Chapter 5
Recommendation with Heterogeneous Information

Abstract Recently, heterogeneous information network (HIN) analysis has attracted a lot of attention, and many data mining tasks have been exploited on HIN. As an important data mining task, recommender system includes a lot of object types (e.g., users, movies, actors, and interest groups in movie recommendation) and the rich relations among object types, which naturally constitute an HIN. The comprehensive information integration and rich semantic information of HIN make it promising to generate better recommendation. In this chapter, we introduce three works on recommendation with HIN. One work recommends items with semantic meta paths, and the other two works extend traditional matrix factorization with rich heterogeneous information.

5.1 Recommendation Based on Semantic Path

5.1.1 Overview

In recent years, some works [5, 9, 24] have taken notice of the benefits of HIN for recommendation, where the objects and their relations in recommended system constitute a heterogeneous information network (HIN). Figure 5.1 shows such an example. The HIN not only contains different types of objects in movie recommendation (e.g., users and movies) but also illustrates all kinds of relations among objects, such as viewing information, social relations, and attribute information. Constructing heterogeneous networks for recommendation can effectively integrate all kinds of informations, which can be potentially utilized for recommendation. Moreover, the objects and relations in the networks have different semantics, which can be explored to reveal subtle relations among objects. For example, the meta path "User-Movie-User" in Fig. 5.1 means users viewing the same movies and can be used to find the similar users according to viewing records. If we recommend movies following this meta path, it will recommend the movies that are seen by users having the same viewing records with the given user. It corresponds to the collaborative filtering model in essence. Similarly, the "User-Interest Group-User" path can find the similar users with similar interests. This path corresponds to the member recommendation [25].

C. Shi and P.S. Yu, *Heterogeneous Information Network Analysis and Applications*, Data Analytics, DOI 10.1007/978-3-319-56212-4_5

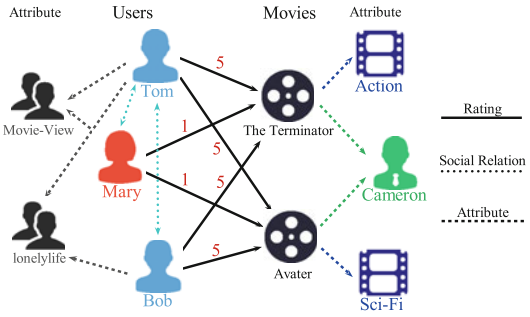


Fig. 5.1 The objects and relations in movie recommended system are organized as a weighted heterogeneous information network

So we can directly recommend items based on the similar users generated by different meta paths connecting users. Moreover, it can realize different recommendation models through properly setting meta paths. However, this idea faces the following two challenges.

Firstly, conventional HIN and meta path cannot be directly applied to recommended system. As we know, conventional HIN and meta path do not consider the attribute values on links. However, this movie recommendation network can contain attribute values on links. Concretely, in recommended system, the users can provide a rating score to each movie viewed. The rating scores usually range from 1 to 5 as indicated on the link between user and movie in Fig. 5.1, where higher score means stronger preference. Ignoring the rating scores may result in bad similarity discovery on users. For example, according to the path "User-Movie-User," Tom has the same similarity with Mary and Bob, since they view the same movies. However, they may have totally different tastes due to different rating scores. In fact, Tom and Bob should be more similar, since they both like the same movies very much with high scores. Mary may have totally different tastes, because she does not like these movies at all. The conventional meta path does not allow links to have attribute values (e.g., rating scores in the above example) [19, 24], and hence, it cannot reveal this subtle difference. However, this difference is very important, especially in recommended system, to more accurately reveal relations of objects. So we need to extend existing HIN and meta path for considering attribute values on links. Moreover, the new similarity measures are urgently needed for development.

Secondly, it is difficult to effectively combine information from multiple meta paths for recommendation. As we have said, different types of similar users will be generated through different meta paths, and these different types of similar users will recommend different items. A weight learning method can be designed to combine these recommendations, and each path can be assigned with a learned weight preference. A good weight learning method should obtain prioritized and personalized weights. That is, the learned weights can represent the importance of paths, and each user should have personalized weights to embody his preferences on paths. The prioritized and personalized weights are very important for recommendation, since they can deeply reveal the characteristics of users. Much more than this, it makes the recommendation more explainable, since meta paths contain semantics.

For example, if a user has high-weight preference on the "User-Interest Group-User" path, we can explain that the recommendation results stem from movies viewed by users in the interest groups he joined in. Unfortunately, the personalized weights may suffer from the rating sparsity problem, especially for users with little rating information. The reasons lie in that so many parameters are needed to be learned and rating information is usually not sufficient.

In this chapter, we extend HIN and meta path for widely existing attribute values on links in information networks and, firstly, propose the weighted HIN and weighted meta path concepts to more subtly reveal object relations through distinguishing link attribute values. Instead of designing an ad hoc similarity measure for weighted meta paths, we design a novel similarity computation strategy that can make existing path-based similarity measures still usable. Furthermore, the semantic path-based personalized recommendation method SemRec is proposed to flexibly integrate heterogeneous information through setting meta paths. In SemRec, we design a novel weight regularization term to obtain personalized weight preferences on paths and alleviate the rating sparsity through employing the consistency rule of weight preferences of similar users.

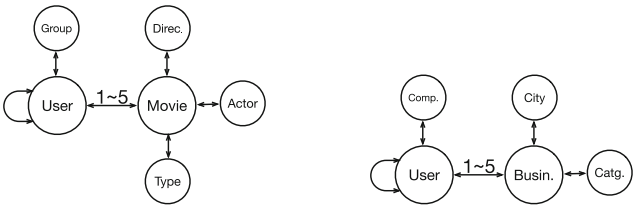
5.1.2 Heterogeneous Network Framework for Recommendation

In this section, we describe notations used in this chapter and present some preliminary knowledge.

5.1.2.1 Basic Concepts

An HIN is a special type of information network with the underneath data structure as a directed graph, which contains either multiple types of objects or multiple types of links. Traditionally, HIN does not consider the attribute values on links. However, many real networks contain attribute values on links. For example, users usually rate movies with a score from 1 to 5 in movie recommended system, and the "author of" relations between authors and papers in bibliographic networks can take values (e.g., 1, 2, 3) which means the order of authors in the paper. In this chapter, we formally propose the weighted heterogeneous information network concept to handle this condition.

Definition 5.1 (*Weighted information network*) Given a schema $S = (\mathtt{A}, \mathtt{R}, \mathtt{W})$ which consists of a set of object types $\mathtt{A} = \{A\}$, a set of relations connecting object pairs $\mathtt{R} = \{R\}$, and a set of attribute values on relations $\mathtt{W} = \{W\}$, a **weighted information network** is defined as a directed graph $G = (V, E, W)$ with an object type mapping function $\varphi : V \to \mathtt{A}$, a link type mapping function $\psi : E \to \mathtt{R}$, and an attribute value type mapping function $\theta : W \to \mathtt{W}$. Each object $v \in V$ belongs to



(a) Douban (b) Yelp

Fig. 5.2 Network schema of weighted heterogeneous information networks constituted by two datasets

one particular object type $\varphi(v) \in \mathtt{A}$, each link $e \in E$ belongs to a particular relation $\psi(e) \in \mathtt{R}$, and each attribute value $w \in W$ belongs to a particular attribute value type $\theta(w) \in \mathtt{W}$. When the types of objects $|\mathtt{A}| = 1$ and the types of relations $|\mathtt{R}| = 1$, it is a **homogeneous information network**. When the types of objects $|\mathtt{A}| > 1$ (or the types of relations $|\mathtt{R}| > 1$) and the types of attribute values $|\mathtt{W}| = 0$, the network is called **unweighted heterogeneous information network**. When the types of objects $|\mathtt{A}| > 1$ (or the types of relations $|\mathtt{R}| > 1$) and the types of attribute values $|\mathtt{W}| > 0$, the network is called **weighted heterogeneous information network** (WHIN).

Conventional HIN is an unweighted HIN, where there are no attribute values on relations or we do not consider them. For a WHIN, there are attribute values on some relation types, and these attribute values may be discrete or continuous values.

Example 5.1 A movie recommended system can be organized as a weighted heterogeneous information network, whose network schema is shown in Fig. 5.2a. The network contains objects from six types of entities (e.g., users, movies, groups, actors) and relations between them. Links between objects represent different relations. For example, links exist between users and users denoting the friendship relations, between users and movies denoting rating and rated relations. In addition, the network also contains one type of attribute value on the rating relation between users and movies, which take values from 1 to 5.

Two objects in an HIN can be connected via different paths, and these paths have different meanings. As an example shown in Fig. 5.2a, users can be connected via "User-User" (UU) path, "User-Group-User" (UGU) path, "User-Movie-User" (UMU), and so on. These paths are called meta paths that are the combination of a sequence of relations between object types. Although meta path is widely used to reveal semantics among objects [20], it fails to distinguish the attribute values between two objects in WHIN. For example, if ignoring the different rating scores of users on items in above movie recommendation, we may obtain incorrect results. Consider a scenario that we use the UMU path to find the similar users of Tom according to their viewing records in Fig. 5.1. We can infer that Tom is very similar to Mary and Bob, since they have the same viewing records. However, it is obvious that Tom and Mary have totally different tastes. So the UMU path cannot subtly

reveal the different ratings of users on the same movies. In order to effectively exploit semantics in WHIN, we extend the conventional meta path to consider attribute values on relations. Without loss of generality, we assume the attribute values on relations in WHIN are discrete. For continuous attribute values on relations, we can convert the continuous attribute values into discrete ones.

Definition 5.2 (*Extended meta path on WHIN*) Extended meta path is a meta path based on a certain attribute value constraint on relations, which is denoted as $A_1 \xrightarrow{\delta_1(R_1)} A_2 \xrightarrow{\delta_2(R_2)} \xrightarrow{\delta_l(R_l)} A_{l+1}|\mathtt{C}$ (also denoted as $A_1(\delta_1(R_1))A_2(\delta_2(R_2)) \cdots\cdots (\delta_l(R_l))A_{l+1}|\mathtt{C}$). If the relation R has attribute values on links, the attribute value function $\delta(R)$ is a set of values from the attribute value range of relation R, else $\delta(R)$ is an empty set. $A_i \xrightarrow{\delta_i(R_i)} A_{i+1}$ represents the relation R_i between A_i and A_{i+1} based on the attribute values $\delta_i(R_i)$. The constraint $\mathtt{C}$ on attribute value functions is a set of correlation constraints among attribute value functions. If all attribute value functions in a meta path are empty set (the corresponding constraint $\mathtt{C}$ is also an empty set), the path is called an **unweighted meta path**, else the path is called a **weighted meta path**.

Note that the conventional meta path is an unweighted meta path that can be considered as the special case of a weighted meta path.

Example 5.2 Taking Fig. 5.2a as an example, the rating relation between users U and movies M can take scores from 1 to 5. The weighted meta path $U \xrightarrow{1} M$ (i.e., $U(1)M$) means movies rated by users with score 1, which implies that users dislike the movies. The weighted meta path $U \xrightarrow{1,2} M \xrightarrow{1,2} U$ (i.e., $U(1,2)M(1,2)U$) means users disliking the same movies as the target user, while the unweighted meta path UMU can only reflect that users have the same viewing records. Furthermore, we can flexibly set the correlation constraints of attribute value functions on different relations in weighted meta paths. For example, the path $U(i)M(j)U|i = j$ means users having exactly the same ratings on some movies as the target user. Under this path, we can easily find that, in Fig. 5.1, Tom is very similar to Bob, while they are totally dissimilar to Mary.

5.1.2.2 Recommendation on Heterogeneous Networks

For a target user, recommended systems usually recommend items according to his similar users. In HIN, there are a number of meta paths connecting users, such as "User-User" and "User-Movie-User". Based on these paths, users have different similarities. Here, we define the path-based similarity as follows.

Definition 5.3 (*Path-based similarity*) In HIN, the path-based similarity of two objects is the similarity evaluation based on the given meta path connecting these two objects.

Table 5.1 The meanings and corresponding recommendation models of meta paths

No.	Meta path	Semantic meaning	Recommendation model
1	UU	Friends of the target user	Social recommendation
2	UGU	Users in the same group of the target user	Member recommendation
3	UMU	Users who view the same movies with the target user	Collaborative recommendation
4	UMTMU	Users who view the movies having the same types with that of the target user	Content recommendation

After obtaining the path-based similarity of users, we can recommend items according to the similar users of the target user. More importantly, the meta paths connecting users have different semantics, which can represent different recommendation models. As an example shown in Fig. 5.2a, "User-User" (UU) means friends of the target user. If we recommend movies according to the similarity of users generated by that path, it will recommend the movies viewed by friends of the target user. Indeed, it is the social recommendation. Another example is that "User-Movie-User" (UMU) means users who view the same movies with the target user. Following that path, it will recommend the movies viewed by users having the similar viewing records with the target user. It is collaborative recommendation in essential. Table 5.1 shows the other representative paths and the corresponding recommendation models. Based on the HIN framework, we can flexibly represent different recommendation models through properly setting meta paths.

5.1.2.3 Similarity Measure Based on Weighted Meta Path

Similarity measure on meta paths have been well studied, and several path-based similarity measures have been proposed on HIN, such as PathSim [19], PCRW [6], and HeteSim [16]. However, these similarity measures cannot be directly applied to weighted meta path, because they do not consider the attribute value constraint on relations. As we know, the essential of the path-based similarity measure is to evaluate the proportion of the number of paths connecting two objects on all possible paths along the meta path [19], so the paths along a weighted meta path must satisfy the attribute value constraint. Moreover, the attribute value on relations may be a variable, even correlated. Taking the $U(i)M(j)U|i = j$ path as an example, the attribute values i and j are variables from 1 to 5, and they satisfy constraint $i = j$. For this kind of paths, existing path-based similarity measures cannot handle it.

In order to address the variable, even correlated, attribute value constraints in a weighted meta path, we extend the meta path concept and propose a general strategy to make existing path-based similarity measure still usable, instead of proposing an ad hoc similarity measure. Specifically, we can decompose the weighted meta path into a group of atomic meta paths with fixed attribute value constraint. For an atomic meta path, the existing path-based similarity measures can be used directly.

Definition 5.4 (*Atomic meta path*) If all attribute value functions $\delta(R)$ in a weighted meta path take a specific value, the path is called an **atomic meta path**. A weighted meta path is **a group of atomic meta paths** which contain all atomic meta paths that satisfy the constraint C.

Example 5.3 Taking Fig. 5.2a as an example, $U(1)M(1)U$ and $U(1)M(2)U$ both are atomic meta paths. The weighted meta path $U(i)M(j)U|i = j$ is a group of five atomic meta paths (e.g., $U(1)M(1)U$ and $U(2)M(2)U$).

Since a weighted meta path is a group of corresponding atomic meta paths, the similarity measure based on a weighted meta path can be considered as the sum of the similarity measure based on the corresponding atomic meta paths. So the similarity measure based on a weighted meta path can be evaluated based on the following two steps: (1) Evaluate the similarity based on each atomic meta path with existing path-based measures; (2) sum up the similarities on all atomic meta paths in the weighted meta path. Note that the similarity measure needs to consider the effect of the normalized term existing in some path-based similarity measures, such as PathSim [19] and HeteSim [16]. Taking PathSim as an example, we illustrate its calculation process along conventional and weighted meta path in Fig. 5.3, where the rating matrix between 3 users and 2 movies is from Fig. 5.1. We know that PathSim counts the number of path instances connecting two objects along conventional meta path with a normalized term (shown in the upper half of Fig. 5.3), and thus, it regards that the users all are the same. As shown in the lower half of Fig. 5.3, PathSim along weighted meta path firstly counts the number of path instances along each atomic meta path and then sums up the number of path instances along all atomic meta paths before normalization. And thus, it can more accurately discover that only u_1 and u_3 are similar, since they have the same tastes in movies.

5.1.3 The SemRec Solution

In this section, we proposed a **Sem**antic path-based personalized **Rec**ommendation method (**SemRec**) to predict the scores of items. Specifically, SemRec first evaluates the similarity of users based on weighted or unweighted meta paths and then infers the predicted scores on items according to the rating scores of similar users. Under different meta paths, the users can obtain different recommendation results. How to effectively combine these recommendations generated by different meta paths is challenging. We need to put different preferences on the various meta paths. This

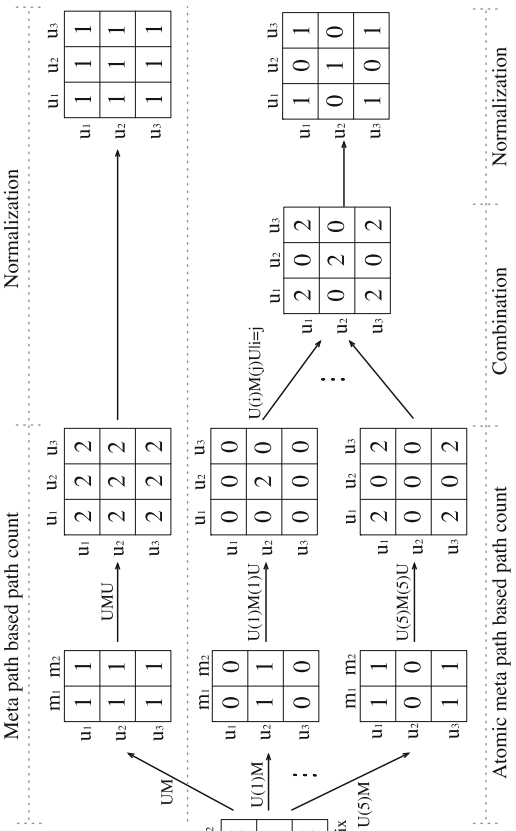


Fig. 5.3 PathSim similarity measure based on conventional and weighted meta path

results in assigning preference weight to each meta path. We abbreviate the preference weight as weight when the context is clear without confusion with the link weight in the weighted meta path. There are two aspects of difficulties in learning the weights. (1) Prioritized weights: That is, the weights learned should embody the importance of paths and reflect users' preferences. However, the similarity evaluations based on different paths have significant bias, which makes path preference hard to reflect the path importances. For example, the similarity evaluations may all be high based on a path with dense relations, while the similarity evaluations may all be low based on another path with sparse relations. So the similarity evaluations based on different paths cannot reflect the similarity of two objects. SemRec designs a normalized rating intensity operation to eliminate the similarity bias, which makes the weight better reflect path importances. (2) Personalized weights: That is, it is better to learn weight preferences for each user. However, personalized weight learning may suffer from the rating sparsity problem, since many users have little rating informations. In order to alleviate the rating sparsity problem for personalized weight learning, we propose *the consistency rule of weight preferences of similar users*. That is, we assume that two similar users have consistent weight preferences on meta paths. While it is reasonable, it is seldom used before. Two users are similar based on a path, which implies the path has similar impacts on these two users. That is to say, these users have the consistent preferences on the path. Following this principle, we design a novel weight regularization term, which effectively alleviates rating sparsity in personalized weight learning.

In the following sections, we firstly design the basic recommendation method based on a single path. And then, we propose three levels of personalized recommendation methods based on multiple paths: unified weights for all users, personalized weights for each user, and personalized weights with weight regularization.

5.1.3.1 Recommendation with Single Path

Based on the path-based similarity of users, we can find the similar users of a target user under a given path, and then, the rating score of the target user on an item can be inferred according to the rating scores of his similar users on the item. Assume that the range of rating scores are from 1 to N (e.g., 5); P is a set of unweighted or weighted meta paths; $R \in \mathbf{R}^{|U|\times|I|}$ is the rating matrix, where $R_{u,i}$ denotes the rating score of user u on item i; and $S \in \mathbf{R}^{|U|\times|U|}$ is the path-based similarity matrix of users, where $S^{(l)}_{u,v}$ is the similarity of users u and v under path P_l. Here, we define the *rating intensity* $Q \in \mathbf{R}^{|U|\times|I|\times N}$, where $Q^{(l)}_{u,i,r}$ represents the intensity of user u rating item i with score r given path P_l. $Q^{(l)}_{u,i,r}$ is determined by two aspects: the number of similar users rating the item i with score r and the similarity of users. So we calculate $Q^{(l)}_{u,i,r}$ as the sum of similarity of users rating i with r.

$$Q_{u,i,r}^{(l)} = \sum_v S_{u,v}^{(l)} \times E_{v,i,r}$$
$$E_{v,i,r} = \begin{cases} 1 & R_{v,i} = r \\ 0 & \text{others} \end{cases} \tag{5.1}$$

where $E_{v,i,r}$ indicates whether user v rates item i with score r.

Under a meta path P_l, the rating of a user u on an item i ranges from 1 to N with different rating intensities $Q_{u,i,r}^{(l)}$. So the *predicted rating score*, denoted as $\hat{R}_{u,i}^{(l)}$, of user u on item i under the path P_l can be the average of rating scores weighted by corresponding normalized intensity.

$$\hat{R}_{u,i}^{(l)} = \sum_{r=1}^{N} r \times \frac{Q_{u,i,r}^{(l)}}{\sum_{k=1}^{N} Q_{u,i,k}^{(l)}} \tag{5.2}$$

and $\hat{R}^{(l)} \in \mathbf{R}^{|U|\times|I|}$ means the predicted rating matrix under path P_l.

According to Eq. 5.2, we can predict the rating score of a user on an item under a given path and then recommend the item with the high score for a target user. Moreover, Eq. 5.2 has an additional advantage that it eliminates the similarity bias existing in different meta paths. As we know, the similarity of users under different meta paths has different scales, which makes similarity evaluation and rating intensity incomparable among different paths. The normalized rating intensity in Eq. 5.2 is able to eliminate those scale differences.

5.1.3.2 Recommendation with Multiple Paths

Under different meta paths, there are different predicted rating scores. In order to calculate the compositive score, we propose three different weight learning methods corresponding to different levels of personalized weights of users.

Unified weight learning for all users For all users, we assign each meta path with a unified weight, which means the user preference on the path. This weight vector is denoted as $\boldsymbol{w} \in \mathbf{R}^{1\times|P|}$, and $\boldsymbol{w}^{(l)}$ means the weight on path P_l. The final predicted rating score under all meta paths, denoted as $\hat{R}_{u,i}$, can be the weighted sum of predicted rating score under each meta path.

$$\hat{R}_{u,i} = \sum_{l=1}^{|P|} \boldsymbol{w}^{(l)} \times \hat{R}_{u,i}^{(l)} \tag{5.3}$$

Hopefully, the predicted rating matrix $\hat{R} \in \mathbf{R}^{|U|\times|I|}$ should be as close as to the real rating matrix R. So a direct optimization objective can be defined as the square error between the real scores and the predicted scores.

$$\begin{aligned} \min_{\boldsymbol{w}} \mathrm{L}_1(\boldsymbol{w}) &= \tfrac{1}{2}||Y \odot (R - \sum_{l=1}^{|P|} \boldsymbol{w}^{(l)} \hat{R}^{(l)})||_2^2 + \tfrac{\lambda_0}{2}||\boldsymbol{w}||_2^2 \\ s.t. \qquad & \boldsymbol{w} \geq 0 \end{aligned} \tag{5.4}$$

where the notation $\odot$ is the Hadamard product (also know as the entrywise product) between matrices, and $||\cdot||_p$ is the matrix L^p-norm. Y is an indicator matrix with $Y_{u,i} = 1$ if user u rated item i, and otherwise, $Y_{u,i} = 0$.

Personalized weight learning for individual user The above optimization objective has a basic assumption: All users have the same path preferences. However, in many real applications, each user has his personal interest preferences. Unified weights cannot provide personalized recommendations for users. To realize personalized recommendation, each user is assigned with weight vector on meta paths. The weight matrix is denoted as $W \in \mathbf{R}^{|U| \times |P|}$, in which each entry, denoted as $W_u^{(l)}$, means the preference weight of user u on path P_l. The column vector $W^{(l)} \in \mathbf{R}^{|U| \times 1}$ means the weight vector of all users on path P_l. So the predicted rating $\hat{R}_{u,i}$ of user u rating item i under all paths is as follows:

$$\hat{R}_{u,i} = \sum_{l=1}^{|P|} W_u^{(l)} \times \hat{R}_{u,i}^{(l)} \tag{5.5}$$

Similarly, we can define the optimization objective as follows:

$$\begin{aligned} \min_{W} \mathrm{L}_2(W) &= \tfrac{1}{2}||Y \odot (R - \sum_{l=1}^{|P|} diag(W^{(l)}) \hat{R}^{(l)})||_2^2 + \tfrac{\lambda_0}{2}||W||_2^2 \\ s.t. \qquad & W \geq 0 \end{aligned} \tag{5.6}$$

where $diag(W^{(l)})$ means the diagonal matrix transformed from a vector $W^{(l)}$.

Personalized weight learning with weight regularization Although Eq. 5.6 consider user's personalized weights, it may be hard to effectively learn weights for those users that have little rating information. There are $|U| \times |P|$ weight parameters to learn, while the training samples are usually much smaller than $|U| \times |I|$. The training samples are usually not sufficient for the weight learning, specially for those cold-start users and items. According to the consistency rule of weight preferences of similar users mentioned above, the path weights of a user should be consistent to that of his similar users. For users with little rating information, their path weights can be learnt from the weights of their similar users, since the similarity information of users are more available through meta paths. So we design a weight regularization term as follows, which compels the weights of a user consistent to the average of weights of his similar users.

$$\sum_{u=1}^{|U|} \sum_{l=1}^{|P|} (W_u^{(l)} - \sum_{v=1}^{|U|} \bar{S}_{u,v}^{(l)} W_v^{(l)})^2 \tag{5.7}$$

where $\bar{S}^{(l)}_{u,v} = \frac{S^{(l)}_{u,v}}{\sum_v S^{(l)}_{u,v}}$ is the normalized user similarity based on path P_l. For convenience, the weight regularization term can be written as the following matrix format:

$$\sum_{l=1}^{|P|} ||W^{(l)} - \bar{S}^{(l)} W^{(l)}||_2^2 \tag{5.8}$$

And thus, the optimization objective is defined as follows:

$$\begin{aligned} \min_W \mathrm{L}_3(W) = \quad & \tfrac{1}{2}||Y \odot (R - \sum_{l=1}^{|P|} diag(W^{(l)})\hat{R}^{(l)})||_2^2 \\ + \quad & \tfrac{\lambda_1}{2} \sum_{l=1}^{|P|} ||W^{(l)} - \bar{S}^{(l)} W^{(l)}||_2^2 + \tfrac{\lambda_0}{2}||W||_2^2 \\ s.t. \quad & W \geq 0 \end{aligned} \tag{5.9}$$

The above optimization objective is a nonnegative quadratic programming problem, a simple special case of nonnegative matrix factorization. Projected gradient method for nonnegative bound-constrained optimization [7] can be applied to solve this problem. The gradient of Eq. 5.9 with respect to $W^{(l)}_u$ can be calculated as follows:

$$\begin{aligned} \frac{\partial \mathrm{L}_3(W)}{\partial W^{(l)}_u} = \quad & -(Y_u \odot (R_u - \sum_{l=1}^{|P|} W^{(l)}_u \hat{R}^{(l)}_u))^T \hat{R}^{(l)}_u + \lambda_0 W^{(l)}_u \\ + \quad & \lambda_1 (W^{(l)}_u - \bar{S}^{(l)}_u W^{(l)}) - \lambda_1 \bar{S}^{(l)T}_u (W^{(l)} - \bar{S}^{(l)} W^{(l)}) \end{aligned} \tag{5.10}$$

$W^{(l)}_u$ can be updated as follows:

$$W^{(l)}_u = max(0, W^{(l)}_u - \alpha \frac{\partial \mathrm{L}_3(W)}{\partial W^{(l)}_u}) \tag{5.11}$$

where α is the step size and can be set according to [7]. Algorithm 1 shows the framework of this version of SemRec.

5.1.4 Experiments

In this section, extensive experiments on two real datasets illustrate the traits of SemRec. We first validate the effectiveness of SemRec, especially for cold-start problem. Then, we thoroughly explore the meanings of weights learned and validate the benefits of the proposed weighted meta path.

Algorithm 1 Framework of SemRec

Require:
G: weighted heterogeneous information network
P: meta paths connecting users
λ_0 and λ_1: controlling parameter
α: step size for updating parameters
ε: convergence tolerance
Ensure:
W: the weight matrix of all users on all paths.
1: **for** $P_l \in P$ **do**
2: Evaluate user similarity $S^{(l)}$
3: Calculate rating intensity $Q^{(l)}$ with Eq. 5.1
4: Calculate predicted rating score $\hat{R}^{(l)}$ with Eq. 5.2
5: **end for**
6: Initialize $W > 0$
7: **repeat**
8: $W_{old} := W$
9: Calculate $\frac{\partial \mathrm{L}_3(W)}{\partial W}$ with Eq. 5.10
10: $W := max(0, W - \alpha \frac{\partial \mathrm{L}_3(W)}{\partial W})$
11: **until** $|W - W_{old}| < \varepsilon$

5.1.4.1 Experiment Settings

In order to get more comprehensive heterogeneous information, we crawled a new dataset from Douban,[1] a well-known social media network in China. The dataset includes 13,367 users and 12,677 movies with 1,068,278 movie ratings ranging from 1 to 5. The dataset includes the social relation among users and the attribute information of users and movies. Another dataset is the Yelp challenge dataset.[2] This dataset contains user ratings on local business and attribute information of users and businesses. The dataset includes 16,239 users and 14,284 local businesses with 198,397 ratings from 1 to 5. The detailed description of these two datasets can be seen in Table 5.2, and their network schemas are shown in Fig. 5.2. We can find that these two datasets have different properties. The Douban dataset has dense rating relations but sparse social relations, while the Yelp dataset has sparse rating relations but dense social relations.

We use two widely used metrics, Root Mean Square Error (RMSE) and Mean Absolute Error (MAE), to measure the rating prediction quantity.

$$RMSE = \sqrt{\frac{\sum_{(u,i)\in R_{test}} (R_{u,i} - \hat{R}_{u,i})^2}{|R_{test}|}} \tag{5.12}$$

$$MAE = \frac{\sum_{(u,i)\in R_{test}} |R_{u,i} - \hat{R}_{u,i}|}{|R_{test}|} \tag{5.13}$$

[1] http://movie.douban.com/.
[2] http://www.yelp.com/dataset_challenge/.

Table 5.2 Statistics of Douban and Yelp datasets

Dataset	Relations (A-B)	Number of A	Number of B	Number (A-B)	Ave. degrees of A/B
Douban	User–Movie	13367	12677	1068278	79.9/84.3
	User–User	2440	2294	4085	1.7/1.8
	User–Group	13337	2753	570047	42.7/207.1
	Movie–Director	10179	2449	11276	1.1/4.6
	Movie–Actor	11718	6311	33587	2.9/5.3
	Movie–Type	12676	38	27668	2.2/728.1
Yelp	User–Business	16239	14284	198397	12.2/13.9
	User–User	10580	10580	158590	15.0/15.0
	User–Compliment	14411	11	76875	5.3/6988.6
	Business–City	14267	47	14267	1.0/303.6
	Business–Category	14180	511	40009	2.8/78.3

where $R_{u,i}$ denotes the real rating user u gave to item i and $\hat{R}_{u,i}$ denotes the predicted rating. R_{test} denotes whole test set. A smaller MAE or RMSE means a better performance.

In order to show the effectiveness of the proposed SemRec, we compare four variations of SemRec with the state of the arts. Besides the personalized weight learning method with weight regularization (called $\text{SemRec}_{\text{Reg}}$), we include three special cases of SemRec: single path-based method (called $\text{SemRec}_{\text{Sgl}}$), unified weight learning method for all users (called $\text{SemRec}_{\text{All}}$), and personalized weight learning method for individual user (called $\text{SemRec}_{\text{Ind}}$). As the baselines, four representative rating predication methods are illustrated as follows. Note that the top k recommendation methods [5, 24] are not included here, since they solve different problems.

- **PMF** [14]: It is the basic matrix factorization method using only user–item matrix for recommendations.
- **SMF** [13]: It adds the social regularization term into PMF, which aims at getting the users' latent factor closer to their friends' latent factors.
- **CMF** [8]: A collective matrix factorization method, which factorizes all relations in HIN and shares the latent factor of same object types in different relations.
- **HeteMF** [22]: A matrix factorization method with entity similarity regularization, which also utilizes the relations in HIN.

We employ 5 meaningful meta paths whose lengths are not longer than 4 for both datasets, since the longer meta paths are not meaningful and they fail to produce good similarity measures [19]. Table 5.3 shows those paths which include the weighted and unweighted meta paths. For SemRec, we use PathSim [19] as the similarity measure

Table 5.3 Meta paths used in experiments

Douban	Yelp
UGU	UU
U(i)M(j)U $\|i = j$	UCoU
U(i)MDM(j)U $\|i = j$	U(i)B(j)U $\|i = j$
U(i)MAM(j)U $\|i = j$	U(i)BCaB(j)U $\|i = j$
U(i)MTM(j)U $\|i = j$	U(i)BCiB(j)U $\|i = j$

to calculate the similarity between users. The parameter λ_0 in SemRec is 0.01, and λ_1 is 10^3 for the best performance. The parameters in other methods are set with the best performances on these datasets.

5.1.4.2 Effectiveness Experiments

For Douban dataset, we use different training data settings (20%, 40%, 60%, 80%) to show the comparison results in different data sparseness. Training data 20%, for example, means that 20% of the ratings from user–item rating matrix is randomly selected as the training data to predict the remaining 80%. From Table 5.2, we can find that the Douban dataset has dense rating relations, while Yelp has very sparse rating relations. So we utilize more training data (60%, 70%, 80%, 90%) on Yelp. The random selection was repeated 10 times independently, and the average results are reported in Table 5.4. Note that $\text{SemRec}_{\text{Sgl}}$ reports the best performances on these five paths.

From the results, we can observe that all versions of SemRec outperform other approaches in most conditions. Particularly, $\text{SemRec}_{\text{Reg}}$ always achieves the best performances on all conditions. For example, on 20% training set of Douban, $\text{SemRec}_{\text{Reg}}$ outperforms PMF up to 19.55% on RSME and 15.89% on MAE. As compared to PMF, CMF improves the recommendation performances through integrating heterogeneous information with matrix factorization. However, its performances are much worse than the proposed SemRec on all conditions, especially on less training set. As the most similar method to SemRec, HeteMF also has good performances, while its performances are still worse than the proposed $\text{SemRec}_{\text{Reg}}$. These all imply that the proposed SemRec has better mechanism to integrate heterogeneous information.

In addition, different versions of SemRec have different performances. Generally, SemRec with multiple paths (e.g., $\text{SemRec}_{\text{All}}$ and $\text{SemRec}_{\text{Reg}}$) have better performances than SemRec with single path (i.e., $\text{SemRec}_{\text{Sgl}}$) except $\text{SemRec}_{\text{Ind}}$, which indicates that the weight learning of SemRec can effectively integrate the similarity information generated by different paths. Because of rating sparsity, $\text{SemRec}_{\text{Ind}}$ has worse performances than $\text{SemRec}_{\text{All}}$ on most conditions. In addition, the better performances of $\text{SemRec}_{\text{Rec}}$ over $\text{SemRec}_{\text{Ind}}$ confirm the benefit of the weight regularization term. In all, $\text{SemRec}_{\text{Reg}}$ always achieves best performances in all conditions.

Table 5.4 Effectiveness experimental results (Res. and Imp. are the abbreviations of result and improvement. The improvement is based on PMF)

Dataset	Method	Criteria	20%		40%		60%		80%		Running Time(s)
			Res.	Imp.	Res.	Imp.	Res.	Imp.	Res.	Imp.	
Douban	**PMF**	**RMSE**	0.9750		0.8455		0.7975		0.7673		260.25
		MAE	0.7198		0.6319		0.6010		0.5812		
	SMF	**RMSE**	0.9743	0.07%	0.8449	0.07%	0.7967	0.10%	0.7674	−0.01%	266.78
		MAE	0.7192	0.08%	0.6313	0.09%	0.6002	0.13%	0.5815	−0.05%	
	CMF	**RMSE**	0.9285	4.77%	0.8273	2.15%	0.8042	−0.84%	0.7741	−0.89%	509.31
		MAE	0.6971	3.15%	0.6263	0.89%	0.6090	−1.33%	0.5900	−1.51%	
	HeteMF	**RMSE**	0.8513	12.69%	0.7796	7.79%	0.7601	4.69%	0.7550	1.60%	736.85
		MAE	0.6342	11.89%	0.5927	6.20%	0.5800	3.49%	0.5758	0.93%	
	SemRec$_{\mathbf{Sgl}}$	**RMSE**	0.8434	13.50%	0.8138	3.75%	0.7937	0.48%	0.7846	−2.25%	0
		MAE	0.6506	9.61%	0.6351	−0.51%	0.6172	−2.70%	0.6142	−5.68%	
	SemRec$_{\mathbf{All}}$	**RMSE**	0.8125	16.67%	0.7814	7.58%	0.7709	3.34%	0.7656	0.22%	1.44
		MAE	0.6309	12.35%	0.6149	2.69%	0.6098	−1.46%	0.6072	−4.47%	
	SemRec$_{\mathbf{Ind}}$	**RMSE**	0.8753	10.23%	0.8083	4.40%	0.7729	3.08%	0.7540	1.73%	155.98
		MAE	0.6412	10.92%	0.6032	4.54%	0.5840	2.83%	0.5739	1.26%	
	SemRec$_{\mathbf{Reg}}$	**RMSE**	0.7844	19.55%	0.7452	11.86%	0.7296	8.51%	0.7216	5.96%	293.14
		MAE	0.6054	15.89%	0.5808	8.09%	0.5698	5.19%	0.5639	2.98%	

(continued)

Table 5.4 (continued)

Dataset	Method	Criteria	20%		40%		60%		80%		Running Time(s)
			Res.	lmp.	Res.	lmp.	Res.	lmp.	Res.	lmp.	
Yelp	**PMF**	**RMSE**	1.6779		1.5931		1.5323		1.4833		31.8
		MAE	1.2997		1.2262		1.1740		1.1324		
	SMF	**RMSE**	1.4843	11.54%	1.4017	12.01%	1.3678	10.74%	1.3377	9.82%	51.19
		MAE	1.0830	16.67%	1.0547	13.99%	1.0282	12.42%	1.0085	10.94%	
	CMF	**RMSE**	1.6161	3.68%	1.5731	1.26%	1.5194	0.84%	1.4793	0.27%	375.38
		MAE	1.2628	2.84%	1.2224	0.31%	1.1740	0.00%	1.1405	−0.72%	
	HeteMF	**RMSE**	1.2333	26.50%	1.2090	24.11%	1.1895	22.37%	1.1755	20.75%	619.25
		MAE	0.9268	28.69%	0.9107	25.73%	0.8969	23.60%	0.8878	21.60%	
	SemRec$_{Sgl}$	**RMSE**	1.3252	21.02%	1.2889	19.09%	1.2576	17.93%	1.2331	16.87%	0
		MAE	0.9657	25.70%	0.9420	23.18%	0.9224	21.43%	0.9067	19.93%	
	SemRec$_{All}$	**RMSE**	1.2166	27.49%	1.1906	25.27%	1.1665	23.87%	1.1496	22.50%	0.25
		MAE	0.9040	30.45%	0.8873	27.64%	0.8723	25.70%	0.8616	23.91%	
	SemRec$_{Ind}$	**RMSE**	1.3654	18.62%	1.3229	16.96%	1.2922	15.67%	1.2658	14.66%	57.22
		MAE	1.0029	22.84%	0.9728	20.67%	0.9517	18.94%	0.9322	17.68%	
	SemRec$_{Reg}$	**RMSE**	1.2025	28.33%	1.1760	26.18%	1.1559	24.56%	1.1423	22.99%	374.57
		MAE	0.8901	31.51%	0.8696	29.08%	0.8548	27.19%	0.8442	25.45%	

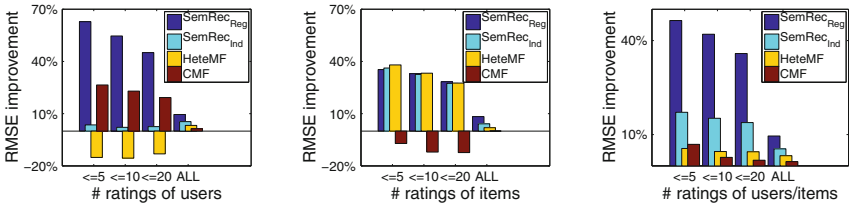
The reason lies in that $\text{SemRec}_{\text{Reg}}$ not only realizes personalized weight learning for all users but also avoids the rating sparsity through the weight regularization in it.

Furthermore, we record the average running time of these methods on the learning process. For two similarity based methods (e.g., SemRec and HeteMF), we do not consider the running time on similarity evaluation, since it can be done off-line beforehand. For the four versions of SemRec, their running times increase when the weight learning tasks become more complex. Both $\text{SemRec}_{\text{Sgl}}$ and $\text{SemRec}_{\text{All}}$ are very fast, which can be applied for online learning. The running times of $\text{SemRec}_{\text{Ind}}$ and $\text{SemRec}_{\text{Reg}}$ are still acceptable when compared to CMF and HeteMF. We can select a proper model through balancing the efficiency and effectiveness of SemRec in real applications.

5.1.4.3 Study on Cold-Start Problem

The above results also show that SemRec has more obvious superiority with less training set, which implies that SemRec has the potential to alleviate the cold-start problem. In this section, we will exploit the ability of SemRec on alleviating the cold-start problem through observing its performances on different levels of cold-start users and items. We run PMF, CMF, HeteMF, $\text{SemRec}_{\text{Ind}}$, and $\text{SemRec}_{\text{Reg}}$ on Douban dataset with users having the different numbers of rated movies. We select four types of users: three types of cold-start users with different numbers of rated movies (e.g., users with the number of rated movies no more than 5, denoted as ≤ 5 in Fig. 5.4) and all users (called ALL in Fig. 5.4). In addition, we also do the similar experiments on cold-start items and users&items (contain both cold-start users and items). We record the RMSE performance improvement of other four algorithms against PMF in Fig. 5.4.

It is clear that $\text{SemRec}_{\text{Reg}}$ always achieves the best performance improvements on almost all conditions, and its superiority is more significant for less rating information. On the contrary, CMF only achieves improvements on cold-start users and HeteMF's improvements are only on items. We think the reason lies in that the collective matrix factorization of all relations in CMF may introduce much noises, especially for items. HeteMF only utilizes the similarity information of items, ignoring



(a) Users (b) Items (c) Users&Items

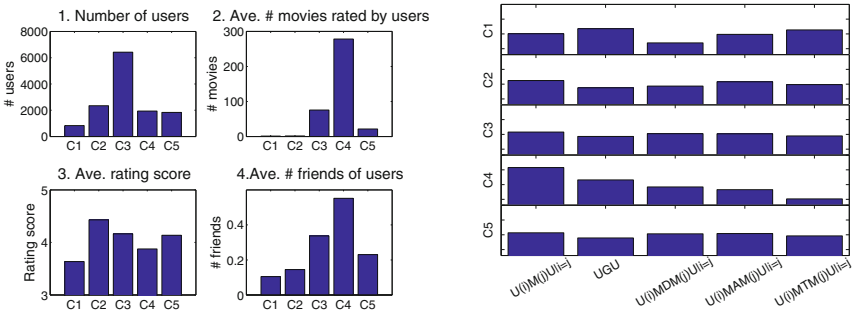
Fig. 5.4 Performance improvements of three HIN methods against PMF on different levels and types of cold-start problems

that of users. Generally, integrating heterogeneous information is helpful in alleviating cold-start problem (see Fig. 5.4c), while the integrating mechanisms may have different impacts on cold-start items and users. The overall performance improvements of $\text{SemRec}_{\text{Reg}}$ are attributed to multiple meta paths that not only contain rich attribute information but also provide comprehensive and complementary similarity evaluation of users and items. In addition, the better performances of $\text{SemRec}_{\text{Reg}}$ over $\text{SemRec}_{\text{Ind}}$ further validate that the weight regularization term employed in $\text{SemRec}_{\text{Reg}}$ is really helpful for the weight learning of cold-start users from similar users.

5.1.4.4 Study of Weight Preferences

In this section, we illustrate the meanings of weights learned by SemRec through a case study. Based on the results of $\text{SemRec}_{\text{Reg}}$ on Douban dataset with 60% training data in the above experiments, we cluster users' weight vectors into 5 groups using K-means and then show the statistics information of users in five clusters in Fig. 5.5a. Moreover, the weight preferences of the five cluster centers on 5 meta paths are also shown in Fig. 5.5b.

Let us observe the relationship of the statistics information of users in different clusters and their weight preferences on paths from Fig. 5.5a, b. As we know, Douban is a unique social media platform in China, in which the major active users are young people who love culture and arts. As the typical and major users in Douban, the users in C3 view a good number of movies, give relatively good rating scores, and have a moderate number of friends. So they also have close weight preferences on all paths. As the top movie fans, the users in C4 view a great many movies, tend to give lower rating scores due to critical attitude, and have many friends. And they obviously like to get recommendation from viewing records of other users (i.e., UMU) and interest group (i.e., UGU), but less paying attentions to movies' content (e.g., UMTMU and



(a) Statistics of clusters (b) Weights on meta paths

Fig. 5.5 Analysis of clusters' characteristics and path preferences of results returned by $\text{SemRec}_{\text{Reg}}$ on Douban dataset. C1–C5 represents the index of five clusters

UMAMU). In addition, the users in C1 and C2 are two types of inactive users, and they view few movies and have few friends. Because of not being fond of movies, these users tend to give much high or low rating scores. These users comparatively prefer to follow movie content (e.g., UMTMU and UMAMU). The picky users in C1 is more likely to get recommendation from interest group (i.e., UGU), while the idealess users in C2 give more preferences to viewing records of other users (i.e., UMU).

In all, the weights of paths learned by SemRec can reflect the users' path preferences, and these path preferences are able to reveal the users' characteristics to a large extent. More importantly, the meaningful weight preferences are very useful for recommendation explanation. We know that the meta path has semantics, so we can tell users the recommendation reason according to the path semantics of the high-weight path. Although some weight learning methods on paths have been proposed [9, 24], their weights fail to reflect users' preferences on paths. We think two strategies adopted in RecSem contribute to its good properties. (1) We design the predicted rating score in Eq. 5.2, which can eliminate the similarity bias on different meta paths by the adoption of normalized rating intensity. (2) We employ the weight regularization term in Eq. 5.9 according to the consistency rule of weight preferences of similar users. The consistency rule makes similar users have similar weight preferences. In other words, weights also reveal users' similarity and preferences.

5.1.4.5 Study on Weighted Meta Path

In this section, we study the effectiveness of weighted meta path on improving the performances of SemRec through more accurately revealing relations among objects. For the meta path UMU, we design two weighted paths $U(i)M(j)U|i = j$ and $U(i)M(j)U||i - j| \leq 1$. $U(i)M(j)U|i = j$ means users rating the exact same scores on the same movies, while $U(i)M(j)U||i - j| \leq 1$ means users rating close scores. Similarly, we design two corresponding weighted paths for UMDMU, UMAMU, and UMTMU. Based on the similarity generated by these meta paths, we employ $\text{SemRec}_{\text{Sgl}}$ to make recommendations. We compare the performances of $\text{SemRec}_{\text{Sgl}}$ with different paths and record the results in Fig. 5.6.

The experimental results on all four paths clearly show that SemRec with weighted meta paths (e.g., $U(i)M(j)U|i = j$ and $U(i)M(j)U||i - j| \leq 1$) significantly outperform SemRec with unweighted meta paths (e.g., UMU). Let us take the UMU path as an example to analyze the reasons. Failing to distinguish the different rating scores of users on the same movies, UMU cannot accurately reveal user similarity, so it has bad performances. The path $U(i)M(j)U|i = j$ and $U(i)M(j)U||i - j| \leq 1$ not only considers the differences of rating scores but also keeps dense relations, so they can achieve better performances than UMU. Compared to $U(i)M(j)U|i = j$, the relatively bad performances of $U(i)M(j)U||i - j| \leq 1$ may be attributed to the noise introduced by some improper relation constraints (e.g., U(3)M(4)U, and U(4)M(3)U). The experiments illustrate that the weighted meta paths are really

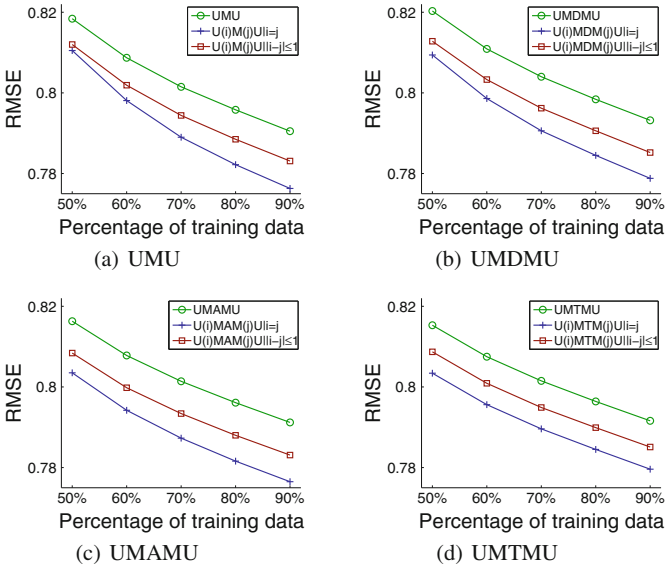


Fig. 5.6 Performances of SemRec with different weighted meta paths

helpful to improve recommendation performances by more accurately revealing object relations.

5.2 Recommendation Based on Matrix Factorization

5.2.1 Overview

With the increasing popularity of social media, there is a surge of social recommendation techniques [4, 12] in recent years, which leverage rich social relations among users, such as friendships in Facebook, following relations in Twitter. However, the emerging social recommendation usually faces the problem of relation sparsity. On the one hand, dense social relations can improve the recommendation performance. However, social relations are very sparse or absent in many real applications. For example, there are no social relations in Amazon, and 80% users in Yelp have less than 3 following relations. On the other hand, users and items in many applications have rich attribute information, which are seldom exploited. These information may be very useful to reveal users' tastes and items' properties. For example, the group attribute of users can reflect their interests, and the type attribute of movies can reveal the content of movies. So it is desirable to effectively integrate all kinds of information for better recommendation performance, including not only feedback and social relations but also attributes of users and items. Some works have began to explore

this issue [5, 23, 24], while they did not focus on revealing the importance of these attributes and their effects on recommendation accuracy.

Although integrating more information is promising to achieve better recommendation performance, how to integrate these information still faces two challenges. (1) The information to be integrated has different types. These mixed information types include integer (i.e., rating information), vector (i.e., attribute information), and graph (i.e., social relations). We need to design a unified model to effectively integrate these different types of information. (2) A unified and flexible method is desirable to integrate all or some of these information. In order to intensively study the impacts of different information, the designed method should flexibly integrate different granularities of information and uniformly utilize different types of information.

As mentioned above, we can organize objects and relations in recommended system as a heterogeneous information network which contains different types of nodes or links. In order to utilize these heterogeneous information, we introduce meta path-based similarity measure to evaluate the similarity betweeof users and items. Based on matrix factorization, a dual regularization framework SimMF is proposed to integrate heterogeneous information through adopting similarity information of users and items as regularization on latent factors of users and items. Moreover, in SimMF, two different regularization models, average-based regularization and individual-based regularization, can flexibly confine regularization on users or items.

5.2.2 The SimMF Method

In this section, we will introduce the **SimMF** method, which utilizes **matrix factorization** framework to incorporate **sim**ilarity information. We firstly introduce the rich similarity generation with HIN. And then, we review the basic low-rank matrix factorization framework and introduce the improved model through constraining similarity regularization on users and items, respectively. Finally, we show the unified model through applying similarity regularization on users and items simultaneously.

5.2.2.1 Similarity Generation

Two objects in a heterogeneous network can be connected via different paths, which can be called meta path [19]. A meta path P is a path defined on a schema $S = (\mathtt{A}, \mathtt{R})$ and is denoted in the form of $A_1 \xrightarrow{R_1} A_2 \xrightarrow{R_2} \cdots \xrightarrow{R_l} A_{l+1}$ (abbreviated as $A_1 A_2 \cdots A_{l+1}$), which defines a composite relation $R = R_1 \circ R_2 \circ \cdots \circ R_l$ between type A_1 and A_{l+1}, where $\circ$ denotes the composition operator on relations. Since different meta paths have different semantics, objects connecting by different meta paths have different similarity. So we can evaluate the similarity of users (or movies) based on different meta paths. For example, for users, we can consider meta paths

UU, UGU, UMU, and so on. Similarly, meaningful meta paths connecting movies include MAM and MDM.

There are several path-based similarity measures to evaluate the similarity of objects in HIN [6, 16, 19]. Considering semantics in meta paths, Sun et al. [19] proposed PathSim to measure the similarity of same-type objects based on symmetric paths. Lao and Cohen [6] proposed a Path Constrained Random Walk (PCRW) model to measure the entity proximity in a labeled directed graph constructed by the rich metadata of scientific literature. The HeteSim [16] can measure the relatedness of heterogeneous objects based on an arbitrary meta path. All these similarity measures can be used in the similarity calculation, and their differences can be seen in Ref. [16].

We define $S_{ij}^{(l)}$ to denote the similarity of two objects u_i and u_j under the given meta path P_l. The similarity ($\mathtt{S}$) is determined by the given meta path (P) and the similarity measure ($\mathtt{M}$). That is, $\mathtt{S} = P \times \mathtt{M}$. We know that the similarity of different paths are different and they are incomparable. So we normalize them with *Sigmoid* function as shown in Eq. 5.14, where $\bar{S}^{(l)}$ means the average of $S_{ij}^{(l)}$ and β is set to 1. The normalization process has the following two advantages. (1) It confines the similarity into [0, 1] without changing their ranking. (2) It can reduce the similarity difference of different paths. In the following section, we directly use the $S_{ij}^{(l)}$ to represent the normalized similarity:

$$S_{ij}^{(l)'} = \frac{1}{1 + e^{-\beta \times (S_{ij}^{(l)} - \bar{S}^{(l)})}} \tag{5.14}$$

Since users (or items) have different similarity under different meta paths, we consider their similarity on all paths through assigning weights on different paths. For users, we define $S^{\mathtt{U}}$ for the similarity matrix of users on all paths and $S^{\mathtt{I}}$ for the similarity matrix of items on all paths. They can be defined as follows, where $w_l^{\mathtt{U}}$ represents the weight of similarity matrix of users under the path P_l and $w_l^{\mathtt{I}}$ represents that of items:

$$\begin{aligned} S^{\mathtt{U}} &= \textstyle\sum_l w_l^{\mathtt{U}} S^{(l)} \;\; \Sigma_l w_l^{\mathtt{U}} = 1; 0 \le w_l^{\mathtt{U}} \le 1 \\ S^{\mathtt{I}} &= \textstyle\sum_l w_l^{\mathtt{I}} S^{(l)} \;\; \Sigma_l w_l^{\mathtt{I}} = 1; 0 \le w_l^{\mathtt{I}} \le 1 \end{aligned} \tag{5.15}$$

5.2.2.2 Low-Rank Matrix Factorization

The low-rank matrix factorization has been widely studied in recommended system [18]. Its basic idea is to factorize the user–item rating matrix R into two matrices (U and V) representing users' and items' distributions on latent semantic, respectively. Then, the rating prediction can be made through these two specific matrices. Assuming an $m \times n$ rating matrix R to be m users' ratings on n items, this approach mainly minimizes the objective function $\mathtt{L}(R, U, V)$ as follows:

$$\min_{U,V} \mathtt{L}(R, U, V) = \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{n} I_{ij} (R_{ij} - U_i V_j^T)^2 + \frac{\lambda_1}{2} \|U\|^2 + \frac{\lambda_2}{2} \|V\|^2, \tag{5.16}$$

where I_{ij} is the indicator function that is equal to 1 if user i rates item j and equal to 0 otherwise. $U \in \mathbb{R}^{m \times d}$ and $V \in \mathbb{R}^{n \times d}$, where d is the dimension of latent factors and $d \ll min(m, n)$. U_i is a row vector derived from the ith row of matrix U, and V_j is a row vector derived from the jth row of matrix V. λ_1 and λ_2 represent the regularization parameters. In summary, the optimization problem minimizes the sum-of-squared-errors objective function with quadratic regularization terms which aim to avoid overfitting. This problem can be effectively solved by a simple stochastic gradient descent technique.

5.2.2.3 Similarity Regularization on Users and Items

As mentioned above, the user-specific factorized matrix describes users' distribution over latent semantic. In this section, we will introduce two different types of similarity regularization (i.e., average-based regularization and individual-based regularization) on users to force the distance between U_p and U_q to be much smaller if user p is highly similar to user q.

Average-based Regularization Intuitively, we have similar behavior model with people who are similar with us. That is, the latent factor of a user is similar to the latent factor of people who are the most similar to the user. Based on this assumption, we add user's similarity regularization to the basic low-rank matrix factorization framework.

$$\min_{U,V} \mathrm{L}(R, U, V) = \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{n} I_{ij} (R_{ij} - U_i V_j^T)^2 + \frac{\alpha}{2} \sum_{i=1}^{m} \| U_i - \frac{\sum_{f \in \mathrm{T}_u^+(i)} S_{if}^{\mathrm{U}} U_f}{\sum_{f \in \mathrm{T}_u^+(i)} S_{if}^{\mathrm{U}}} \|^2 + \frac{\lambda_1}{2} \|U\|^2 + \frac{\lambda_2}{2} \|V\|^2 \tag{5.17}$$

where $\mathrm{T}_u^+(i)$ is the set of users who are in the top k similarity list of user i and S_{if}^{U} is the element located on the ith row and the fth column of user similarity matrix S^{U}. The average-based regularization confines that the latent factor of a user is close to the average of the latent factor of the top k similar people to the user. The analogous regularization has been used in social recommendation [13], while it just enforces constraints on friends of users. Here, the average-based regularization not only extends to the top k similarity list of users but also considers the similarity values as the weights. The parameter k can be set to trade-off accuracy and computation cost. Large k usually means high accuracy but low efficiency. In our experiments, k is set to 5% of the vector dimension. A local minimum of the objective function given by Eq. 5.17 can be solved by performing gradient descent in feature vectors U_i and V_j, which is shown in Eqs. 5.18 and 5.19. Here, $\mathrm{T}_u^-(i)$ represents the set of users whose top k similarity list contains user i.

$$\frac{\partial \mathtt{L}}{\partial U_i} = \sum_{j=1}^{n} I_{ij}(U_i V_j^T - R_{ij})V_j + \alpha(U_i - \frac{\sum_{f\in \mathtt{T}_u^+(i)}(S_{if}^{\mathtt{U}} U_f)}{\sum_{f\in \mathtt{T}_u^+(i)} S_{if}^{\mathtt{U}}}) + \alpha \sum_{g\in \mathtt{T}_u^-(i)} \frac{-S_{ig}^{\mathtt{U}}(U_g - \frac{\sum_{f\in \mathtt{T}_u^+(g)}(S_{gf}^{\mathtt{U}} U_f)}{\sum_{f\in \mathtt{T}_u^+(g)} S_{gf}^{\mathtt{U}}})}{\sum_{f\in \mathtt{T}_u^+(g)} S_{gf}^{\mathtt{U}}} + \lambda_1 U_i, \tag{5.18}$$

$$\frac{\partial \mathtt{L}}{\partial V_j} = \sum_{i=1}^{m} I_{ij}(U_i V_j^T - R_{ij})U_i + \lambda_2 V_j. \tag{5.19}$$

Individual-based Regularization The above average-based regularization constrains user's taste with the average taste of people who are the most similar users. However, it may be ineffective for users whose similar users have diverse tastes. In order to avoid this disadvantage, we employ individual-based regularization on users as follows:

$$\min_{U,V} \mathtt{L}(R, U, V) = \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{n} I_{ij}(R_{ij} - U_i V_j^T)^2 + \frac{\alpha}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} S_{ij}^{\mathtt{U}}\|U_i - U_j\|^2 + \frac{\lambda_1}{2}\|U\|^2 + \frac{\lambda_2}{2}\|V\|^2. \tag{5.20}$$

In essential, the individual-based regularization enforces a large $S_{ij}^{\mathtt{U}}$ to have a small distance between U_i and U_j. That is, similar users have smaller distance on latent factors. With the same optimization technique, a local minimum of Eq. 5.20 can also be found by performing gradient descent in U_i and V_j.

$$\frac{\partial \mathtt{L}}{\partial U_i} = \sum_{j=1}^{n} I_{ij}(U_i V_j^T - R_{ij})V_j + \alpha\sum_{j=1}^{m}(S_{ij}^{\mathtt{U}} + S_{ji}^{\mathtt{U}})(U_i - U_j) + \lambda_1 U_i, \tag{5.21}$$

$$\frac{\partial \mathtt{L}}{\partial V_j} = \sum_{i=1}^{m} I_{ij}(U_i V_j^T - R_{ij})U_i + \lambda_2 V_j. \tag{5.22}$$

Similarity Regularization on Items For simplicity, we define the notation Reg_y^x to represent the average-based or individual-based regularization term on users or items, where $x \in \{\mathtt{U}, \mathtt{I}\}$ means Users or Items and $y \in \{ave, ind\}$ means *ave*rage-based or *ind*ividual-based regularization. That is, for similarity regularization on users, we have

$$Reg^{\mathtt{U}}_{ave} = \sum_{i=1}^{m} \|U_i - \frac{\sum_{f \in \mathtt{T}^{+}_{u}(i)} S^{\mathtt{U}}_{if} U_f}{\sum_{f \in \mathtt{T}^{+}_{u}(i)} S^{\mathtt{U}}_{if}}\|^2, \tag{5.23}$$

$$Reg^{\mathtt{U}}_{ind} = \sum_{i=1}^{m} \sum_{j=1}^{m} S^{\mathtt{U}}_{ij} \|U_i - U_j\|^2. \tag{5.24}$$

Similar to the regularization on users, we can also define these two different types of regularization on items as follows:

$$Reg^{\mathtt{I}}_{ave} = \sum_{j=1}^{n} \|V_j - \frac{\sum_{f \in \mathtt{T}^{+}_{i}(j)} S^{\mathtt{I}}_{jf} V_f}{\sum_{f \in \mathtt{T}^{+}_{i}(j)} S^{\mathtt{I}}_{jf}}\|^2, \tag{5.25}$$

$$Reg^{\mathtt{I}}_{ind} = \sum_{i=1}^{n} \sum_{j=1}^{n} S^{\mathtt{I}}_{ij} \|V_i - V_j\|^2, \tag{5.26}$$

where $\mathtt{T}^{+}_{i}(j)$ is the set of items who are in the top k similarity list of item j, and $S^{\mathtt{I}}_{jf}$ is the element located on the jth row and the fth column of similarity matrix $S^{\mathtt{I}}$. We can also define the optimization function based on these two regularization terms on items and derive their gradient learning algorithms as above.

5.2.2.4 A Unified Dual Regularization

Now, we consider regularization on users and items simultaneously. The corresponding optimization function is shown as follows:

$$\begin{aligned} \min_{U,V} \mathtt{L}(R, U, V) = & \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{n} I_{ij} (R_{ij} - U_i V_j^T)^2 + \frac{\alpha}{2} Reg^{\mathtt{U}}_{y} + \frac{\beta}{2} Reg^{\mathtt{I}}_{y} \\ & + \frac{\lambda_1}{2} \|U\|^2 + \frac{\lambda_2}{2} \|V\|^2, \end{aligned} \tag{5.27}$$

where α and β control the effect of user and item regularization, respectively. For $y \in \{ave, ind\}$, there are four regularization models. Similarly, we can use the gradient descent method to solve this optimization problem. The whole algorithm framework is shown in Algorithm 2.

5.2.3 Experiments

In this section, we will verify the superiority of our model by conducting a series of experiments compared to the state-of-the-art recommendation methods.

Algorithm 2 Algorithm Framework of SimMF

Require:
G: heterogeneous information network
P_U, P_I: sets of meta paths related to users and items
η: learning rate for gradient descent
α, β, λ_1, λ_2: controlling parameters defined above
ε: convergence tolerance
Ensure:
U, V: the latent factor of users and items

1: Calculate similarity matrix of user S_{U} based on P_U, G
2: Calculate similarity matrix of item S_{I} based on P_I, G
3: Initialize U, V
4: **repeat**
5: $U_{old} := U$,$V_{old} := V$
6: Calculate $\frac{\partial \mathrm{L}}{\partial U}$, $\frac{\partial \mathrm{L}}{\partial V}$
7: Update $U := U - \eta * \frac{\partial \mathrm{L}}{\partial U}$
8: Update $V := V - \eta * \frac{\partial \mathrm{L}}{\partial V}$
9: **until** $\|U - U_{old}\|^2 + \|V - V_{old}\|^2 < \varepsilon$

5.2.3.1 Experiment Settings

In experiments, we employs two real datasets from two various domains. Douban Movie[3] is from the movie domain. Stemming from the business domain, the widely used Yelp challenge dataset[4] [23, 24] records users' ratings on local business and also contains social relations and attribute information of business (e.g., cities and categories). In addition, we use Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) to evaluate the performance of different methods.

In this section, we compare SimMF with six representative methods. There are different variations for SimMF. We use SimMF-U(y)I(y) to represent SimMF with regularization on users and items, where $y \in \{a, i\}$, and it represents the average-based or individual-based regularization. Similarly, SimMF-U(y) (SimMF-I(y)) means SimMF with regularization only on users (items). There are six baseline methods, including four types. There are two basic methods (i.e., UserMean and ItemMean), a collaborative filtering with low-rank matrix factorization (i.e., PMF), a social recommendation method (i.e., SoMF), and two HIN-based methods (i.e., HeteMF and HeteCF). These baselines are summarized as follows.

- **UserMean**. This method uses the mean value of every user to predict the missing values.
- **ItemMean**. This method utilizes the mean value of every item to predict the missing values.

[3] http://movie.douban.com/.

[4] http://www.yelp.com/dataset_challenge/.

- **PMF**. This method is a typical matrix factorization method proposed by Salakhutdinov and Minh [15]. And in fact, it is equivalent to basic low-rank matrix factorization in the previous section.
- **SoMF**. This is the matrix factorization-based recommendation method with social average-based regularization proposed by Ma et al. [13].
- **HeteMF**. This is the matrix factorization-based recommendation framework combining user ratings and various entity similarity matrices proposed by Yu et al. [22].
- **HeteCF**. This is the social collaborative filtering algorithm using heterogeneous relations [9].

We employ HeteSim [16] to evaluate the similarity of objects. For the Douban Movie dataset, we use 7 meaningful meta paths for user whose length is smaller than 4 (i.e., UU, UGU, ULU, UMU, UMDMU, UMTMU, UMAMU) and 5 meaningful meta paths for movie whose length is smaller than 3 (i.e., MTM, MDM, MAM, MUM, MUUM). For the Yelp dataset, we use 4 meta paths for user (i.e., UU, UBU, UBCBU, UBLBU) and 4 meta paths for business (i.e., BUB, BCB, BLB, BUUB). Similarly, we utilize 5 meta paths for user (i.e., UGU, UAU, UOU, UMU, UMTMU) and 2 meta paths for movie (i.e., MTM, MUM) for the MovieLens dataset. And for the Douban Book dataset, we utilize 7 meta paths for user (i.e., UU, UGU, ULU, UBU, UBABU, UBPBU, UBYBU) and 5 meta paths for book (i.e., BAB, BPB, BYB, BUB, BUUB). These similarity data are fairly used for HeteCF and SimMF. HeteMF uses similarity data of users, since the model only considers the similarity relationships between items.

5.2.3.2 Effectiveness Experiments

This section will validate the effectiveness of SimMF through comparing its different variations to baselines. Here, we run four versions of SimMF-U(y)I(y) ($y \in \{a, i\}$) and record the worst (denoted as SimMF-max in Tables 5.5 and 5.6), the best (denoted as SimMF-min), and the average (denoted as SimMF-mean) performance of these four versions. The α and β are set to 100 and 10, respectively, for Douban Movie dataset, as suggested in the following parameter experiment. For other datasets, α and β are set to the optimal values according to related parameter experiments. For all the experiments in this chapter, the values of λ_1 and λ_2 are set to a trivial value 0.001 and the length of latent feature vectors U_i and V_j are set to 10. The parameters of other methods are set to the optimal values obtained in parameter experiments.

For these datasets, we use different ratios (80%, 60%, 40%, 20%) of data as training set. For example, the training data 80% means that we select 80% of the ratings from user–item rating matrix as the training data to predict the remaining 20% of ratings. The random selection was carried out 10 times independently in all the experiments. We report average results on Douban Movie and Yelp datasets in Tables 5.5 and 5.6, respectively, and record the improvement ratio of all methods compared to the PMF. In addition, we also report the average running time of these methods with the 80% training ratio in the last line of above tables. For those

HIN-based methods (i.e., HeteCF, HeteMF, and SimMF), we only report the running time of the model learning process, ignoring the running time of similarity computation. Note that we report the mean running time for SimMF, since the four versions of SimMF have the similar computational complexity.

The results are shown in Tables 5.5 and 5.6. In addition, we also conduct the t-test experiments with 95% confidence, which shows that the MAE/RMSE improvement difference is statistically stable and non-contingent. Due to the space limitation, they are omitted in the paper, but the results can be found in [17]. From the experimental comparisons, we can observe the following phenomena.

- SimMF always outperforms the baselines in most conditions, even for the worst performance of SimMF (i.e., SimMF-max). It validates that more attribute information from users and items exploited in SimMF is really helpful to improve the recommendation performance. In addition, the model integrating more information usually has better performances. That is, the reason why other matrix factorization models integrating heterogeneous information usually have better performance than the basic matrix factorization model PMF.
- Although HeteMF and HeteCF also utilize the attribute information from users and items, they have worse performance than SimMF, which implies the proposed SimMF has better mechanism to integrate heterogeneous information. We know that HeteMF only integrates attribute information of items, while the same parameter for similarity regularization terms of users and items may cause the bad performance of HeteCF.
- When considering different training data ratios, we can find that the superiority of SimMF is more significant for less training data. It indicates that SimMF can effectively alleviate data sparsity problem. We think the reason lies in that, through exploiting different meta paths, we can make full use of rich attribute information of users and items to reflect the similarity of users and items from different aspects. The integration of similarities can comprehensively reveal the similarity of users and items, which compensates for shortage of training data.

Observing the running time of different methods in the last row of Tables 5.5 and 5.6, we can find that the running time becomes longer as the models become more complex. That is, HIN-based methods (i.e., HeteMF, HeteCF, and SimMF) have longer running time than the other methods, since they have more parameters to be learned. However, SimMF is still faster than the other two HIN-based methods because SimMF does not need to learn the weights of meta paths.

5.2.3.3 Impact of Different Regularizations

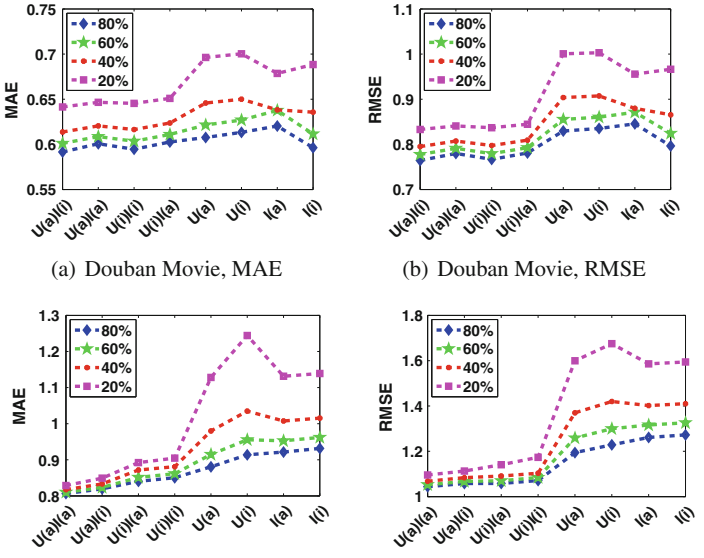
Experiments in this section will validate the effect of different regularization models on users and items. Ma et al. [13] have explored the effect of average-based and individual-based regularization on social relations of users. However, in this chapter, we not only explore the effect on more complex relations, but also consider the effect on both users and items.

Table 5.5 Performance comparisons on Douban Movie (the baseline of improved performance is PMF)

Training	Metrics	UserMean	ItemMean	PMF	SoMF	HeteMF	HeteCF	SimMF-mean	SimMF-max	SimMF-min
80%	MAE	0.6958	0.6476	0.6325	0.6073	0.6221	0.6273	0.5974	0.6026	**0.5926**
	Improve	−10.01%	−2.83%		3.99%	1.64%	0.82%	5.55%	4.73%	6.31%
	RMSE	0.8846	0.8537	0.8815	0.8283	0.8609	0.8664	0.7729	0.7809	**0.7656**
	Improve	−0.35%	3.15%		6.03%	2.34%	1.71%	12.32%	11.41%	13.14%
60%	MAE	0.6986	0.6557	0.6591	0.6219	0.6490	0.6509	0.6060	0.6110	**0.6008**
	Improve	−6.00%	0.35%		5.63%	1.53%	1.24%	8.06%	7.30%	8.85%
	RMSE	0.8925	0.8748	0.9281	0.8584	0.9100	0.9118	0.7852	0.7927	**0.7772**
	Improve	3.84%	5.75%		7.51%	1.95%	1.76%	15.40%	14.59%	16.26%
40%	MAE	0.7052	0.6733	0.7092	0.6457	0.6933	0.7029	0.6186	0.6237	**0.6134**
	Improve	0.57%	5.07%		8.96%	2.24%	0.89%	12.77%	12.06%	13.51%
	RMSE	0.9085	0.9139	1.0107	0.9034	0.9842	0.9941	0.8023	0.8093	**0.7952**
	Improve	10.11%	9.57%		10.62%	2.62%	1.64%	20.62%	19.93%	21.32%
20%	MAE	0.7227	0.7124	0.8367	0.6973	0.8235	0.8302	0.6461	0.6509	**0.6417**
	Improve	13.63%	14.85%		16.66%	1.58%	0.78%	22.78%	22.21%	23.31%
	RMSE	0.9502	1.0006	1.2060	1.0037	1.1838	1.1963	0.8388	0.8446	**0.8335**
	Improve	21.21%	17.03%		16.78%	1.84%	0.80%	30.45%	29.97%	30.89%
Running time(s)		0.5157	0.5242	1096	1385	4529	7342	3168		

Table 5.6 Performance comparisons on Yelp (the baseline of improved performance is PMF)

Training	Metrics	UserMean	ItemMean	PMF	SoMF	HeteMF	HeteCF	SimMF-mean	SimMF-max	SimMF-min
80%	MAE	0.9664	0.8952	1.2201	0.8789	0.9307	1.2117	0.8292	0.8503	**0.8059**
	Improve	20.79%	26.63%		27.96%	23.72%	0.69%	32.04%	30.31%	33.95%
	RMSE	1.3443	1.2327	1.6479	1.1912	1.2773	1.6249	1.0577	1.0708	**1.0465**
	Improve	18.42%	25.20%		27.71%	22.49%	1.40%	35.82%	35.02%	36.49%
60%	MAE	0.9803	0.9247	1.3835	0.9156	0.9708	1.3510	0.8366	0.8615	**0.8109**
	Improve	29.14%	33.16%		33.82%	29.83%	2.35%	39.53%	37.73%	41.39%
	RMSE	1.3556	1.2893	1.8438	1.2591	1.3352	1.7940	1.0684	1.0842	**1.0532**
	Improve	26.48%	30.07%		31.71%	27.58%	2.70%	42.05%	41.20%	42.88%
40%	MAE	1.0219	0.9819	1.7081	0.9790	1.0409	1.6360	0.8509	0.8810	**0.8186**
	Improve	40.17%	42.52%		42.68%	39.06%	4.22%	50.18%	48.42%	52.18%
	RMSE	1.4241	1.3873	2.2123	1.3682	1.4343	2.1116	1.0863	1.1031	**1.0686**
	Improve	35.63%	37.29%		38.15%	35.17%	4.55%	50.90%	50.12%	51.70%
20%	MAE	1.1344	1.1202	2.6935	1.1252	1.8429	2.5782	0.8687	0.9047	**0.8290**
	Improve	57.88%	58.41%		58.23%	31.58%	4.28%	67.75%	66.41%	69.22%
	RMSE	1.5958	1.5981	3.2512	1.5907	2.3357	3.0807	1.1307	1.1733	**1.0944**
	Improve	50.92%	50.85%		51.07%	28.16%	5.24%	65.22%	63.91%	66.34%
Running time(s)		0.0646	0.0642	100	137	1963	2378	1414		



(a) Douban Movie, MAE (b) Douban Movie, RMSE

(c) Yelp, MAE (d) Yelp, RMSE

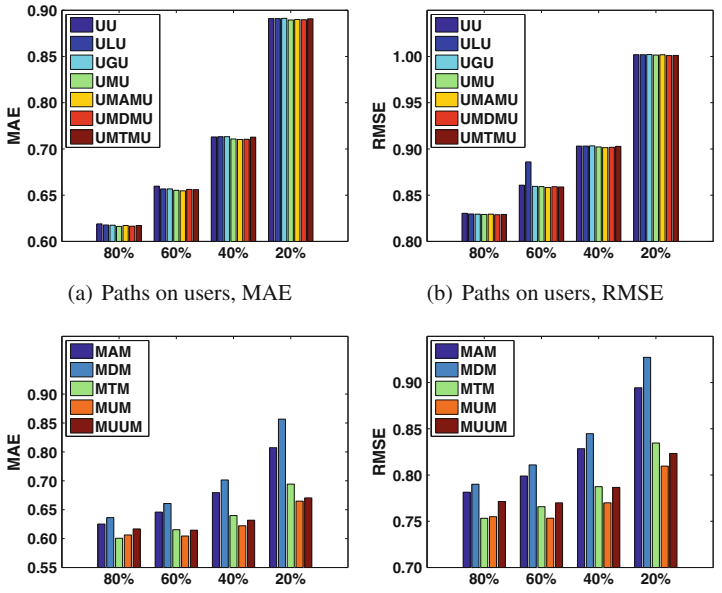
Fig. 5.7 Performance of SimMF with different regularizations on Douban Movie and Yelp datasets

We employ four variations of SimMF with average-based and individual-based regularization on users and items (i.e., SimMF with U(a)I(i), U(a)I(a), U(i)I(i), and U(i)I(a)) and four variations of SimMF with average-based or individual-based regularization on users or items (i.e., SimMF with U(a), U(i), I(a), and I(i)). The same parameters are set with above experiments, and the average results are shown in Fig. 5.7. We can find that SimMF, integrating similarity information on both users and items, always has better performance than the one only integrating similarity information on users or items. Again, we can observe the difference is far more pronounced when the fraction of training set is low; e.g., at 20%, SimMF-U(i) and SimMF-U(a) perform very bad. Moreover, we can also observe an interesting phenomena: Regularization models have different effects on users and items. SimMF-U(a) has better performance than SimMF-U(i) on both datasets, which indicates average-based regularization may be more suitable for users. However, it is not the case for items. SimMF-I(i) performs better than SimMF-I(a) on Douban Movie, while SimMF-I(a) outperforms SimMF-I(i) on Yelp. As a result, SimMF-U(a)I(i) has the best performance on Douban Movie, while SimMF-U(a)I(a) is the best one on Yelp. Although it is hard to draw general conclusions, the above study indicates that different regularization model may significantly affect performance of matrix factorization methods. In summary, we need to find the optimal regularization model according to data properties in real applications.

5.2.3.4 Impact of Different Meta Paths

In this section, we study the impact of different meta paths. Due to similar analysis, we only show results on Douban Movie dataset. As illustrated above, we employ 7 meta paths on users and 5 meta paths on movies. We will observe performance of SimMF with similarity matrix generated by one single meta path. Under the same parameters with above experiments, we run SimMF-U(a) with similarity matrix generated by each meta path on users. Similarly, we also run SimMF-I(i) with similarity matrix generated by each meta path on movies.

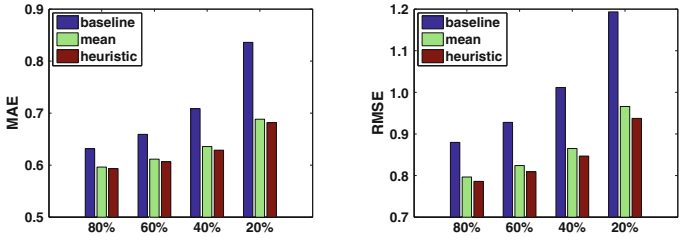
The experiment results on Douban Movie dataset are shown in Fig. 5.8. We can observe different impacts of meta paths on users and movies. The SimMF-U(a) with different meta paths (see Fig. 5.8a, b) on users all have close performance. Moreover, SimMF-U(a) with MUM has slightly better performance and SimMF-U(a) with UU has worse performance. However, it is not the case for meta paths on items. The SimMF-I(i) with different meta paths on items (see Fig. 5.8c, d) have totally different performance. We can find that SimMF-I(i) with MDM has the worst performance, even worse than PMF in some conditions, while SimMF-I(i) with MTM and MUM achieve much better performance on both criteria. We think there are two reasons: (1) Note that the performance of SimMF is much affected by the density of relations. The density of relations on MT and MU is much higher than that on MD and MA. The dense relations are helpful to generate good similarity of items. The similar



(a) Paths on users, MAE (b) Paths on users, RMSE

(c) Paths on movies, MAE (d) Paths on movies, RMSE

Fig. 5.8 Performance of SimMF with different meta paths on Douban Movie dataset



(a) MAE (b) RMSE

Fig. 5.9 Performance of SimMF on MAE and RMSE with different weights setting methods

phenomena have been widely observed in social recommendation [10, 13]. (2) The meaningful meta paths are helpful to reveal the similarity of objects. MTM means movies with same type, and MUM means movies seen by same users. These two paths are highly correlated as both reveal properties of the movies. These two reasons can also explain the slightly worse performance of the meaningful but sparse UU meta path as compared to other meta paths of users. The experiments imply that we only need to use one single dense and meaningful meta path to generate similarity information, which also can obtain good enough performance.

We further design an experiment to illustrate different importance of meta paths. Concretely, we observe the performance of above SimMF-I(i) with different weight combination methods on 5 meta paths. Except mean weight and random weight on 5 paths, we design a heuristic weight method, i.e., setting the weights according to the performance of these paths. That is, paths with good performance have higher weights. Assume the MAE performance value of a path (P_l) is P_l, and the max MAE value is P_{max}. Then, the difference is $d_l = e^{P_{max}-P_l}$. And thus, the weight of the path is $w_l^{\mathrm{I}} = \frac{d_l}{\sum_l d_l}$. The experiment also includes PMF as the baseline. The results are shown in Fig. 5.9. It is obvious that SimMF-I(i) with the heuristic weight method has the best performance, which further validates that the meaningful and dense meta paths are more important. The more detailed method description and experiment validation can be seen in [17].

5.3 Social Recommendation with Heterogeneous Information

5.3.1 Overview

With the boom of social media, social recommendation has become a hot research topic, which utilizes the social relations among users for better recommendation. Some researchers utilized trust information among users [10, 11], and some began to use friend relationship among users [13, 21] or other types of information

[1, 2]. Most of these social recommendation methods employ social regularization to confine similar users under the low-rank matrix factorization framework. Specifically, we can obtain the similarity of users from their social relations, and then, the social regularization, as a constraint term, confines the latent factors of similar users to be closer. It is reasonable, since similar users should have similar latent features.

However, the widely used social regularization in social recommendation has several shortcomings. (1) The similarity information of users is only generated from social relations of users. But we can obtain users' similarity from many ways in real applications, such as users' contents. (2) The social regularization only has constraint on users. In fact, we can also obtain the similarity of items and impose constraint on the latent factors of items. (3) The social regularization may be ineffective for dissimilar users, which may lead to dissimilar users having similar factors. The analysis and experiments in the next section validate this point.

In order to address the limitation of traditional social recommendation, we propose a dual similarity regularization-based recommendation method (called DSR). Inspired by the success of Heterogeneous Information Network (HIN) in many applications, we organize a recommended system as an HIN, which can integrate all kinds of information, including interactions between users and items, social relations among users, and attribute information of users and items. Based on the HIN, we can generate rich similarity information on both users and items by setting proper meta paths. Furthermore, we propose a new similarity regularization which can impose the constraint on users and items with high and low similarity. With the similarity regularization, DSR adopts a new optimization objective to integrate those similarity information of users and items. Then, we derive its solution to learn the weights of different similarities.

5.3.2 The DSR Method

In this section, we propose the dual similarity regularization-based matrix factorization method **DSR** and infer its learning algorithm.

5.3.2.1 Limitations of Social Recommendation

Recently, with the increasing popularity of social media, there is a surge of social recommendations which leverage rich social relations among users to improve recommendation performance. Ma et al. [13] first proposed the social regularization to extend low-rank matrix factorization, and then, it is widely used in a lot of work [9, 22]. A basic social recommendation method is illustrated as follows:

$$\min_{U,V} \mathcal{J} = \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{n} I_{ij}(R_{ij} - U_i V_j^T)^2 + \frac{\alpha}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} S_U(i,j)\|U_i - U_j\|^2 + \frac{\lambda_1}{2}(\|U\|^2 + \|V\|^2), \tag{5.28}$$

where $m \times n$ rating matrix R depicts users' ratings on n items and R_{ij} is the score user i gives to item j. I_{ij} is an indicator function which equals to 1 if user i rated item j and equals to 0 otherwise. $U \in \mathbb{R}^{m\times d}$ and $V \in \mathbb{R}^{n\times d}$, where $d \ll min(m,n)$ is the dimension number of latent factor. U_i is the latent vector of user i derived from the ith row of matrix U, while V_j is the latent vector of item j derived from the jth row of V. S_U is the similarity matrix of users, and $S_U(i,j)$ denotes the similarity of user i and user j. $\|\cdot\|^2$ is the Frobenius norm. Particularly, the second term is the social regularization which is defined as follows:

$$SocReg = \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} S_U(i,j)\|U_i - U_j\|^2. \tag{5.29}$$

As a constraint term in Eq. 5.28, $SocReg$ forces the latent factors of two users to be close when they are very similar. However, it may have two drawbacks.

- The similarity information may be simple. In social recommendation, the similarity information of users is usually generated from rating information or social relations, and only one type of similarity information is employed. However, in many applications, we can obtain much more rich similarity information of users and items from various ways, such as rich attribute information and interactions. We need to make full use of these similarity information of users and items for recommendation.
- The constraint term may not work well when two users are not very similar. The minimization of optimization objective should force the latent factors of two users with high similarity to be close. However, when two users are not similar (i.e., $S_U(i,j)$ is small), $SocReg$ may still force the latent factors of these two users to be close. However, these two users should be dissimilar which means their latent factors should have large distance.

In order to uncover the limitations of social regularization, we apply the model detailed in Eq. 5.28 to conduct four experiments each with different levels of similarity information ($None$, Low, $High$, All). $None$ denotes that we utilize no similarity information in the model (i.e., $\alpha = 0$ in the model), Low denotes that we utilize bottom 20% users' similarity information generated in the model, $High$ is that of top 20%, and All denotes we utilize all users' similarity information. The Douban dataset is employed in the experiments, and we report MAE and RMSE in Fig. 5.10. The results of Low, $High$, and All are better than that of $None$, which implies social regularization really works in the model. However, in terms of performance improvement compared to $None$, Low does not improve as much as $High$ and All

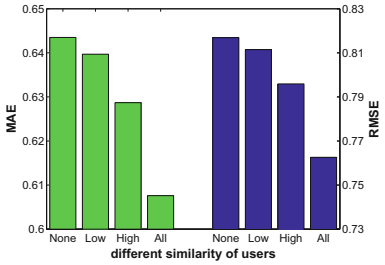


Fig. 5.10 Limitations of social regularization

do. The above analysis reveals that the social regularization may not work well in recommender models when users are with low similarity.

5.3.2.2 Rich Similarity Generation

Traditional social recommendations only consider the constraint of users with their social relations. However, rich similarity information on users and items can be generated in a heterogeneous information network. Two types of objects in an HIN can be connected via various meta paths [19], which is a composite relation connecting these two types of objects. Therefore, we can evaluate the similarity of users (or movies) based on different meta paths. For example, for users, we can consider UU, UGU, UMU, etc. Similarly, meaningful meta paths connecting movies include MAM and MDM.

Several path-based similarity measures have been proposed to evaluate the similarity of objects under given meta path in HIN [16, 19]. We assume that $S_U^{(p)}$ denotes similarity matrix of users under meta path $P_U^{(p)}$ connecting users, and $S_U^{(p)}(i, j)$ denotes the similarity of users i and j under the path $P_U^{(p)}$. Similarly, $S_I^{(q)}$ denotes similarity matrix of items under the path $P_I^{(q)}$ connecting items, and $S_I^{(q)}(i, j)$ denotes the similarity of items i and j. Since users (or items) have different similarities under various meta paths, we combine their similarities on all paths through assigning weights on these paths. For users and items, we define S_U and S_I to represent the similarity matrix of users and items on all meta paths, respectively.

$$S_U = \sum_{p=1}^{|P_U|} \boldsymbol{w}_U^{(p)} S_U^{(p)}, \tag{5.30}$$

$$S_I = \sum_{q=1}^{|P_I|} \boldsymbol{w}_I^{(q)} S_I^{(q)}, \tag{5.31}$$

where $\boldsymbol{w}_U^{(p)}$ denotes the weight of meta path $P_U^{(p)}$ among all meta paths P_U connecting users, and $\boldsymbol{w}_I^{(q)}$ denotes the weight of meta path $P_I^{(q)}$ among all meta paths P_I connecting items.

5.3.2.3 Similarity Regularization

Due to the limitations of widely used social regularization, we design a new similarity regularization to constraining users and items simultaneously with much available similarity information of users and items. The basic idea of similarity regularization is that the distance of latent factors of two users (or items) should be negatively correlated to their similarity, which means two similar users (or items) should have a short distance while two dissimilar ones should have a long distance with their latent factors. We note that the Gaussian function meets above requirement. Moreover, the range of Gaussian function is [0,1], same with the range of similarity function. Following this idea, we design a similarity regularization on users as follows:

$$SimReg^{\mathrm{U}} = \frac{1}{8}\sum_{i=1}^{m}\sum_{j=1}^{m}(S_U(i,j) - e^{-\gamma\|U_i - U_j\|^2})^2, \tag{5.32}$$

where γ controls the radial intensity of Gaussian function and the coefficient $\frac{1}{8}$ is convenient for deriving the learning algorithm. This similarity regularization can enforce constraint on both similar and dissimilar users. In addition, the similarity matrix S_U can be generated from social relations or the above HIN. Similarly, we can also design the similarity regularization on items as follows:

$$SimReg^{\mathrm{I}} = \frac{1}{8}\sum_{i=1}^{n}\sum_{j=1}^{n}(S_I(i,j) - e^{-\gamma\|V_i - V_j\|^2})^2, \tag{5.33}$$

The Proposed DSR Model We propose the **D**ual **S**imilarity regularization for **R**ecommendation (called DSR) through adding the similarity regularization on users and items into low-rank matrix factorization framework. Specifically, the optimization model is proposed as follows:

$$\begin{aligned}\min_{U,V,\boldsymbol{w}_U,\boldsymbol{w}_I} \mathrm{J} = & \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{n} I_{ij}(R_{ij} - U_i V_j^T)^2 \\ & + \frac{\lambda_1}{2}(\|U\|^2 + \|V\|^2) + \frac{\lambda_2}{2}(\|\boldsymbol{w}_U\|^2 + \|\boldsymbol{w}_I\|^2) \\ & + \alpha SimReg^{\mathrm{U}} + \beta SimReg^{\mathrm{I}} \\ s.t. \quad & \sum_{p=1}^{|P_U|} \boldsymbol{w}_U^{(p)} = 1, \boldsymbol{w}_U^{(p)} \geq 0\end{aligned} \tag{5.34}$$

$$\sum_{q=1}^{|P_I|} \boldsymbol{w}_I^{(q)} = 1, \boldsymbol{w}_I^{(q)} \geq 0,$$

where α and β control the ratio of similarity regularization term on users and items, respectively.

5.3.2.4 The Learning Algorithm

The learning algorithm of DSR can be divided into two steps. (1) Optimize the latent factor matrices of users and items (i.e., U, V) with the fixed weight vectors $\boldsymbol{w}_U = [\boldsymbol{w}_U^{(1)}, \boldsymbol{w}_U^{(2)}, \cdots, \boldsymbol{w}_U^{(|P_U|)}]^T$ and $\boldsymbol{w}_I = [\boldsymbol{w}_I^{(1)}, \boldsymbol{w}_I^{(2)}, \cdots, \boldsymbol{w}_I^{(|P_I|)}]^T$. (2) Optimize the weight vectors $\boldsymbol{w}_U$ and $\boldsymbol{w}_I$ with the fixed latent factor matrices U and V. Through iteratively optimizing these two steps, we can obtain the optimal U, V, $\boldsymbol{w}_U$, and $\boldsymbol{w}_I$.

Optimize U and V With the fixed $\boldsymbol{w}_U$ and $\boldsymbol{w}_I$, we can optimize U and V by performing stochastic gradient descent.

$$\begin{aligned}\frac{\partial \mathcal{J}}{\partial U_i} = &\sum_{j=1}^{n} I_{ij}(U_i V_j^T - R_{ij})V_j \\ &+\alpha \sum_{j=1}^{m} \gamma[(S_U(i,j) - e^{-\gamma\|U_i - U_j\|^2})e^{-\gamma\|U_i - U_j\|^2}(U_i - U_j)] \\ &+\lambda_1 U_i,\end{aligned} \tag{5.35}$$

$$\begin{aligned}\frac{\partial \mathcal{J}}{\partial V_j} = &\sum_{i=1}^{m} I_{ij}(U_i V_j^T - R_{ij})U_i \\ &+\beta \sum_{i=1}^{n} \gamma[(S_I(i,j) - e^{-\gamma\|V_i - V_j\|^2})e^{-\gamma\|V_i - V_j\|^2}(V_i - V_j)] \\ &+\lambda_1 V_j,\end{aligned} \tag{5.36}$$

Optimize $\boldsymbol{w}_U$ and $\boldsymbol{w}_I$ With the fixed U and V, the minimization of $\mathcal{J}$ with respect to $\boldsymbol{w}_U$ and $\boldsymbol{w}_I$ is a well-studied quadratic optimization problem with nonnegative bound. We can use the standard trust region reflective algorithm to update $\boldsymbol{w}_U$ and $\boldsymbol{w}_I$ at each iteration. We can simplify the optimization function of $\boldsymbol{w}_U$ as the following standard quadratic formula:

$$\min_{\boldsymbol{w}_U} \ \frac{1}{2}\boldsymbol{w}_U^T H_U \boldsymbol{w}_U + f_U^T \boldsymbol{w}_U \qquad (5.37)$$
$$s.t. \ \sum_{p=1}^{|P_U|} \boldsymbol{w}_U^{(p)} = 1, \boldsymbol{w}_U^{(p)} \geq 0.$$

Here, H_U is a $|P_U| \times |P_U|$ symmetric matrix as follows:

$$H_U(i,j) = \begin{cases} \frac{\alpha}{4}(\sum\sum S_U^{(i)} \odot S_U^{(j)}) & i \neq j, \ 1 \leq i,j \leq |P_U| \\ \frac{\alpha}{4}(\sum\sum S_U^{(i)} \odot S_U^{(j)}) + \lambda_2 & i = j, \ 1 \leq i,j \leq |P_U|, \end{cases}$$

$\odot$ denotes the dot product. f_U is a column vector with length $|P_U|$, which is calculated as follows:

$$f_U(p) = -\frac{\alpha}{4}\sum_{i=1}^{m}\sum_{j=1}^{m} S_U^{(p)}(i,j)e^{-\gamma\|U_i - U_j\|^2}.$$

Similarly, we can also infer the optimization function of $\boldsymbol{w}_I$.

5.3.3 Experiments

In this section, we conduct experiments to validate the effectiveness of DSR and further explore the cold-start problem.

5.3.3.1 Experiment Settings

We use two real datasets: Douban and Yelp in experiments. Note that the Douban dataset has sparse social relationship with dense rating information, while the Yelp dataset has dense social relationships with sparse rating information. We still use Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) to evaluate the performance of rating prediction.

In order to validate the effectiveness of DSR, we compare it with following representative methods. Besides the classical social recommendation method SoMF, the experiments also include two recent HIN-based methods, HeteCF and HeteMF. In addition, in order to validate the effectiveness of similarity regularization, we include the revised version of SoMF with similarity regularization (i.e., SoMF_{SR}).

- **UserMean**. It employs a user's mean rating to predict the missing ratings directly.
- **ItemMean**. It employs an item's mean rating to predict the missing ratings directly.
- **PMF** [14]. Salakhutdinov and Minh proposed the basic low-rank matrix factorization method for recommendation.

- **SoMF** [13]. Ma et al. proposed the social recommendation method with social regularization on users.
- **HeteCF** [9]. Luo et al. proposed the social collaborative filtering algorithm using heterogeneous relations.
- **HeteMF** [22]. Yu et al. proposed the HIN-based recommendation method through combining user ratings and items' similarity matrices.
- **SoMF$_{\mathbf{SR}}$**. It adapts SoMF through only replacing the social regularization with the similarity regularization $SimReg^{\mathrm{U}}$.

For Douban dataset, we utilize 7 meta paths for user (i.e., UU, UGU, ULU, UMU, UMDMU, UMTMU, and UMAMU) and 5 meta paths for item (i.e., MTM, MDM, MAM, MUM, and MUUM). For Yelp dataset, we utilize 2 meta paths for user (i.e., UB and UU) and 2 meta paths for item (i.e., BC and BL). HeteSim [16] is employed to evaluate the object similarity based on above meta paths. These similarity matrices are fairly utilized for HeteCF, HeteMF, and DSR. We set $\gamma = 1$, $\alpha = 10$, and $\beta = 10$ through parameter experiments on Douban dataset. In the experiments on Yelp dataset, we set the parameters $\gamma = 1$, $\alpha = 10$, and $\beta = 10$. Meanwhile, optimal parameters are set for other models in the experiments.

5.3.3.2 Effectiveness Experiments

For Douban dataset, we use different ratios (80%, 60%, 40%) of data as training sets and the rest of the dataset for testing. Considering the sparse density of Yelp dataset, we use 90%, 80%, and 70% of data as training sets and the rest of the dataset for testing for Yelp dataset. The random selection is carried out 10 times independently, and we report the average results in Table 5.7.

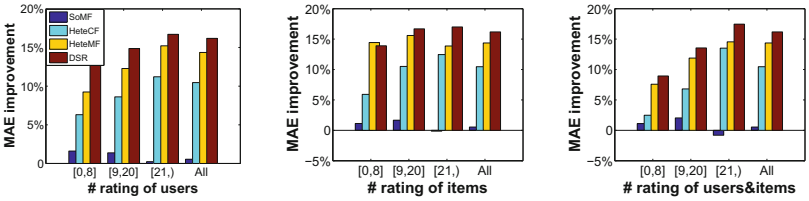
It is clear that three HIN-based methods (DSR, HeteCF, and HeteMF) all achieve significant performance improvements compared to PMF, UserMean, ItemMean, and SoMF. It implies that integrating heterogeneous information is a promising way to improve recommendation performance. Particularly, DSR always has the best performance on all conditions compared to other methods. It indicates that the dual similarity regularization on users and items may be more effective than traditional social regularization. It can be further confirmed by the better performance of SoMF$_{\mathrm{SR}}$ over SoMF. Although the superiority of SoMF$_{\mathrm{SR}}$ over SoMF is not significant, the improvement is achieved on the very weak social relations in Douban dataset. In addition, we can also find that DSR has better performance improvement for less training data. It reveals that DSR has the potential to alleviate the cold-start problem.

5.3.3.3 Study on Cold-Start Problem

Furthermore, we validate the superiority of DSR on cold-start problem. We run PMF, SoMF, HeteCF, HeteMF, and DSR on Douban dataset with 40% training ratio. We set four levels of users: three types of cold-start users with various numbers

Table 5.7 Effectiveness experimental results on Douban and Yelp (The improvement is based on PMF)

Dataset	Training	Metrics	PMF	UserMean	ItemMean	SoMF	HeteCF	HeteMF	SoMF_{SR}	DSR
Douban	80%	MAE	0.6444	0.6954	0.6284	0.6396	0.6101	0.5941	0.6336	**0.5856**
		Improve		−7.92%	2.47%	0.73%	5.32%	7.79%	1.68%	9.12%
		RMSE	0.8151	0.8658	0.7928	0.8098	0.7657	0.7520	0.8000	**0.7379**
		Improve		−6.23%	2.73%	0.64%	6.05%	7.73%	1.85%	9.46%
	60%	MAE	0.6780	0.6967	0.6370	0.6696	0.6317	0.6056	0.6648	**0.5946**
		Improve		−2.76%	6.05%	1.25%	6.84%	10.68%	1.96%	12.31%
		RMSE	0.8569	0.8687	0.8135	0.8445	0.7901	0.7665	0.8358	**0.7483**
		Improve		−1.37%	5.07%	1.45%	7.80%	10.56%	2.46%	12.68%
	40%	MAE	0.7364	0.7009	0.6629	0.7245	0.6762	0.6255	0.7141	**0.6092**
		Improve		4.83%	9.99%	1.63%	8.18%	15.07%	3.03%	17.28%
		RMSE	0.9221	0.8747	0.8747	0.9058	0.8404	0.7891	0.8950	**0.7629**
		Improve		5.14%	5.13%	1.76%	8.86%	14.42%	2.94%	17.27%
Yelp	90%	MAE	0.8475	0.9543	0.8822	0.8460	0.8461	0.8960	0.8459	**0.8158**
		Improve		−12.60%	−4.09%	0.18%	0.17%	−5.72%	0.18%	3.74%
		RMSE	1.0796	1.3138	1.2106	1.0772	1.0773	1.1272	1.0772	**1.0369**
		Improve		−21.69%	−12.13%	0.22%	0.21%	−4.41%	0.22%	3.95%
	80%	MAE	0.8528	0.9621	0.8931	0.8527	0.8528	0.8907	0.8526	**0.8206**
		Improve		−12.82%	−4.72%	0.01%	0.00%	−4.44%	0.01%	3.78%
		RMSE	1.0850	1.3255	1.2304	1.0849	1.0850	1.1195	1.0848	**1.0413**
		Improve		−22.17%	−13.40%	0.01%	0.00%	−3.18%	0.02%	4.03%
	70%	MAE	0.8576	0.9706	0.9062	0.8575	0.8576	0.8976	0.8575	**0.8250**
		Improve		−13.17%	−5.67%	0.01%	0.00%	−4.66%	0.01%	3.80%
		RMSE	1.0894	1.3395	1.2547	1.0936	1.0894	1.1313	1.0894	**1.0461**
		Improve		−22.96%	−15.17%	−0.39%	0.00%	−3.85%	0.00%	3.97%



(a) Users_MAE (b) Items_MAE (c) Users&Items_MAE

Fig. 5.11 MAE improvement against PMF on various cold-start levels

of rated movies (e.g., [0, 8] denotes users rated no more than 8 movies and "All" means all users in Fig. 5.11). We conduct similar experiments on cold-start items and users&items (users and items are both cold-start). The experiments are shown in Fig. 5.11. Once again, we find that 3 HIN-based methods all are effective for cold-start users and items. Moreover, DSR always has the highest MAE improvement on almost all conditions, due to dual similarity regularization on users and items. It is reasonable since the DSR method takes much constraint information of users and items into account which would play a crucial role when there is little available information of users or items. The more detailed method description and experiment validation can be seen in [3, 26].

5.4 Conclusions

In recent years, recommendation has become a very popular application to alleviate information overload, and many recommendation techniques have been proposed. Recommender system includes a lot of object types and the rich relations among object types, so we can naturally constitute a heterogeneous information network from recommended system. The comprehensive information integration and rich semantic information of HIN make it promising to generate better recommendation. In this chapter, we introduce two types of recommendation methods with HIN. One type of methods employ the semantic path-based similarity measure to recommend items directly, and the other type of methods utilize rich similarity generated by meta paths to extend traditional matrix factorization methods. Experiments not only validate the effectiveness of these proposed methods but also show the benefits of information integration with heterogeneous network. In the future work, we can exploit the power and benefits of information integration with heterogeneous network in more applications.

References

1. BellogíN, R., Cantador, I., Castells, P.: A comparative study of heterogeneous item recommendations in social systems. Inf. Sci. **221**, 142–169 (2013)
2. Cantador, I., Bellogin, A., Vallet, D.: Content-based recommendation in social tagging systems. In: RecSys, pp. 237–240 (2010)
3. Cao, X., Zheng, Y., Shi, C., Li, J., Wu, B.: Link prediction in Schema-Rich heterogeneous information network. In: PAKDD, pp. 449–460 (2016)
4. Jamali, M., Ester, M.: Trustwalker: a random walk model for combining trust-based and item-based recommendation. In: KDD, pp. 397–406 (2009)
5. Jamali, M., Lakshmanan, L.V.S.: HeteroMF: recommendation in heterogeneous information networks using context dependent factor models. In: WWW, pp. 643–654 (2013)
6. Lao, N., Cohen, W.: Fast query execution for retrieval models based on path constrained random walks. In: KDD, pp. 881–888 (2010)
7. Lin, C.J.: Projected gradient methods for non-negative matrix factorization. In: Neural Computation, pp. 2279–2756 (2007)
8. Lippert, C., Weber, S.H., Huang, Y., Tresp, V., Schubert, M., Kriegel, H.P.: Relation prediction in multi-relational domains using matrix factorization. In: NIPS Workshop on Structured Input Structure Output (2008)
9. Luo, C., Pang, W., Wang, Z.: Hete-CF: social-based collaborative filtering recommendation using heterogeneous relations. In: ICDM, pp. 917–922 (2014)
10. Ma, H., Yang, H., Lyu, M.R., King, I.: SoRec: social recommendation using probabilistic matrix factorization. In: CIKM, pp. 931–940 (2008)
11. Ma, H., King, I., Lyu, M.R.: Learning to recommend with social trust ensemble. In: SIGIR, pp. 203–210 (2009)
12. Ma, H., Zhou, T.C., Lyu, M.R., King, I.: Improving recommender systems by incorporating social contextual information. ACM Trans. Inf. Syst. **29**(2), 9 (2011)
13. Ma, H., Zhou, D., Liu, C., Lyu, M.R., King, I.: Recommender systems with social regularization. In: WSDM, pp. 287–296 (2011)
14. Salakhutdinov, R., Mnih, A., Salakhutdinov, R., Mnih, A.: Probabilistic matrix factorization. In: NIPS **20**, 1257–1264 (2008)
15. Shardanand, U., Maes, P.: Social information filtering: algorithms for automating word of mouth. In: Sigchi Conference on Human Factors in Computing Systems, pp. 210–217 (1995)
16. Shi, C., Kong, X., Huang, Y., Philip, S.Y., Wu, B.: Hetesim: a general framework for relevance measure in heterogeneous networks. IEEE Trans. Knowl. Data Eng. **26**(10), 2479–2492 (2014)
17. Shi, C., Liu, J., Zhuang, F., Yu, P.S., Wu, B.: Integrating heterogeneous information via flexible regularization framework for recommendation. Knowl. Inf. Syst. **49**(3), 1–25 (2015)
18. Srebro, N., Jaakkola, T.: Weighted low-rank approximations. In: ICML, pp. 720–727 (2003)
19. Sun, Y.Z., Han, J.W., Yan, X.F., Yu, P.S., Wu, T.: PathSim: meta path-based top-K similarity search in heterogeneous information networks. In: VLDB, pp. 992–1003 (2011)
20. Sun, Y., Han, J.: Mining heterogeneous information networks: a structural analysis approach. SIGKDD Explor. **14**(2), 20–28 (2012)
21. Yang, X., Steck, H., Liu, Y.: Circle-based recommendation in online social networks. In: KDD, pp. 1267–1275 (2012)
22. Yu, X., Ren, X., Gu, Q., Sun, Y., Han, J.: Collaborative filtering with entity similarity regularization in heterogeneous information networks. In: IJCAI-HINA Workshop (2013)
23. Yu, X., Ren, X., Sun, Y., Sturt, B., Khandelwal, U., Gu, Q., Norick, B., Han, J.: Recommendation in heterogeneous information networks with implicit user feedback. In: RecSys, pp. 347–350 (2013)
24. Yu, X., Ren, X., Sun, Y., Gu, Q., Sturt, B., Khandelwal, U., Norick, B., Han, J.: Personalized entity recommendation: a heterogeneous information network approach. In: WSDM, pp. 283–292 (2014)
25. Yuan, Q., Chen, L., Zhao, S.: Factorization vs. regularization: fusing heterogeneous social relationships in top-n recommendation. In: RecSys, pp. 245–252 (2011)

26. Zheng, J., Liu, J., Shi, C., Zhuang, F., Li, J., Wu, B.: Dual similarity regularization for recommendation. In: PAKDD, pp. 542–554 (2016)