta paths or graph embedding to extract features from HINs, cannot fully mine the network structure and semantic features of users and items. Besides, these methods, utilizing the global dataset to learn personalized latent factors, usually suffer individuality loss problem. In this paper, we propose a neural graph filtering method for context-aware recommendation, called NGF. First, we use an unified HIN to model both the users' feedback information and the context data. Then, we adopt graph filtering to predict aspect-level ratings on a series of independent subgraphs of the unified HIN and feed a deep neural network (DNN) to fuse the predictions for CAR. Concretely, graph filtering is a case-by-case algorithm for personalized recommendation on HINs, which predicts the further behavior by all its similar historical behaviors. We split the unified HIN into many single-aspect networks according to the semantic relations and utilize graph filtering to predict user's behavior on each subgraphs. The following deep neural network is to fuse the personalized predictions in aspect-level. Extensive experiments on two real-world datasets demonstrate the effectiveness of our neural graph filtering for CAR.

Keywords: Context-aware recommendation; Graph filtering; Deep neural network; Heterogeneous information network

1. Introduction

Main contents here. In recent years, recommender systems, which help users discover items of interest from a large resource collection, have been playing an increasingly important role in various online services. Traditional recommendation methods (e.g., matrix factorization) mainly aim to learn an effective prediction function for characterizing user-item feedback records (e.g., user-item rating matrix). Comparing with a mass of users and near-infinite items of a recommender system, the users' feedback data are too much sparse, which seriously restricts the effectiveness of the traditional recommendation methods [Hu et al. (2019)]. With the rapid development of web services, various kinds of context data (a.k.a., side information) become available in recommender systems. As shown in Fig.1, these context data contain but not limit attributes of users and items, social relationships, date time and locations. Exploiting the context data in recommender system be a promising direction to handler the data sparsity problem, called context-aware recommendation (CAR) [Xin et al. (2019)]. Although context data is likely to contain useful information for recommendation, it is difficult to model and utilize these heterogeneous and complex information in recommender systems. Furthermore, it is more challenging to develop a relatively general approach to model these varying data in different systems or platforms.

Generally, CAR methods fall into two categories according to the way of data modeling: (1) Factorization machine (FM); (2) HIN-based methods. FM is a general predictor working with any real valued feature vectors [Liang et al. (2020)]. To exploit the context data, FMs first adopt one/multi-hot encoding technology to model users' feedback records with their corresponding context data into high-dimensional generic feature vectors, and then estimate the target by modelling all interactions between each pair of features via factorized interaction parameters. The following variants of FM improve in two ways: embedding the high-dimensional feature vectors and utilizing deep neural network (DNN) to explore higher-order feature interactions [He and Chua (2017),Xin et al. (2019)]. Although FMs have achieved remarkable performances for CAR, they are still limited by their modeling capability. One/multi-hot encoding can easily model attribute information in the context data, but cannot model the structure information, e.g., social relationships. Due to the flexibility of HIN in modeling data heterogeneity, HIN has been adopted in recommender systems to characterize rich context data, called HIN-based recommendation [Shi et al. (2019)]. Recently, HIN-based recommendation methods have received much attention in the literature and achieved performance improvement to some extent. In Fig. 1, we present an example for movie recommendation characterized by HINs. We can see that the HIN contains multiple types of entities connected by different types of relations. Compared with one/multi-hot encoding, HINs can easily model both attribute information and structure information.

For the HIN-based methods, there are two way to extract latent features from HINs and exploit them for recommendation. The basic idea of most existing HIN-based recommendation methods is to leverage path based semantic relatedness between users and items over HINs, e.g., meta path based similarities [Sun et al. (2011)], and integrate the similarities into complex matrix factorization (MF) or DNN models for recommendation [Luo et al. (2014), Han et al. (2018)]. Given a HIN, meta path based similarities, e.g., PathSim [Sun et al. (2011)], can be easily calculated. However, these similarity measures rely on specified meta path, may not be reliable to used for recommendation when path connections are sparse. In other words, meta-path based similarity, as a kind of local distance measure on HIN, cannot fully min the latent relevance between nodes. Further, the derived similarities have no explicit impact on the recommendation in some case, and may not be directly applicable to recommender systems [Shi et al. (2019)]. Although the graph embedding approach itself is more resistant to sparse data than meta-path similarity, it also suffers information loss problem. Generally, the learned embedding vectors are more compact, the more information lost. In fact, graph embedding is to translate the nodes of HINs into low-dimension space according to their structural relevance [Maheshwari et al. (2021)]. As the same to meta path based similarity, the structural relevance may not be valuable to the recommendation performance in some case. Finally, we investigate the learning models, including complex MF and DNN, and find that all the models of CAR systems are to learn personalized predictors based on the global datasets. We call them *global learning models*. Specifically, MF is to learn the latent feature vectors for each user and item; DNN models take the identification vector of user and item as input to train a perceptron. Although the global learning models may achieve a global optimal solution under a specific objective function, they will suffer individuality loss problem. An obvious example is that most of existing recommender systems tend to recommend popular items to any user since most users choose them [Li et al. (2021)]. The personal interests that are different from public will be downplayed or neglected in the global learning process, and the final predictor mainly persist universal characteristics. Above problems essentially reflect two fundamental issues, namely *valuable information extraction from HINs* and *prediction without individuality loss*.

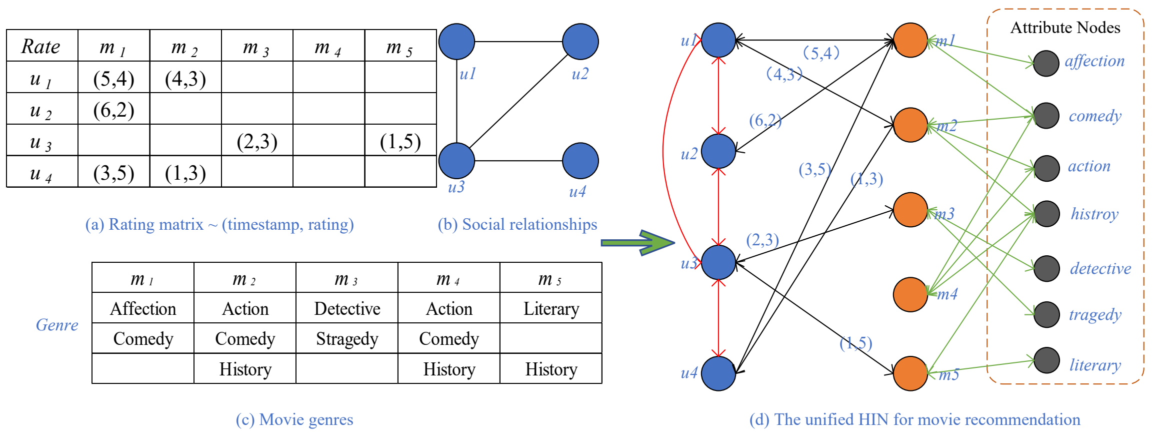
In this paper, we challenge the problems of HIN-based recommendation and propose a novel solution for CAR, called *neural graph filtering* (NGF). we first use an unified HIN to model both the user feedback information and the context information. Then, we adopt graph filtering to predict type-level ratings on a series of independent subgraphs of the unified HIN and feed a deep neural network to fuse the predictions for CAR. Concretely, graph filtering is a case-by-case algorithm for personalized recommendation on HINs, which predicts the further behavior by all its similar historical behaviors. We use *rating pair* to represent user's behavior on HINs and measure the rating pair similarity based on *General SimRank* (GSR), which is extended from SimRank and can fully exploit the structure and semantic information for a global similarity computation on HINs [Zhang et al. (2020)]. Distinct from traditional HIN based recommendation methods, graph filtering directly calculates the prediction results for recommendation, rather than extracting similarities or learning embedding vectors which may have no explicit impact to recommendation. To weight the contributions of different typed data in HINs, we split it into many independent single-aspect networks according to the semantic relations and utilize graph filtering to predict user's behavior on each subgraphs. The following deep neural network is to fuse the personalized predictions in type-level. We conduct extensive experiments to evaluate our proposed algorithms, as well as other state-of-the-art recommendation techniques, using two real datasets. The experimental results demonstrate the superiority of our methods in various measurements.

The rest of this paper is organized as follows. We give the preliminaries of NGF in Section 2. In Section 3, we present an overview of graph filtering. In Section 4, we give the neural graph filtering algorithm for CAR. Extensive experiments on two real datasets are presented in Section 5. We review the related work in Section 6. Finally, we offer our concluding remarks in the last section.

2. Preliminary

In this section, we introduce the basic preliminaries of HIN and a straightforward way to model the recommendation data. Then, we define *Rating Pair* to represent user's behavior on HIN. In Final, we formulates our problem.

Definition 1 (Information Network): An information network is defined as a directed graph $G(V, E)$ where each object $v \in V$ belongs to one particular object type $\phi(v) \in A$, and each edge $e \in E$ belongs to one particular relation $\varphi(e) \in R$. If the



Rate	m_1	m_2	m_3	m_4	m_5
u_1	(5,4)	(4,3)			
u_2	(6,2)				
u_3			(2,3)		(1,5)
u_4	(3,5)	(1,3)			

(a) Rating matrix ~ (timestamp, rating)

Genre	m_1	m_2	m_3	m_4	m_5
	Affection	Action	Detective	Action	Literary
	Comedy	Comedy	Stragedy	Comedy	
		History		History	History

(c) Movie genres

Figure 1: Example of creating the unified HIN for movie recommendation based on rating matrix, user relationships and the movie genres.

size of A or E is bigger than 1, the network G is called *heterogeneous information network*; otherwise, it is a *homogeneous information network*.

We can model various recommendation data via an unified HIN in the following way. First, we create a HIN $G_R(V, E)$ with $A = \{user, item\}$. The edge between user and item represents user's behavior information and two weights (t, r) are set for each edge. Suppose $e(u, i) \in E$, $u \in user$, $i \in item$ and the weights are (t_{ui}, r_{ui}) . The weight t_{ui} indicates when the behavior happened and r_{ui} is the rating score. For example, the movie-rating matrix, as the users' behavior information, represents the rating which user assign to movie they have watched. As shown in Fig.1, u_1 has watched m_1 in timestamp 6 and gives a rating 4. Thus, we have an edge $e(u_1, m_1)$ with weights $(1, 4)$. If not otherwise specified, the edges of HIN in this paper are all bidirectional and the weights are symmetric. Then, we will integrate the context data into $G_R(V, E)$. Generally, the context data fall into two categories: structural and attribute information. Structural information, representing the relations of objects, e.g., social friendships, can be modelled as edge in $G_R(V, E)$. For the attribute information, a kind of appended data to corresponding objects, we create new attribute nodes and edges to model them. Given an attribute a_i, if its value domain is discrete, e.g., the "genre" of movie in Fig.1, we model each attribute value $a_{i,j}$ (i.e., the j-th value of a_i) as a node and create an attribute edge $e_a(v, a_{i,j})$ if the node v has an attribute a_i with value $a_{i,j}$; if the value domain is continuous, e.g., "price" of product, we segment the value domain into discrete fragments. For each fragment, we can create corresponding node and edges in the same way. Finally, we get an unified HIN, denoted as $G_U(V, E)$, which integrates both the user feedback data and context data in graph model.

Given G_U, we use *Rating pair* to represent user's behavior.

Definition 2 (Rating Pair): Rating pair is a two-tuples about user and item of $G_U(V, E)$, describing a behavior of $u, u \in user$, related to a specific item $i, i \in item$, denoted as $r(u, i)$. If there is an edge between u and i, $r(u, i)$ is called *Feedback Rating Pair* with the corresponding rating r_{ui}; otherwise, it's *Predicting Rating Pair*, and the predicted score is denoted as $\hat{r}_{ui}$.

Now we define our task as follows.

Problem 1. Given $G_U(V, E)$ that integrates both the user feedback data and the context data, the context-aware recommendation is to compute $\hat{r}$ for every predicting rating pair based on the feedback rating pairs though exploiting $G_U(V, E)$.

3. Overview of Graph Filtering

3.1. General SimRank

The intuition of graph filtering is that "a behavior can be predicted by its similar historical behaviors." Given an information network, various link-based measures are useful for computation of similarity. However, most of them are designed for homogeneous information network, likely random walk with restart, personalized PageRank and SimRank [Jeh and Widom (2002)]. Without the capability of semantic recognition, they cannot work well on HINs which contain rich semantics behind different edges. PathSim first studies the semantic problem and proposes a peer similarity measure on HIN. Since it compute similarity based on specific meta path, PathSim is a kind of local measure. That is to say it cannot fully exploit the structure and semantic information of HINs [Sun et al. (2011)]. Recently, General SimRank (GSR), as the general form of the famous SimRank is proposed. GSR can distinguish the various semantics and computer similarities based on the global structure of HINs [Zhang et al. (2020)].

SimRank first formally introduced the famous intuition that "two objects are similar if they are related to similar objects." Given an information network $G(V, E)$, SimRank score of a node pair (u, v), denoted as $s(u, v)$, specifies how soon two random surfers are expected to meet at the same node,

$$(u, v) = \begin{cases} 1, & u = v \\ \frac{c}{|I(u)||I(v)|} \sum_{i=1}^{|I(u)|} \sum_{j=1}^{|I(v)|} s(I_i(u), I_j(v)), & u \neq v \end{cases} \tag{1}$$

where c is a decay factor, $0 < c < 1$, $I(*)$ is the in-neighbor set of $*$, $I_i(*)$ is the i-th in-neighbor and $|I(*)|$ is the size of $I(*)$. Further, General SimRank (GSR) proposes the general form of SimRank via the intuition that "two objects are similar if they are related to similar objects of the same type." Formally, GSR calculates $s(u, v)$ by

$$s(u, v) = \begin{cases} 1, & u = v \\ \frac{c}{|I(u)||I(v)|} \sum_{k=1}^{|A(u,v)|} \sum_{i=1}^{|I_k(u)|} \sum_{j=1}^{|I_k(v)|} s(I_{k,i}(u), I_{k,j}(v)), & u \neq v \end{cases} \tag{2}$$

where $A(u, v)$ it the type set of in-neighbors of u and v, $I_k(*)$ is the in-neighbor set of node $*$ with $\phi(*) = A_k$ and $I_{k,i}$ is the i-th node of the set $I_k(*)$.

GSR has some good properties, likely symmetric, self-maximum and semantic aware. With these good properties, General SimRank can give reasonable global similarity scores on HINs. However, GSR does not consider the weights of HINs. Based on our definition of $G_U(V, E)$, there are two types of weight in the unified HIN: time weight t and rating weight r, which only exist in the edges between users and items. We denote these edges as

$R(user, item)$. The weights will impact the transition probability from node pair (u, v) to its neighbor pair $(I_{k,i}(u), I_{k,j}(v))$ at the same time if $e(u, I_{k,i}(u)) \in R(user, item)$. Obviously, if two edges have similar timestamps and rating scores, the transition probability will be bigger; on the contrary, it will be smaller. Based on this intuition, we design a decay function of weights for the GSR random surfers, denoted as $f(u, v, k, i, j)$,

$$f(u,v,k,i,j) = \begin{cases} exp(-\alpha \frac{(t_{u,I_{k,i}(u)} - t_{v,I_{k,j}(v)})^2}{\sigma_t^2} - \beta \frac{(r_{u,I_{k,i}(u)} - r_{v,I_{k,j}(v)})^2}{\sigma_r^2}), & \exists\ t, r \\ 1, & otherwise \end{cases} \tag{3}$$

where α and β are the decay factors for time weight and rating weight respectively, $0 \leq \alpha, \beta \leq 1$, and σ_t^2 and σ_r^2 are their variances.

Further, we rewrite the transition probability for node pair (u, v) to its neighbor pair $(I_{k,i}(u), I_{k,j}(v))$, denoted as $p(u, v, k, i, j)$,

$$p(u,v,k,i,j) = \frac{cf(u,v,k,i,j)}{|I(u)|\,|I(v)|} \tag{4}$$

Finally, the weight-aware General SimRank on HINs is computed by

$$s(u,v) = \begin{cases} 1, & u = v \\ \sum\limits_{k=1}^{|A(u,v)|} \sum\limits_{i=1}^{|I_k(u)|} \sum\limits_{j=1}^{|I_k(v)|} p(u,v,k,i,j) s(I_{k,i}(u), I_{k,j}(v)), & u \neq v \end{cases} \tag{5}$$

3.2. Graph Filtering

Graph Filtering predicts user's further behavior based on its similar historical behaviors. Since we use rating pair to represent user's behavior on the unified HIN, we should define the rating pair similarity firstly. Given two rating pairs $r(u, i)$ and $r(v, j)$, the similarity between them, denoted as $s[(u, i), (v, j)]$, is computed by

$$s[(u,i),(v,j)] = s(u,v)s(i,j) \tag{6}$$

where $s(*, *)$ is calculated by Eq. 5.

Since rating pair is a two-tuples, we calculate rating pair similarity in Eq. 6 via measuring the similarities of the corresponding objects, i.e., user similarity and item similarity. Given G_U, we can computer $s(u, v)$ and $s(i, j)$ through weight-aware GSR of Eq. 5, and then get $s[(u, i), (v, j)]$. Let study some special cases. If $u = v$, the two rating pairs have the same user and the difference between them only relies on the items. Meanwhile, the corresponding equation $s[(u, i), (v, j)] = s(i, j)$ since $s(u, v) = 1$. Hence, our rating pair similarity of Eq. 6 satisfies the basic cognition about similarity measure. In the same way, if $i = j$, we have $s[(u, i), (v, j)] = s(u, v)$. That is also reasonable.

The rating pair similarity also have some good properties:

- *Symmetric*: $s[(u, i), (v, j)] = s[(v, j), (u, i)]$;
- *Self-maximum*: $s[(u, i), (v, j)] \in [0, 1]$; $s[(u, i), (v, j)] = 1$ only if $(u, i) = (v, j)$;

- *Semantic aware*: Eq. 6 have the capability of semantic recognition and can calculate similarities on HINs;
- *Weight aware*: Eq. 6 considers the time and rating score for similarity computation.

Proof.

1. Since $s[(u,i),(v,j)] = s(u,v)s(i,j)$, $s[(v,j),(u,i)] = s(v,u)s(j,i)$, and GSR score is symmetric, we have $s[(u,i),(v,j)] = s[(v,j),(u,i)]$.
2. Since the value field of GSR score is $[0,1]$, we can get $s[(u,i),(v,j)] \in [0,1]$ due to Eq. 6. If $(u,i) = (v,j)$, $s(u,v)$ and $s(i,j)$ take the maximum value 1. Otherwise, $s(*,*) < 1$. Thus, $s[(u,i),(v,j)]$ is self-maximum.
3. The GSR score in Eq. 5 is risen by the similarities of same-typed neighbor pairs. Based on the iterative definition in Eq. 5, we can infer that every step in any pairwise meeting path from (u,v) to (x,x) has the same semantic. Hence, the GSR is semantic aware. Based on Eq. 6, we can further infer that the rating pair similarity is also semantic aware.
4. There are two weight factor for each rating pair, i.e., time and rating. Let Δt and Δr be the absolute difference values of time weights and rating weights respectively. For the user pair (u,v), they must have neighbors of item type. We can infer that $s(v,u)$ is monotone decreasing about Δt and Δr since $f(u,v,k,i,j)$ is a monotone decreasing function about Δt and Δr when $\exists t, r$. In the same way, we can infer that $s(v,u)$ is monotone decreasing about Δt and Δr. Hence, we get that the rating pair similarity measure of Eq. 6 is monotone decreasing about Δt and Δr. That is to say, given two rating pair, if the differences of time weight and rating weights are bigger, the similarity between rating pairs will be smaller. On the contrary, the similarity score will be bigger. Hence, we can infer that Eq. 6 is weight aware. □

With these good properties, Eq. 6 can give a reasonable rating pair similarity in term of global structure, semantic and weights of HINs. Based on Eq. 6, we can predict user's behavior through its similar historical behaviors. Formally, graph filtering calculates $\hat{r}_{ui}$ for a given predicting rating pair $r(u,i)$ by

$$\hat{r}_{ui} = \bar{r}_u + \frac{\sum\limits_{r(v,j)\in FRP} s[(u,i),(v,j)](r_{vj} - \bar{r}_v)}{\sum\limits_{r(v,j)\in FRP} s[(u,i),(v,j)]} \tag{7}$$

where FRP is the feedback rating pair set of $G_U(V,E)$, and $\bar{r}_u$ is the average rating score of user u.

Graph filtering gives an basic solution to predict user's further behavior based on HIN. Distinct from the *global learning models*, graph filtering does not min the latent feature for individual objects or personalized perceptron through the global dataset, but predict each rating pair only by its similar rating pairs, which maximizes the individuality. To overcome the data sparsity problem, graph filtering use link-based similarity measure, rather than vector-based similarity of CF, to find more similar historical behaviors. Further, weight-aware GSR can fully exploit the unified HIN in term of global structure, semantics and weights.

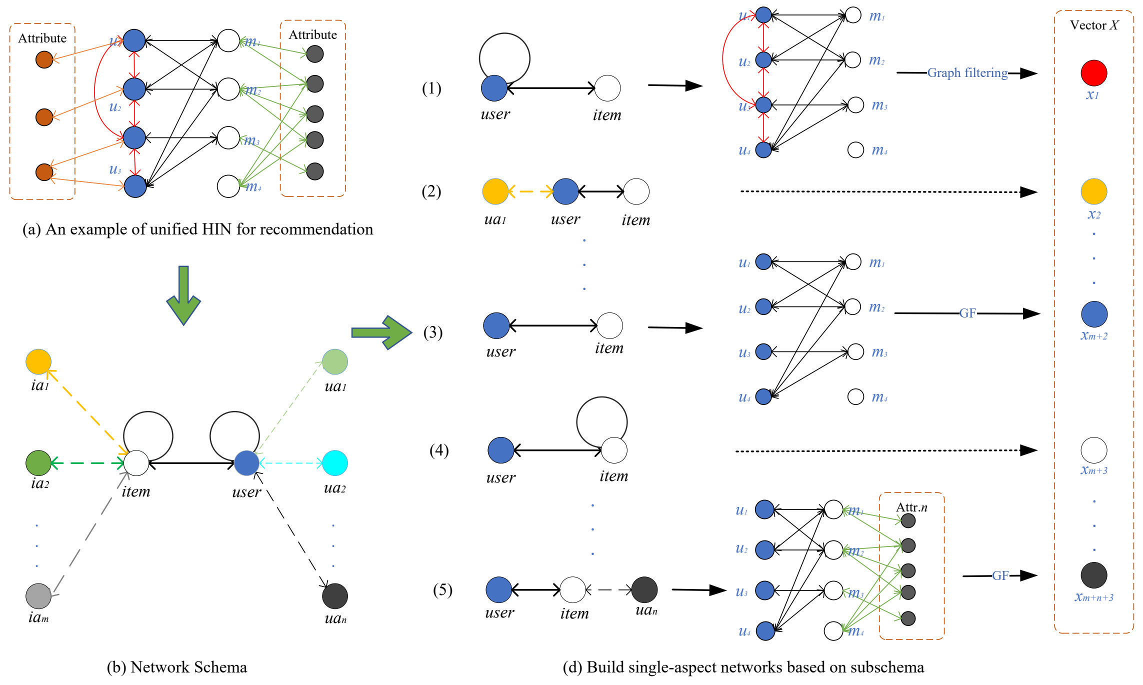

Attribute
Attribute
(a) An example of unified HIN for recommendation
(b) Network Schema
(d) Build single-aspect networks based on subschema
Vector X
Graph filtering
GF
user
item
Attr.n

Figure 2: Build single-aspect networks based on sub-schema and calculate X for DNN model.

4. Neural Graph Filtering for Recommendation

The unified HIN models various data, which have different contributions for recommendation. In this section, we propose a novel approach to weight the contributions of different typed data in HIN, called *neural graph filtering* (NGF). Formally, the NGF algorithm for CAR is as follow:

1. *Given $G_U(V, E)$, split it into multiple independent single-aspect networks according to the type of information, denoted as $\{G_x\}$;*
2. *For every rating pair $r(u, i)$, compute $\hat{r}_{ri}$ on each G_x via graph filtering, denoted as $\hat{r}_x(u, i)$;*
3. *Build aspect-level vector for $r(u, i)$, denoted as X_{ui}, $X_{ui} = \{\hat{r}_1(u, i), \cdots, \hat{r}_x(u, i), \cdots\}$;*
4. *Taking the FRP as training dataset, learn a DNN model $\hat{y}_{ui} = f(X_{ui})$;*
5. *Calculate $\hat{r}$ for predicting rating pair, $\hat{r}_{vj} = f(X_{vj})$.*

4.1. Single-aspect network

To split the unified HIN into independent single-aspect networks, we should investigate its template firstly. The meta template of HIN is a type of abstract concept graph, representing the basic structure, known as network schema.

Definition 3 (Network Schema): A meat template for $G(V,E)$ with the object mapping $\phi : V \to A$ and the edge mapping $\varphi : E \to R$, is a directed graph defined over object types A and semantic relations R, denoted as $T_G(A,R)$.

Although the HINs are various according to different recommendation scenarios, they all have the same network schema, as shown in Fig. 2 (b). That is to say the various HINs are different network instances of a same network schema. Generally, the T_G of G_U contains 5 types of semantic relations, and each semantic relation represents a kind of information that impacts the performance of recommendation. For example, the semantic relation $R(user, item)$ represents the rating information; $R(user, ua_i)$ means all the users have an attribute ua_i, and its edge instance $e(u, ua_{i,j})$ in G means u has a value $ua_{i,j}$ in attribute ua_i; $R(user, user)$ represents the social relationships and so on. In term of the relations of $T_G(A,R)$, we can get 5 types of sub-schemas, as shown in Fig. 2. Given a sub-schema, we can get the corresponding subgraph of $G_U(V,E)$, called *single-aspect network*.

Definition 3 (Single-aspect network): The single-aspect network is a subgraph of $G_U(V,E)$, whose network schema only contains rating relation and no more than 1 other semantic relation.

Except the subgraph of rating sub-schema (Fig. 2 (c)(3)), there are two semantic relations in other single-aspect networks. That is because we cannot calculate $\hat{r}$ for the predicting rating pairs without rating information. In fact, single semantic relation has no sense to recommendation until it is related to the rating relation. On the one hand, graph filtering can fully exploit the HIN with weight-aware GSR; on the other hand, the extracted information must be related to recommendation since itself is the prediction.

Except the rating relation, other semantic relations are independent. Given a single-aspect network, the prediction indicates the impact of a single information to recommendation. Finally, for any rating pair $r(u,i)$ on $G_U(V,E)$, we can calculate its predictions on different single-aspect networks by Eq. 5-7. All the predictions compose a vector X_{ui}, $X_{ui} = \{\hat{r}_1(u,i), \cdots, \hat{r}_x(u,i), \cdots\}$.

4.2. Recommendation Based on Neural Graph Filtering

In this section, we present the neural graph filtering model that utilize DNN to fuse the different predictions by learning aspect-level weights. NGF is a general machine learner working with any real valued feature vector. Given a vector $X \in \mathbf{R}^n$ as input, NGF estimates the target as:

$$\hat{y}(X) = w_0 + \sum_{i=1}^{n} w_i x_i + g(X) \tag{8}$$

where the first and second terms are the linear regression part, which models global bias of data and weight of X. The third term $g(X)$ is the core component of NGF for modelling feature interactions, which is a multi-layered feed-forward neural network.

We model the CAR as a regression problem which predicts the rating scores from users to items in the future. In the following, we elaborate the design of $g(x)$ layer by layer.

Input Layer. Given a rating pair $r(u,i)$ on $G_U(V,E)$, NGF takes the prediction vector X_{ui} as input. Formally, $H_0 = X$.

Hidden Layer. Above the input layer is a stack of fully connected layers, which are capable of learning higher-order interactions between aspect-level features, i.e. predictions.

Formally, the definition of fully connected hidden layers is as follows:

$$\begin{aligned}
&H_l = h_1(W_1^T \cdot H_0 + b_1) \\
&H_2 = h_2(W_2^T \cdot H_1 + b_2) \\
&\cdots \\
&H_L = h_L(W_L^T \cdot H_{L-1} + b_L)
\end{aligned} \tag{9}$$

where L denotes the number of hidden layers, W_l, b_l and h_l denote the weight matrix, bias vector and activation function for the l-th layer, respectively. Since the CAR task is modeled as a regression problem, NGF uses Rectifier (ReLU) as the activation function of the hidden layers, i.e., $h_l(x) = max\{0, x\}$.

Prediction Layer. At last, the output vector of the last hidden layer H_L is transformed to the final prediction score:

$$g(X) = W_O^T H_L \tag{10}$$

where vector W_O denotes the neuron weights of the prediction layer.

To summarize, we give the formulation NGF's predictive model as:

$$\hat{y}(X) = w_0 + \sum_{i=1}^{n} w_i x_i + W_O^T h_L(W_L^T(\cdots h_1(W_1^T H_0 + b_1)\cdots) + b_L) \tag{11}$$

with all model parameters $\Theta = \{w_0, \{w_i\}, W_O, \{W_l, b_l\}\}$.

Learning. To estimate model parameters of NGF, we need to specify an objective function to optimize. The CAR task, as a regression problem, commonly uses the squared loss as the objective function:

$$L_{recm} = \sum_{X_{ui} \in D} (\hat{y}(X_{ui}) - r_{ui})^2 \tag{12}$$

where D denotes the set of predictive vectors calculated for feedback rating pairs on G_U, and r_{ui} is the feedback rating score from user u to item i. The regularization terms are optional and omitted here, since some techniques in neural network modelling such as dropout can well prevent NGF from overfitting.

At last, we use stochastic gradient descent (SGD) to optimize the objective function of Eq. 11. SGD is a universal solver for optimizing DNN models, which iteratively updates the parameters until convergence. In each time, it randomly selects a training instance X, updating each model parameters towards the direction of its negative gradient:

$$\theta = \theta - \eta \cdot 2(\hat{y}(X) - r)\frac{d\hat{y}(X)}{d\theta} \tag{13}$$

where $\theta \in \Theta$ is a trainable model parameter, and $\eta > 0$ is the learning rate that controls the step size of gradient descent. As the NGF is a multi-layered neural network model, the gradient of $\hat{y}(X)$ to each model parameter can be derived using the chain rule, which has been widely implemented in ML toolkits like TensorFlow and Keras [He and Chua (2017)].

5. Experimental Studies

5.1. Evaluation Datasets

We experimented with two publicly accessible datasets: Amazon [1] and IMDb [2].

Amazon. We use book set of the real traces from Amazon dataset. The book dataset consists of 3.52M ratings from 798,431 users for 482,879 books. And the attributes include 231,452 authors, 51,812 publisher, and 156 time-nodes. The ratings vary from 1 to 5 with an increment of 1. The sparsity is about 99.47%

IMDb. This dataset is about movies, collected from the famous imdb.com. In this website, detailed descriptions are given for each movie, gross (integer, in millions), rating, reviewer number (integer, in thousands) and length. In detail, the IMDb network consists of 1482 movies, 112 genres, 529 directors, 4795 actors and 239 time-nodes. The rating range is in $(0, 10)$. There are 25,154 links and the sparsity is about 95.16%.

We use the widely used mean absolute error (MAE) and root mean square error (RMSE) to measure the quality of recommendation of different models. The metrics MAE and RMSE are defined as follows:

$$MAE = \frac{1}{|D_{test}|} \sum_{r(i,j) \in D_{test}} |r_{ij} - \hat{r}_{ij}| \tag{14}$$

$$RMSE = \sqrt{\frac{1}{|D_{test}|} \sum_{r(i,j) \in D_{test}} (r_{ij} - \hat{r}_{ij})^2} \tag{15}$$

where D_{test} denotes the test set of rating records.

5.2. Methods to Compare

We consider the following methods to compare:

- ItemKNN. It is a classical memory-based approach that computes the cosine item similarity to provide recommendation .
- BPR [Koren et al. (2009)]. The Bayesian Personalized Ranking approach optimizes the MF model with a pairwise ranking loss.
- NFM [He and Chua (2017)]. It combines the linearity of FM and the non-linearity of neural network for modeling higher-order feature interactions.
- HERec [Shi et al. (2019)]. It exploits auxiliary information based on graph embedding and then utilizes non-linear fusion function to integrate the embedded information into MF model.
- NeuACF [Han et al. (2018)]. It is a neural network based aspect-level CF which exploits the HINs via meta paths.
- MetaHIN [Lu et al. (2020)]. It exploits meta-learning on HINs for cold-start recommendation via multifaceted semantic contexts and a co-adaption meta-learner.

1. http://jmcauley.ucsd.edu/data/amazon/
2. https://www.imdb.com/

- GF. Graph filtering is our basic solution for HIN-based recommendation.
- NGF. Neural graph filtering is our enhancement algorithm, which fuses the graph filtering predictions via DNN model.

In above baselines, HERec, NeuACF and MetaHIN use the same meta paths for similarities or embedding in our experiments. The embedding dimension number $d = 64$, and other baseline parameters adopt the original optimal setting.

5.3. Experimental Results and Analysis

5.3.1. Effectiveness Experiments

For each dataset, we split the entire rating information into a training set and a test set, with the training ratios as in {80%, 60%, 40%,20%}. Obviously, smaller ratio means much sparser data for recommendation, and the context data will play more important role in handling the data sparsity problem. The result is shown in Table 1.

Table 1: Results of effectiveness experiments on two datasets.

Dataset	Training	Metrics	ItemKNN	BPR	NFM	HERec	NeuACF	MetaHIN	GF	NGF
Amazon	20%	MAE	1.2354	1.0125	0.9827	0.9581	0.9198	0.8904	0.9815	**0.8635**
		RMSE	1.3895	1.1418	1.1283	1.0957	1.0513	1.0418	0.1226	**1.0163**
	40%	MAE	1.1018	0.9585	0.8862	0.8327	0.8195	0.7862	0.8874	**0.7635**
		RMSE	1.2351	1.0129	0.9873	0.9652	0.9421	0.9346	0.9862	**0.9063**
	60%	MAE	1.0275	0.8324	0.7854	0.7547	0.7684	0.7521	0.7938	**0.7328**
		RMSE	1.1849	0.9705	0.9246	0.9079	0.8812	0.8775	0.9327	**0.8502**
	80%	MAE	0.9818	0.8071	0.7362	0.7027	0.6827	0.6954	0.7932	**0.6804**
		RMSE	1.1451	0.9429	0.9035	0.8652	0.8432	0.8526	0.9221	**0.8226**
IMDb	20%	MAE	1.1275	0.9811	0.9327	0.9114	0.8553	0.8226	0.9124	**0.8028**
		RMSE	1.3742	1.1226	1.0972	1.0321	1.0167	0.9862	1.0481	**0.9601**
	40%	MAE	1.0197	0.8245	0.8093	0.7672	0.7528	0.7465	0.7945	**0.7426**
		RMSE	1.1685	0.9813	0.9558	0.9587	0.9329	0.9318	0.9482	**0.9105**
	60%	MAE	0.9632	0.7985	0.7216	0.7042	0.7012	0.7149	0.7949	**0.6512**
		RMSE	1.1104	0.9331	0.9061	0.8618	0.8542	0.8605	0.9273	**0.8169**
	80%	MAE	0.9297	0.7624	0.6593	0.6427	0.6421	0.6465	0.7049	**0.6082**
		RMSE	1.0895	0.9018	0.8755	0.8242	0.8172	0.8268	0.9171	**0.7885**

The major findings are summarized as follows:

1. All these models are sensitive to data sparsity. With the increases of training ratio, the metric results improve significantly. Among all these baselines, the methods that exploit context data perform better than pure CF, i.e., ItemKNN and BPR. It indicates that exploiting context data is a promising way to handle the data sparsity problem in recommendation.

2. The proposed NGF method is consistently better than the baselines. Compared with other CAR methods, NGF combines graph filtering and DNN model, to fully exploit HINs and make personalized prediction. Graph filtering utilizes weight-aware GSR to compute the global similarities of rating pairs and predicts rating scores case-by-case on the single-aspect networks. DNN model is just to fuse the predictions in aspect level.

3. NGF performs better than graph filtering (GF), which indicates that the various data of HIN have different contributions for recommendation. It is meaningful to exploit the weights of themselves and their high-order interactions.

5.3.2. Individuality Loss Experiments

In this section, we consider studying the performance w.r.t. special rating pairs, i.e., individual behaviors different from public. Generally, since most of users like popular items, a user chooses a niche item can be considered as an "individual" behavior. Thus, the special rating pairs can be found through the degrees of items on G_U. We first categorize special items into three groups according to the numbers of their rating records, i.e., $(0, 5]$, $(5, 10]$ and $(10, 20]$. It is easy to see that the case in the first group is the most difficult since the items from this group have fewest rating records and their predicted scores can be easily impacted by the public. Here, we only select the baselines that have best performance in above experiments, including HERec, NeuACF, MetaHIN and NGF. We present the performance comparison of different methods in Fig. 3. Overall, NGF performs the best in all cases. We also find that the performances of NGF and MetaHIN have different improvements. While the performances of other baselines are below their averages in Table 1. That indicates less individuality information lost in NGF.

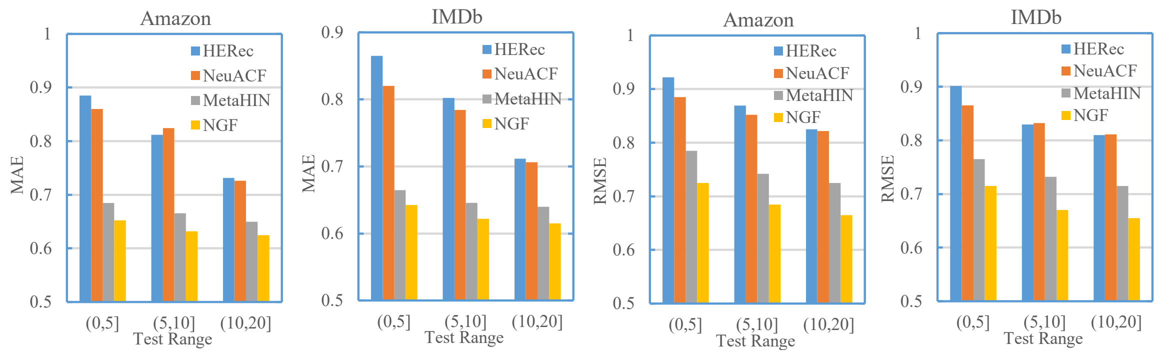


Figure 3: Experiments on individuality loss.

5.3.3. Impact of Time and Rating Weights

To analyze the impact of temporal and rating information, we check the performance change w.r.t. the corresponding decay parameters, i.e., α and β. We vary it from 0 to 1 with a step of 0.1, and the tuning results are shown in Fig. 4. Overall, the change trend is not smooth, indicating that NGF is sensitive to time and rating data on HINs. We also find that rating information is more important than time since RMSE turns to bigger when $\beta = 0$. At last, we suggest α, β are set in range [0.75, 0.85] for better performance.

6. Related Work

Since CF methods usually suffer from data sparsity and cold-start problems, some works attempt to leverage side information, such as social data , location information and content information. The side information are also known as context data and these methods are called context-aware recommendation (CAR) [Xin et al. (2019)]. Nowadays, there are many

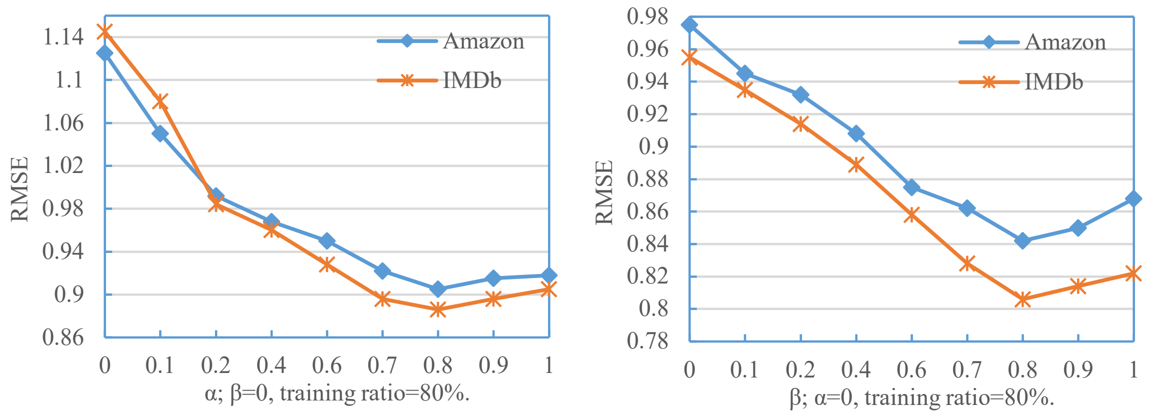


Figure 4: Impact of the time and rating information on HINs.

kinds of learning models to exploit context data for recommendation, likely FM, MF, feature mapping, DNN and meta-learning.

FMs are a general predictor working with any real valued feature vectors, which estimates the target by modelling all interactions between each pair of features via factorized interaction parameters [He and Chua (2017), Liang et al. (2020)]. To improve the effectiveness by exploiting context data, many works rewrite the objective function of MF via adding similarity terms [Shi et al. (2019)]. With the surge of deep learning, deep neural networks are also employed to deeply capture the latent features of users and items for recommendation, likely NeuACF, CFM [Han et al. (2018)]. The recent episodic meta-learning paradigm has offered insights into model new users or items with context data. It focuses on deriving prior knowledge across different leaning task, so as to rapidly adapt to a new task [Lu et al. (2020)]. These learning methods all belong to *global learning models* that suffer individuality loss problem.

7. Conclusion

In this paper, we challenge the problems in HIN-based recommendation, i.e., information extraction and individuality loss and propose a neural graph filtering solution for CAR. We first split the HIN into many independent single-aspect networks according to the semantic relations. Then, we use graph filtering to calculate the predictions on subgraphs. The following DNN is to fuse the personalized predictions in aspect-level. Extensive experiments on real-world datasets demonstrate the effectiveness of NGF.

References

Xiaotian Han, Chuan Shi, Senzhang Wang, Philip S. Yu, and Li Song. Aspect-level deep collaborative filtering via heterogeneous information networks. In Jérôme Lang, editor, *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, pages 3393–3399. ijcai.org, 2018. doi: 10.24963/ijcai.2018/471. URL https://doi.org/10.24963/ijcai.2018/471.

Xiangnan He and Tat-Seng Chua. Neural factorization machines for sparse predictive analytics. In Noriko Kando, Tetsuya Sakai, Hideo Joho, Hang Li, Arjen P. de Vries, and Ryen W. White, editors, *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, Shinjuku, Tokyo, Japan, August 7-11, 2017*, pages 355–364. ACM, 2017. doi: 10.1145/3077136.3080777. URL https://doi.org/10.1145/3077136.3080777.

Liang Hu, Songlei Jian, Longbing Cao, Zhiping Gu, Qingkui Chen, and Artak Amirbekyan. HERS: modeling influential contexts with heterogeneous relations for sparse and cold-start recommendation. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 3830–3837. AAAI Press, 2019. doi: 10.1609/aaai.v33i01.33013830. URL https://doi.org/10.1609/aaai.v33i01.33013830.

Glen Jeh and Jennifer Widom. Simrank: a measure of structural-context similarity. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, July 23-26, 2002, Edmonton, Alberta, Canada*, pages 538–543. ACM, 2002. doi: 10.1145/775047.775126. URL https://doi.org/10.1145/775047.775126.

Yehuda Koren, Robert M. Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009. doi: 10.1109/MC.2009.263. URL https://doi.org/10.1109/MC.2009.263.

Jingjing Li, Ke Lu, Zi Huang, and Heng Tao Shen. On both cold-start and long-tail recommendation with social data. *IEEE Trans. Knowl. Data Eng.*, 33(1):194–208, 2021. doi: 10.1109/TKDE.2019.2924656. URL https://doi.org/10.1109/TKDE.2019.2924656.

Junjie Liang, Dongkuan Xu, Yiwei Sun, and Vasant G. Honavar. LMLFM: longitudinal multi-level factorization machine. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 4811–4818. AAAI Press, 2020. URL https://aaai.org/ojs/index.php/AAAI/article/view/5916.

Yuanfu Lu, Yuan Fang, and Chuan Shi. Meta-learning on heterogeneous information networks for cold-start recommendation. In Rajesh Gupta, Yan Liu, Jiliang Tang, and B. Aditya Prakash, editors, *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, pages 1563–1573. ACM, 2020. doi: 10.1145/3394486.3403207. URL https://doi.org/10.1145/3394486.3403207.

Chen Luo, Wei Pang, Zhe Wang, and Chenghua Lin. Hete-cf: Social-based collaborative filtering recommendation using heterogeneous relations. In Ravi Kumar, Hannu Toivonen, Jian Pei, Joshua Zhexue Huang, and Xindong Wu, editors, *2014 IEEE International Conference on Data Mining, ICDM 2014, Shenzhen, China, December 14-17, 2014*, pages

917–922. IEEE Computer Society, 2014. doi: 10.1109/ICDM.2014.64. URL https://doi.org/10.1109/ICDM.2014.64.

Paridhi Maheshwari, Ritwick Chaudhry, and Vishwa Vinay. Scene graph embeddings using relative similarity supervision. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 2328–2336. AAAI Press, 2021. URL https://ojs.aaai.org/index.php/AAAI/article/view/16333.

Chuan Shi, Binbin Hu, Wayne Xin Zhao, and Philip S. Yu. Heterogeneous information network embedding for recommendation. *IEEE Trans. Knowl. Data Eng.*, 31(2):357–370, 2019. doi: 10.1109/TKDE.2018.2833443. URL https://doi.org/10.1109/TKDE.2018.2833443.

Yizhou Sun, Jiawei Han, Xifeng Yan, Philip S. Yu, and Tianyi Wu. Pathsim: Meta path-based top-k similarity search in heterogeneous information networks. *Proc. VLDB Endow.*, 4(11):992–1003, 2011. URL http://www.vldb.org/pvldb/vol4/p992-sun.pdf.

Xin Xin, Bo Chen, Xiangnan He, Dong Wang, Yue Ding, and Joemon Jose. CFM: convolutional factorization machines for context-aware recommendation. In Sarit Kraus, editor, *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 3926–3932. ijcai.org, 2019. doi: 10.24963/ijcai.2019/545. URL https://doi.org/10.24963/ijcai.2019/545.

Chuanyan Zhang, Xiaoguang Hong, and Zhaohui Peng. Gsimrank: A general similarity measure on heterogeneous information network. In Xin Wang, Rui Zhang, Young-Koo Lee, Le Sun, and Yang-Sae Moon, editors, *Web and Big Data - 4th International Joint Conference, APWeb-WAIM 2020, Tianjin, China, September 18-20, 2020, Proceedings, Part I*, volume 12317 of *Lecture Notes in Computer Science*, pages 588–602. Springer, 2020. doi: 10.1007/978-3-030-60259-8_43. URL https://doi.org/10.1007/978-3-030-60259-8_43.