Heterogeneous Neural Attentive Factorization Machine for Rating Prediction

Liang Chen
School of data and computer science, Sun Yat-Sen University, China
chenliang6@mail.sysu.edu.cn

Yang Liu
School of data and computer science, Sun Yat-Sen University, China
liuy296@mail2.sysu.edu.cn

Zibin Zheng
School of data and computer science, Sun Yat-Sen University, China
zhzibin@mail.sysu.edu.cn

Philip S Yu
Department of Computer Science, University of Illinois at Chicago, USA
psyu@cs.uic.edu

ABSTRACT

Heterogeneous Information Network(HIN) has been employed in recommender system to represent heterogeneous types of data, and meta path has been proposed to capture semantic relationship among objects. When applying HIN to the recommendation, there are two problems: how to extract features from meta paths and how to properly fuse these features to further improve recommendations. Some recent work has employed deep neural network to learn user and item representation, and attention mechanism has been explored to integrate information for recommendation. Inspired by these work, in this paper, we propose Heterogeneous Neural Attentive Factorization Machine(HNAFM) to solve above problems. Specifically, we first calculate the commuting matrices based on meta paths and use multilayer perceptrons to learn user and item features. A hierarchical attention mechanism is employed to find the meta path that best describes user's preference and item's property. Comprehensive experiments based on real-world datasets demonstrate that the proposed HNAFM significantly outperforms state-of-the-art rating prediction methods.

KEYWORDS

heterogeneous information network, recommendation, neural network, attention mechanism, meta path

ACM Reference Format:
Liang Chen, Yang Liu, Zibin Zheng, and Philip S Yu. 2018. Heterogeneous Neural Attentive Factorization Machine for Rating Prediction. In 2018 ACM Conference on Information and Knowledge Management (CIKM'18), October 22-26, 2018, Torino, Italy, ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3269206.3271759

CIKM '18, October 22–26, 2018, Torino, Italy

ACM ISBN 978-1-4503-6014-2/18/10...$15.00.
https://doi.org/10.1145/3269206.3271759

1 INTRODUCTION

In the information explosion era, many online services including E-commerce, question answering community and online news have adopted recommender systems to capture user's preference. Traditional latent factor based models[14, 18] make predictions by factorizing the user-item interaction matrix into two low-rank user and item factor vectors. Recently, deep neural networks are employed in recommendation tasks to learn user-item interactions in a non-linear way. NeuMF[9] replaces the inner product in latent factor model with a neural architecture to model the user-item latent structures. DMF[28] maps users and items in a common low dimensional space via a deep neural network. While above models perform well, they only involve interactions of two types of entities, i.e., user and item. More types of objects and their relations could be utilized into recommendation tasks to generate better results.

Heterogeneous information network(HIN) has been proposed to represent different types of data, where meta path[23] is utilized to describe complex semantic between two HIN entities. Figure 1 displays a toy example of HIN. Under this assumption, in place of the latent factor model, we can define a simple meta path *User* → *Movie* and learn features from this meta path. More complex meta path can be built to express complicated semantic like *User* → *Movie* → *Type* ← *Movie* ← *User*. This meta path reveals user preference to some degree through those who watch the same type of movies. Thus, Constructing different meta paths for recommendation can effectively integrate more information and fuse richer semantic information. Recently, some works[22, 30, 32] have paid attention to the benefits of HIN for recommendation and tried to incorporate meta paths in recommendation models. However, the realization of this idea faces the following challenges.

First, **how to effectively extract features from meta paths**. There have been some works about this topic. PathSim[23] has been proposed to measure the similarity between objects. Meta path instances are counted and then divided by a normalization term, resulting in a similarity matrix between the starting and ending nodes. HetaRec[30] and SemRec[22] obtain the similarity matrix following the same pattern and feed the matrix to the model directly. Furthermore, FMG[32] factorizes the similarity matrix into user and item feature vectors. Despite the effectiveness of such methods, they all model the features in a linear way, which is insufficient

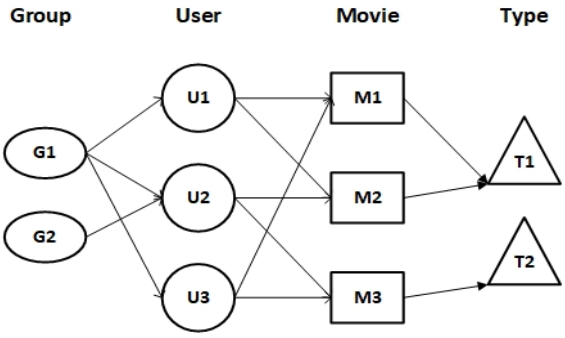

Figure 1: A toy example of HIN

for capturing the non-linear and complex inherent structure of real-world data.

Second, **how to properly fuse features extracted from different meta paths**. Up to now, there are two kinds of solutions. One way is to build the model based on single meta path, and then use a weight learning methods to assemble all the results. This solution fail to utilize total information for training and probably suffer data sparsity problem. The other way uses all information of meta paths for training, such as FMG[32]. FMG uses matrix factorization to obtain the latent features of users and items. After that, FMG concatenate all of the corresponding user and item features from the paths for further training. However, since working with many meta paths may bring noises, it requires efficient filtering technique. Moreover, this method adopts an independent model to learn the features, which means these features do not directly contribute to the final objective. Therefore, it may lead to a suboptimal solution.

To address the above two challenges, we propose a novel neural framework named Heterogeneous Neural Attentive Factorization Machine (HNAFM). First, instead of factorizing the similarity matrix, we take the row and column of the matrix as the raw features of user and item respectively, and feed them into multilayer perceptrons to get user and item latent vectors. Second, inspired by recent work of Attentive Collaborative Filtering(ACF)[3], which introduces a novel attention mechanism to address the item- and component-level implicit feedback in multimedia recommendation, we design a hierarchical attention layer to fuse the set of user and item features learned from meta paths. Specifically, we first employ a user-level and item-level attention layer to combine user and item features respectively, and then adopt a user-item level attention layer to merge all the features. In this way, useful meta paths could be selected automatically. Finally, for the purpose of user-item rating prediction, factorization machine[19, 20] is applied to compute final results.

Overall, the main contributions of our work could be summarized as follows:

(1) We propose a novel HIN-based recommendation model named Heterogeneous Neural Attentive Factorization Machine, in which two levels of attention mechanisms are employed to model the importance of user and item latent vectors generated by different meta-paths. To the best of our knowledge, HNAFM is the first HIN-based recommendation model integrating attention mechanism.

(2) We propose to use the row and column vectors of commuting matrices as the raw features of users and items, and utilize multi-layer perceptron to map user and item features of all meta-paths into a common low-dimensional space with non-linear projections.

(3) Extensive experiments conducted on two real-world datasets demonstrate that our approach successfully outperforms state-of-the-art methods in the task of rating prediction.

The rest of this paper is organized as follows: Section 2 highlights the related work of heterogeneous information network in recommendation, deep learning in recommendation and attention mechanism, respectively. Section 3 introduces the definition, architecture, and details of our proposed HNAFM model. Section 4 shows the experimental results. Finally Section 5 concludes this paper and introduces the future work.

2 RELATED WORK

In this section, we highlight some related works to this paper, including the works of heterogeneous information network in recommendation, deep learning in recommendation, and attention mechanism, respectively.

2.1 HIN in Recommendation

Due to the ability to represent heterogeneous types of data, many data mining tasks have been exploited in HINs, such as similarity measure[11, 15, 21, 23], classification[10, 13, 17] and representation learning[2, 5, 6]. Recently, HIN-based recommendation[22, 26, 30, 32] works has shown their efficiency in utilizing auxiliary information for recommendation. HeteRec[30] first generates user and item latent features based on multiple meta paths and then use a weight ensemble methods to recover the similarity matrices. SemRec[22] builds recommendation model on weighted HIN. They calculate user similarity matrices on extended meta paths and the combination of different meta paths are used to predict the final results. FMG[32] is proposed to solve meta path fusion and data sparsity problem with a "matrix factorization + factorization machine" approach. Different from previous work, HNAFM uses the deep neural network to map users and items in a common latent space to solve data sparsity problem and better model real-world data. After that, a two layer attention mechanism is employed to fuse all the information for training, finally factorization machine is adopted to predict the rates, From Table 3, we can see that HNAFM is more powerful.

2.2 Deep Learning in Recommendation

In recent years, some work[4, 8, 9, 24, 25, 31] started to explore deep neural network for recommendation. Some work[24, 25, 31] primarily uses DNNs for processing auxiliary information such as text, music and image. NCF[9] replaces the inner product with a neural architecture to learn interactions between users and items. Cheng et al. proposes Wide & Deep[4] for app recommendation, where the wide part is a generalized linear model and deep part is a multilayer

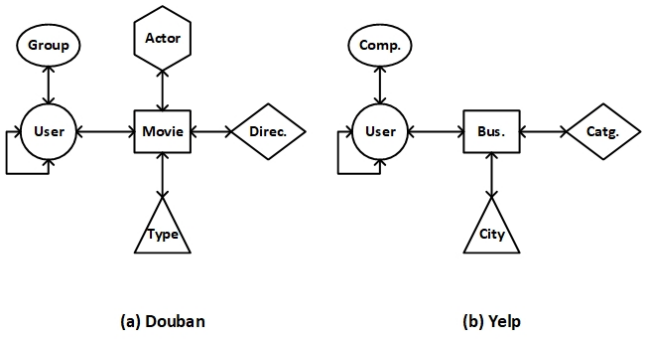

Figure 2: Network schema of Douban and Yelp datasets

perceptron on the concatenation of feature embedding vectors. He and Chua proposed Neural Factorization Machine[8] which using the deep neural network to model high-order feature interactions. Previous HIN-based recommendation models are linear, which are insufficient to learn real-world data. We introduce deep learning methods in HIN-based recommendation. With the help of DNNs, features based on meta path can be learned effectively.

2.3 Attention Mechanism

Attention mechanism is developed based on the assumption that people tend to focus on the selective part of context information. It has been shown effective in multiple machine learning tasks such as image captioning[27, 29] and machine translation[1]. Some researchers have noticed the benefits of attention mechanism and employed it in the area of recommender systems. Chen et al. proposed Attentive Collaborative Filtering(ACF)[3] to address the item- and component-level implicit feedback in multimedia recommendation. In [16], attention mechanism is adopted to model the user sequential behavior and capture users' main purpose in a session based recommendation scenario.

In our proposed HNAFM model, attention mechanism is proposed to combine the meta path features. More precisely, a two-layer attention model is used to address the meta path that best describes the user and item. To the best of our knowledge, HNAFM is the first HIN-based recommendation model integrating attention mechanism.

3 HNAFM FRAMEWORK

In this section, we first introduce the definition of HIN, and then provide an overview of HNAFM model and the details, respectively.

3.1 Heterogeneous Information Network

Heterogeneous information network(HIN) is the connerstone of the HNAFM framework. It is crucial to form a HIN given a dataset. We start from the definition of heterogeneous information network. In this paper, we use the definition from [23].

Definition 3.1. **A Heterogeneous Information Network** is defined as a graph $G = (V, E)$ with an object type mapping function $\phi : V \rightarrow \mathcal{A}$ and a link type mapping function $\varphi : E \rightarrow \mathcal{R}$, where each object $v \in V$ belongs to one particular object type $\phi(v) \in \mathcal{A}$ and each link $e \in E$ belongs to a particular relation $\varphi(e) \in \mathcal{R}$, when the types of objects $|\mathcal{A}| > 1$ or the types of relations $|\mathcal{R}| > 1$.

Figure 1 shows a toy example of HIN whose has four object types and three relationship types. Given a HIN, it is necessary to provide a description of the whole network. Network schema is proposed to give a meta level description of a network. In this paper, the definition of network schema[23] is as follows.

Definition 3.2. **The Network Schema** is a meta template for a heterogeneous network $G = (V, E)$ with the object type mapping function $\phi : V \rightarrow \mathcal{A}$ and the link type mapping function $\varphi : E \rightarrow \mathcal{R}$, which is a directed graph defined over object types $\mathcal{A}$, with edges as relations from $\mathcal{R}$, denoted as $T_G = (\mathcal{A}, \mathcal{R})$.

Figure 2 shows the network schemas of experiment datasets. Douban dataset has six object types and six relation types. Yelp dataset has five object types and five relation types. In a HIN, two objects can be connected via different paths. For example, in Douban, two users can be connected via $User \rightarrow Movie \leftarrow User$ or $User \rightarrow Group \leftarrow User$. These paths are called meta paths, defined as follows.

Definition 3.3. **A Meta Path** $\mathcal{P}$ is a path defined on the graph of network schema $T_G = (\mathcal{A}, \mathcal{R})$, and is denoted in the form of $A_1 \xrightarrow{R_1} A_2 \xrightarrow{R_2} \ldots \xrightarrow{R_l} A_{l+1}$, which defines a composite relation $R = R_1 \circ R_2 \circ \ldots \circ R_l$ between type A_1 and A_{l+1}, where $\circ$ denotes the composition operator on relations.

The meta paths used in experiments are shown in Table 1. Intuitively, these meta paths contain different semantic information. For instance, The first meta path in Yelp dataset, i.e., U → U ← B, represents the business user's friends consumed, which can be seemed as a social recommendation model. The third meta path in Yelp dataset, i.e., U → B ← U → B, represents the business that user who consumed the same business consumed, which can be seemed as a collaborative filtering model. Thus, it is possible to generate better recommendation by fusing the information learned from multiple meta paths.

Table 1: Meta paths used in experiments

Datasets	Meta Path
Douban	1: U → G ← U → M
	2: U → M ← U → M
	3: U → M → D ← M ← U → M
	4: U → M → A ← M ← U → M
	5: U → M → T ← M ← U → M
Yelp	1: U → U ← B
	2: U → Co ← U → B
	3: U → B ← U → B
	4: U → B → Ca ← B ← U → B
	5: U → B → Ci ← B ← U → B

collab. filtering

3.2 The HNAFM Model

Figure 3 illustrates the architecture of HNAFM model, which has three main steps. First, according to [8], using an informative operation in a low-level can ease the burden of higher-level layers for

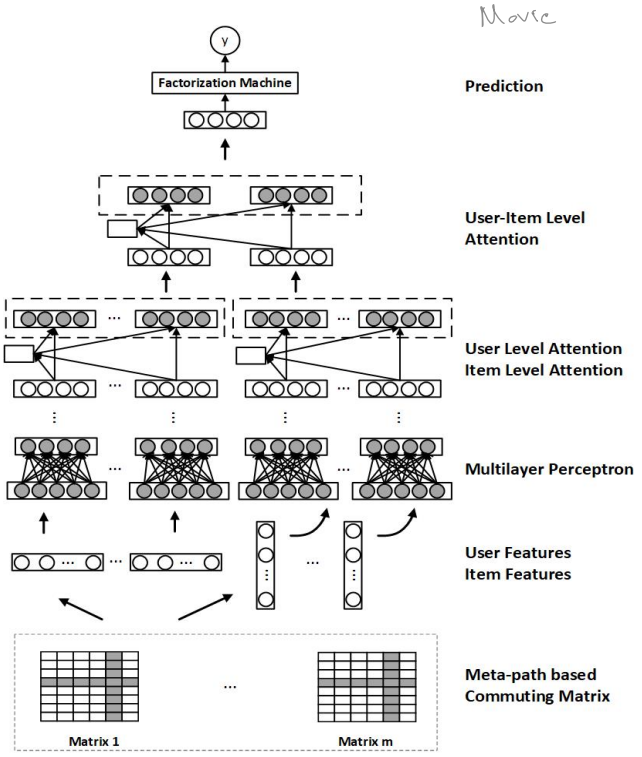

Figure 3: The HNAFM model

learning informative information, which means less layers need to be used. Sun et al. introduces commuting matrix to count the instance of meta path. In this paper, we compute the commuting matrices under the given l meta paths to encode the user preference information and feed the row and column of commuting matrices to next part of model. Each commuting matrices represents user's preference in a way. However, using single meta path possibly suffers from data sparsity problem and linear features are hard to model the real-world data, which is shown in section 4.5. Second, multilayer perceptrons(MLP) are employed to model the non-linearity and handle data sparsity problem. We take the row of the commuting matrices to learn the user features and the column of the commuting matrices to learn the item features. Finally, to fuse all the information for training, these features are combined by a hierarchical attention layer and fed to a factorization machine to predict ratings. In what follows, we will elaborate the design of HNAFM step by step.

3.3 Meta-path based Commuting Matrix

We utilize meta path to extract user and item features. As described above, meta path is able to express semantics. To extract features from meta path, we can compute the PathCount[23] of each meta path. This can be done effectively through matrix multiplication. Table 1 shows all of the meta paths used in this paper.

Given a meta path $P = (A_1, A_2, ..., A_l)$, where A_i is the object type of node i, matrix W_{A_i,A_j} is defined as the adjacency matrix between type A_i and type A_j. Then the commuting matrix of path P is calculated as $C_P = W_{A_1,A_2} \cdot W_{A_2,A_3} \cdot ... \cdot W_{A_{l-1},A_l}$. Take the meta path $User \rightarrow Movie \leftarrow User \rightarrow Movie$ as example, we compute the commuting matrix of this path as $W_{User,Movie} \cdot W_{Movie,User} \cdot W_{User,Movie}$. Therefore, by computing the commuting matrices of L meta paths, we can obtain a set of user-item commuting matrices $\hat{R}^1, \hat{R}^2, ..., \hat{R}^L \in R^{m \times n}$, where $\hat{R}^1_{ij}$ represents the meta path instance count between user i and item j(m and n are the number of users and items).

3.4 Learning Latent Features

Building recommendation model in a single meta path possibly suffer from data sparsity problem. The SemRec[22] model predicts ratings by first computing the user similarity based on a single meta path. Then a weight ensemble method is adopted to fuse the results learned from all meta paths. In section 4.5, we show that in sparse data, SemRec might suffer from data sparsity problem and lead to poor performance. In addition, linear matrix multiplication is insufficient to model non-linear real-world data. To address above problems, we use multilayer perceptron to learn the user and item features with the row and the column of commuting matrix as input respectively.

After obtaining L commuting matrices, we use multilayer perceptron to model non-linear factors. For each commuting matrix $\hat{R}^l$, the row vector $\hat{R}^l_{i*}$ represents user i' s interest in items following the semantic of l-th meta path, and similarly, the column vector $\hat{R}^l_{*j}$ represents item j' s popularity to some extent. Thus, we use the row and column vectors of each commuting matrices to represent the users and items. As shown in Figure 3, we then feed the row vector to a multilayer perceptron to learn user latent vectors. Item latent vectors are learned in the same way. More precisely, the user latent vector u^l_i is defined as

$$\begin{aligned} z_0 &= \hat{R}^l_{i*} \\ z_1 &= a(W_1^T \times z_0 + b_1) \\ &... \\ z_{n-1} &= a(W_{n-1}^T \times z_{n-2} + b_{n-1}) \\ u^l_i &= a(W_n^T \times z_{n-1} + b_n) \end{aligned} \tag{1}$$

where W_x, b_x and a_x denote the weight matrix, bias vector and activation function for the x-th layer's perceptron respectively. We can freely choose sigmoid, hyperbolic tangent(tanh) and Rectifier as activation function. As Figure 3 demonstrated, item latent vectors are learned through another multilayer perceptron with column vector $\hat{R}^l_{*j}$ as input. Thus we obtain L pairs of user latent features $u^1_i, ..., u^L_i$ and item latent features $v^1_j, ..., v^L_j$, where u^l_i and v^l_j represent user and item features generated by l-th meta path.

3.5 Attentive Latent Features Fusion

Although group lasso[32] regulation has been proposed to select useful meta paths, it still needs to tune regulation parameters. Further more, group lasso regulation can only select useful meta path, which means they delete meta paths to achieve better results. The weakness of this method is that it may drop useful information in the mean time. In this section, we introduce a two-layer attention mechanism which is able to automatically fuse user and item features in finer granularity.

$\beta(i,l)_u$ - how important meta path l for user u_i

For each user i and item j, we access a set of user features $u_i^1, ..., u_i^L$ and item features $v_j^1, ..., v_j^L$. Then, user-level and item-level attention modules take user latent features and item latent features as input respectively and output the first layer attentive weights $\beta(i,l)_u$ and $\beta(j,l)_v$ for the l-th meta paths. Thus, the final representation of user u_i and item v_j is calculated by

$$u_i = \sum_{l \in P} \beta(i,l)_u \times u_i^l \tag{2}$$

$$v_j = \sum_{l \in P} \beta(j,l)_v \times v_i^l \tag{3}$$

where P is the set of all meta paths.

After we have obtained u_i and v_j, we can use user-item-level attention module to calculate the second level attentive weight α. Similar to first layer attention, we obtain the final feature vector x by

$$x = \alpha_u \times u_i + \alpha_v \times v_j \tag{4}$$

User-level and item-level attention. The goal of this layer attention is to select meta paths that best describe users' preference and items' property and then aggregate the representation of users and items to characterize users and items respectively. Different from image captioning and machine translation where users focus on selective parts of image or sentence, users in recommendation have their own preference and pay attention to different items. Therefore, we compute the attention score $\beta(i,l)_u^s$ and $\beta(j,l)_v^s$ as

$$\beta(i,l)_u^s = a(W_{ui}^T \times u_i^l + b_{ui}) \tag{5}$$

$$\beta(j,l)_v^s = a(W_{vj}^T \times v_j^l + b_{vj}) \tag{6}$$

where W_{ui} and W_{vj} are the i-th and j-th row of matrices W_u and W_v respectively, b_{ui} and b_{vj} are the i-th and j-th elements of vector b_u and b_v respectively. The final first level attention weights are obtained by normalizing the above attention scores using Softmax

$$\beta(i,l)_u = \frac{exp(\beta(i,l)_u^s)}{\sum_{n \in P} exp(\beta(i,n)_u^s)} \tag{7}$$

$$\beta(j,l)_v = \frac{exp(\beta(j,l)_v^s)}{\sum_{n \in P} exp(\beta(j,n)_v^s)} \tag{8}$$

User-item-level attention. After obtaining u_i and v_j, we employ a user-item-level attention to measure which factor contribute to final rating most. We compute the user-item-level attention score as

$$\alpha_u^s = a(w^T \times u_i + c) \tag{9}$$

$$\alpha_v^s = a(w^T \times v_j + c) \tag{10}$$

where vector w and bias c are second layer parameters, the final attention weights are normalized as

$$\alpha_u = \frac{exp(\alpha_u^s)}{exp(\alpha_u^s) + exp(\alpha_v^s)} \tag{11}$$

$$\alpha_v = \frac{exp(\alpha_v^s)}{exp(\alpha_u^s) + exp(\alpha_v^s)} \tag{12}$$

Algorithm 1 The HNAFM Model

Input:

A set of commuting matrices $\hat{R}^1, \hat{R}^2, ..., \hat{R}^L$, represent as $\hat{R}_l$

Output:

The parameters of MLP Δ, the hierarchical attention layer Ω and factorization machine Θ.

```
Initialize Δ, Ω, Θ with Gaussian distribution.
repeat
  draw (i, j) from R_train
  for each commuting matrix R̂_l in R̂_l do
    compute user and item features according to (2).
    compute β(i,l)_u and β(j,l)_v according to (8) and (9).
  end for
  compute u_i and v_j according to (3) and (4).
  compute α_u and α_v according to (12) and (13).
  compute x according to (5).
  compute y_ij according to (14).
  for each parameters θ in Δ, Ω, Θ do
    Update θ ← θ + 2 · ∂y_ij/∂θ
  end for
until convergence
```

3.6 Model Optimization

We model the recommendation problem as rating prediction problem. We first predict the rating of users giving items and recommend items that of top rates to the target user. Factorization machine can effectively model the second-order feature interactions and is widely employed in regression and classification task. Given sample user i and sample item j, we compute input vector x^n based on (4) and adopt factorization machine to predict the rating as follows

$$\hat{y}^n(w,V) = w_0 + \sum_{i=1}^{d} w_i x_i^n + \sum_{i=1}^{d} \sum_{j=i+1}^{d} \langle v_i, v_j \rangle x_i^n x_j^n \tag{13}$$

where d, w_0, $w \in R^d$ and $V \in R^{d \times K}$ represent the dimension of input, global bias, first-order weights and second-order weights respectively. $\langle \cdot, \cdot \rangle$ is the dot product of two vectors of size K and v_i is the i-th row of the matrix V.

Then we define the square error loss as follows

$$Loss = \sum_{n=1}^{N} (y^n - \hat{y}^n(w,V))^2 \tag{14}$$

where y^n is the observed rating of the n-th sample. N is the number of all the observed ratings. We minimize it by utilizing stochastic gradient descent approach. The pseudo code of the model is shown in algorithm 1.

4 EXPERIMENTS

In this section, we conduct extensive experiments to evaluate the performance of our proposed framework. Specifically, we first introduce the datasets, evaluation metric, baseline models and experiments settings in Section 4.1, 4.2, 4.3, 4.4, respecitvely. And then we show the comprehensive performance evaluation results in Section 4.5, 4.6, 4.7, 4.8, and 4.9.

4.1 Datasets

We evaluate our proposed framework over the datasets provided by [22]. The first dataset is Douban[1]. Douban is a famous website in China where a user can rate movies, books, and songs based on her preference. The rates range from 1 to 5, where higher ratings mean users like the movie. Users can also join the group where people have similar interest. Another dataset is Yelp[2]. Similar to Douban, the user in Yelp can rate local business or post photo and reviews about them. The statistics of datasets are shown in Table 2. As illustrated in the table, Douban dataset includes 1,068,278 rating relations(User-Movie) and Yelp dataset includes 198,397 rating relations(User-Business). The difference between Douban ans Yelp is that the Douban dataset has dense rating relations and sparse social relations while the Yelp dataset has dense social relations and sparse rating relations. Density is calculated as follows

$$Density = \frac{\#Ratings}{\#Users \times \#Items} \tag{15}$$

4.2 Evaluation Metric

In this paper, we use two widely employed metrics, i.e., Root Mean Square Error(RMSE) and Mean Absolute Error(MAE), to evaluate rating prediction performance.

$$RMSE = \sqrt{\frac{\sum_{(i,j)\in R_{test}}(R_{ij} - \hat{R}_{ij})}{|R_{test}|}} \tag{16}$$

$$MAE = \frac{\sum_{(i,j)\in R_{test}}|R_{i,j} - \hat{R}_{ij}|}{|R_{test}|} \tag{17}$$

where R_{test} denotes all the user-item pairs(i, j) in the test set. $\hat{R}_{ij}$ is the predicted rate of user u_i to item v_j and $R_{i,j}$ is the observed rate of user u_i to item v_j. A smaller RMSE and MAE means better performance.

4.3 Baseline Models

In order to show the effectiveness of HNAFM, we compare our methods to following models.

- **PMF**[18]: PMF is the basic matrix factorization models which uses only the user-item rating matrix for recommendation. We use the implementation in Librec[7].
- **FMR**[20]: FMR is the factorization machine using only the user-item rating matrix. We use the code provided by pyFM[3].
- **SemRec**[22]: SemRec is semantic path based recommendation model based on weighted meta path. It computes the user similarity matrix based on weighted meta paths and ratings are predicted by combining the results learned from different meta paths. We implement it based on [22].
- **NeuMF**[9]: NeuMF fuses a linear kernel and a non-linear kernel to learn feature interaction function. To apply NeuMF to rating prediction task, we replace the pairwise loss with square error loss and change the activation function of final layer to ReLU function. We use the code provided by the authors[4].
- **FMG**[32]: FMG is a "matrix factorization + factorization machine" framework for rating prediction problem. Although the adjacency matrix along the meta path is sparse, commuting matrix is dense. We implement matrix factorization using stochastic gradient descent and factorization machine under tensorflow framework.

Table 2: The statistics of Douban/Yelp datasets

Douban				
Density	0.630%			
Relations (A-B)	Number of A	Number of B	Number of (A-B)	Avg Degrees of (A/B)
User-Movie	13367	12677	1068278	79.9/84.3
User-User	2440	2294	4085	1.7/1.8
User-Group	13367	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				
Density	0.086%			
Relations (A-B)	Number of A	Number of B	Number of (A-B)	Avg Degrees of (A/B)
User-Business	16239	14284	198397	12.2/13.9
User-User	10580	110580	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

4.4 Experimental Settings

We use the meta path shown in Table 1. We use a two layer multi layer perceptron with hidden layers having 128 units and output layers having 64 units to learn the latent features. Learning rate is set to 0.01 and Adam[12] is used to minimize the loss function. We implement our framework based on Tensorflow[5] and all experiments run in a Linux Server with Intel i7 CPU and 128GB RAM.

To compare experiments result under different data sparsity, we randomly split training dataset under different settings. Since Douban has dense rating relations, (80%, 60%, 40%, 20%)of the whole data are used for training and the remaining (20%, 40%, 60% 80%) are for testing. For Yelp dataset, we utilize (90%, 80%, 70%, 60%) of the dataset for training and the remaining(10%, 20%, 30%, 40%) are for testing. We repeat the experiments five times independently and the average results are reported in Table 3.

[1]https://movie.douban.com/
[2]https://www.yelp.com/dataset
[3] https://github.com/coreylynch/pyFM
[4]https://github.com/hexiangnan/neural_collaborative_filtering
[5]https://www.tensorflow.org

Table 3: Experiment results of different methods

Dataset	Traning Settings	PMF		FMR		SemRec		NeuMF		FMG		HNAFM	
		RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
Douban	20%	0.9134	0.6781	0.7960	0.6404	0.8580	0.6330	0.7730	0.6127	0.7500	0.5824	**0.7315**	**0.5701**
		+19.9%	+15.9%	+8.1%	+10.9%	+14.7%	+9.9%	+5.3%	+7.0%	+2.5%	+2.1%		
	40%	0.8396	0.6297	0.7623	0.6107	0.7686	0.5849	0.7544	0.5980	0.7232	0.5659	**0.7092**	**0.5553**
		+15.5%	+11.8%	+7.9%	+9.2%	+7.7%	+5.1%	+6.1%	+7.1%	+1.9%	+1.9%		
	60%	0.7814	0.5968	0.7465	0.5977	0.7420	0.5709	0.7490	0.5939	0.7139	0.5602	**0.7016**	**0.5505**
		+10.2%	+7.8%	+6.0%	+7.8%	+5.4%	+3.6%	+6.3%	+7.3%	+1.7%	+1.7%		
	80%	0.7520	0.5796	0.7367	0.5878	0.7306	0.5659	0.7451	0.5905	0.7090	0.5577	**0.6974**	**0.5486**
		+7.3%	+5.4%	+5.3%	+6.8%	+4.5%	+3.1%	+6.4%	+7.1%	+1.6%	+1.6%		
	Time(min)	1.70		2.35		29.72		3.29		214.72		122.22	
Yelp	60%	1.6368	1.2709	1.1199	0.8893	1.3894	1.0319	1.1484	0.9333	1.1256	0.8572	**1.0757**	**0.8374**
		+34.3%	+34.1%	+3.9%	+5.8%	+22.6%	+18.8%	+6.3%	+10.3%	+4.4%	+2.2%		
	70%	1.5542	1.1909	1.1151	0.8800	1.3122	0.9709	1.1288	0.9148	1.1032	0.8429	**1.0590**	**0.8238**
		+31.9%	+30.8%	+5.0%	+6.4%	+19.3%	+15.2%	+6.2%	+9.9%	+4.0%	+2.3%		
	80%	1.4915	1.1369	1.1019	0.8701	1.2565	0.9316	1.1091	0.8661	1.0852	0.8291	**1.0529**	**0.8168**
		+29.4%	+28.2%	+4.4%	+6.1%	+16.2%	+12.3%	+5.1%	+5.7%	+3.0%	+1.5%		
	90%	1.3797	1.0385	1.0912	0.8596	1.1900	0.8835	1.0937	0.8675	1.0711	0.8228	**1.0302**	**0.7981**
		+25.3%	+23.1%	+5.6%	+7.2%	+13.4%	+9.7%	+5.8%	+8.0%	+3.8%	+3.0%		
	Time(min)	1.20		2.79		31.47		2.35		108.83		21.44	

4.5 Recommendation Effectiveness

As shown in Table 3, by comparing the results in different density training settings, it could be found that HNAFM performs better under denser training data settings in both two datasets. This is because more training data brings more information about user preference, which is helpful for more accurate ratings prediction.

From Table 3, we can see that HNAFM outperforms all the models compared. By comparing PMF, FMR and NeuMF, which only use the user-item rating matrix, with SemRec, FMG and HNAFM, which use heterogeneous information for recommendation, we can see that under dense training settings, SemRec, FMG and HNAFM outperform PMF, FMR and NeuMF, when training data becomes sparse, SemRec suffers from data sparsity problem that little information is learned from single meta path. The performance of SemRec is worse than NeuMF and FMR in Yelp dataset under all training settings. FMG is using a similar framework with HNAFM but learns and fuses features in a linear way. Compared to FMG, HNAFM achieves a better performance in both datasets because we learn feature in a non-linear way and fuse features by a novel attention-based method. Furthermore, with less training data, HNAFM achieves much better performance than other methods in Yelp dataset.

We record the average running time of these models on the training process. SemRec spends most of the time in learning the weights of each meta paths. Running matrix factorization on commuting matrices is time-consuming because the product of a set of the sparse adjacency matrix is dense.The performance gap between FMR, NeuMF and FMG in Yelp datasets is not large, while FMR and NeuMF cost less time.The reason that HNAFM spend less time than FMG is that HNAFM directly learn user and item features for final prediction while FMG has to learn features separately.

Table 4: Result of HNAFM with different layers

Dataset	Metric	MLP-1	MLP-2	MLP-3	MLP-4
Douban	RMSE	0.7419	0.7326	0.7298	0.7311
	MAE	0.5829	0.5698	0.5691	0.5692
Yelp	RMSE	1.0647	1.0547	1.0516	1.0395
	MAE	0.8170	0.8144	0.8164	0.8094

4.6 Effect of MLP Layers

Previous works[8, 9] have shown that with an informative representation in a low level, stacking more layers is beneficial for better performance. Since there is little work on learning user and item feature vectors on HIN with neural networks, it is crucial to see whether using deep network structures is beneficial to rating prediction. Therefore, we further investigate Multi-Layer Perceptron (MLP) with different number of hidden layers. As it is computationally expensive to tune the size of each hidden layer separately, we use the same setting for all layers. The results are summarized in Table 4. The MLP-1 indicates the HNAFM model with one hidden layer and similar notations for others.

As we can see from Table 4, when we stack more layers, the performance is improved constantly in Yelp dataset, and the best performance of Douban dataset lies is MLP-3 whle the gap between MLP-3 and MLP-4 is small. These results indicate the effectiveness of using the deep neural network for learning user and item features vector.

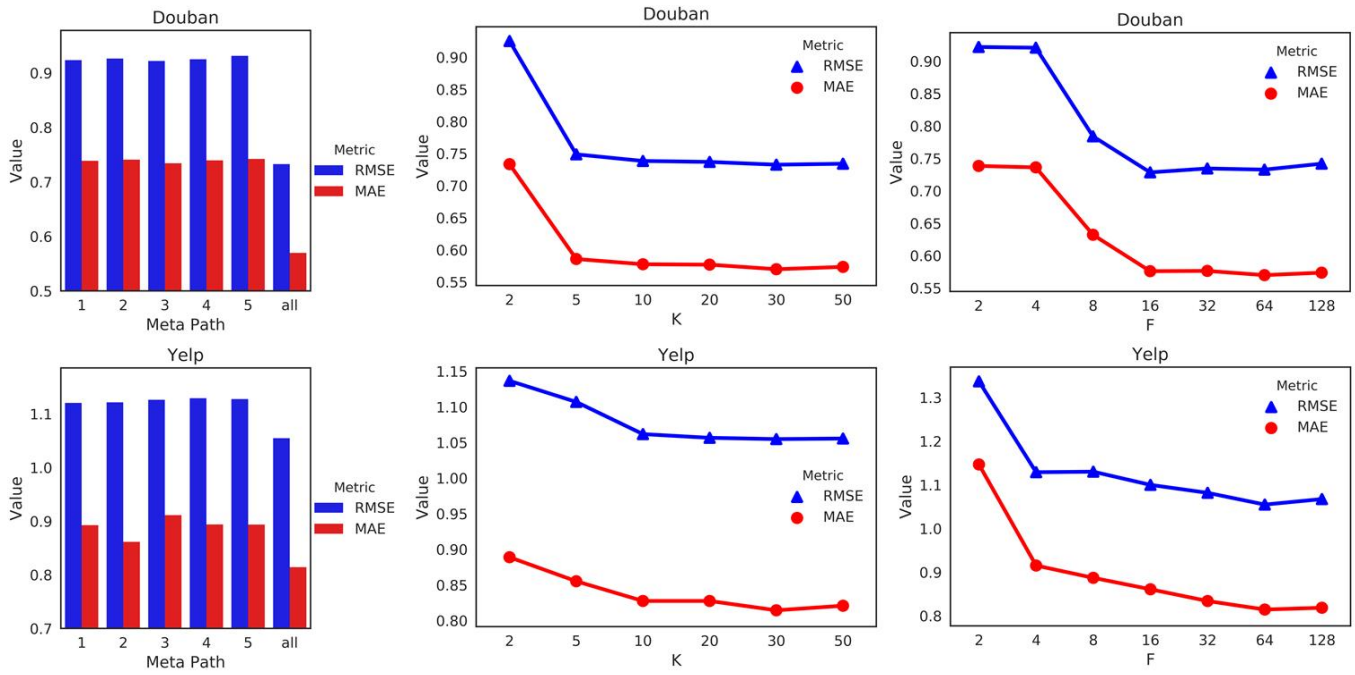

Figure 4: Performance of HNAFM model with different meta path, K, F

4.7 Recommendation with Single Meta Path

As meta paths have their own semantic, it curious to see whether fusing all the meta paths contributes to the final results. In this part, we compare the contribution of each meta path shown in Table 1. During the training process, we remove the attention layers and only learn user and item features from one meta path and then predict and evaluate the results. The RMSE and MAE of each meta path are shown in Figure 4, we use all to represent the performance of model using all the meta paths.

As shown in Figure 4, we can see that in both Douban and Yelp datasets, when using only single meta path, the performance of HNAFM has little difference. However, after fusing and integrating the information from all meta paths, the performance of HNAFM greatly improves, which demonstrates the importance of our attention layer. We will further investigate the reason why HNAFM achieve better performance in section 4.9.

4.8 Effect of Parameters F and K

In this part, we study the influence of two parameters: F and K. F is the dimension of user and item feature vectors learned from multilayer perceptron. K is the number of factors to factorize the second-order weights V of the factorization machine. We use two layer MLP to learn the feature vectors. We conduct extensive experiments in both Douban and Yelp datasets. We set K in the range of [2, 5, 10, 20, 30, 50] with F equal to 64 and set F in the range of [2, 4, 8, 16, 32, 64, 128] with K equal to 30. The results are shown in Figure 4.

From Figure 4, it could be found that with bigger value of K, the RMSE and MAE value are lower in both two dataset. Further, the performance becomes stable after a threshold value, where the threshold value of Douban is 5 and the one of Yelp is 10. Regarding to the effect of F, it could be found that with larger F, the performance in Douban improves rapidly, and after 16, the results become stable. In Yelp, the RMSE and MAE first decrease quickly before 8 and then decrease slowly until it reaches the best performance in 64.

4.9 Attention Visualization

To further investigate the effect of hierarchical attention layer, we randomly select ten users and items in both Douban and Yelp datasets and visualize the attention weights in Figure 5. The deeper color means the bigger weight. As can be seen from Figure 5, in Douban dataset, meta path 2 best describes user 1 while meta path 3 best describes user 0. All meta path contribute a same weight when describing Movie 2. In Yelp dataset, meta path 1, 2, 5 is more important than meta path 2, 3 when modelling the user 1's preference. In FMG, group lasso regulation is proposed to select useful meta path. However, when it selects meta path, it removes multiple meta paths entirely in the mean time. In this way, it will lose some information and results on a worse performance. From Figure 5, we can find that HNAFM selects useful meta path for each user and remove unnecessary meta paths by assigning a small weight. Thus, HNAFM can utilize the useful information of all meta paths and remove the noise in the mean time. These observations suggest that our model is able to capture both user preference and item property.

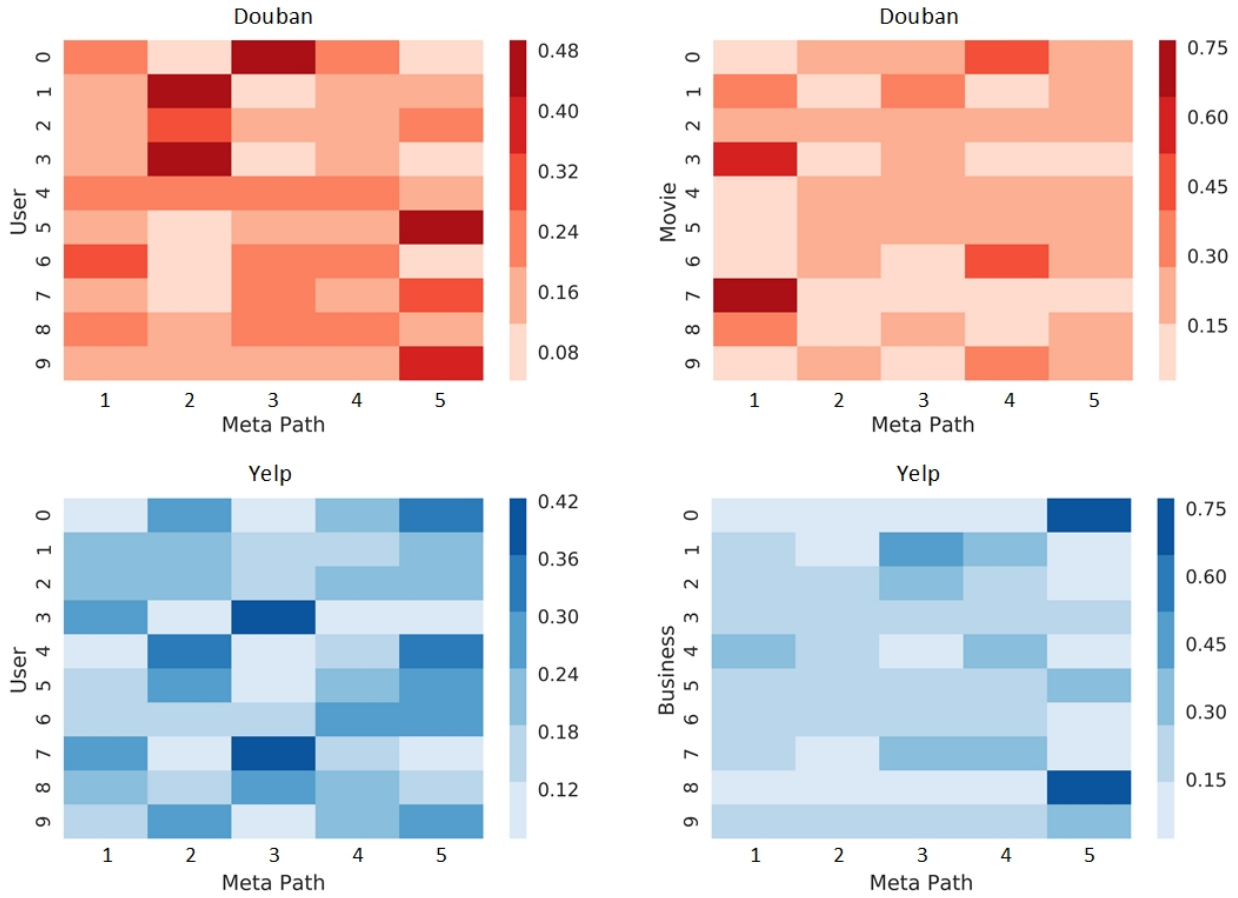

Figure 5: The visualization of attention weights of Douban and Yelp

5 CONCLUSION

In this paper, we propose Heterogeneous Neural Attentive Factorization Machine to model HIN-based recommendation. We first use multilayer perceptrons to generate user and item features based on meta paths to solve data sparsity problem and map user and item to a common latent space. Then, to fuse information learned from all the meta paths, we introduce a hierarchical attention layer to address the features that best describe users' preference and items' property. This work first introduce deep neural network and attention mechanism to HIN-based recommendation. Experimental results demonstrate the effectiveness of our framework.

In the future, we plan to improve our model in two directions. First, we will extend the meta path to meta graph to capture more complex semantic and investigate whether it would improve the recommendation performance. Second, some work[2, 5, 6] has studied the problem of representation learning in HIN which aims to learn a low-dimensional latent representation of nodes to capture the structural and semantic relations. Dong et al. develops the meta-path-guided random walk strategy in a heterogeneous network and Fu et al. learns the representation based on neural networks. The latent heterogeneous network embeddings can be further applied to various network mining tasks. It is meaningful to develop models by utilizing these embedding methods.

ACKNOWLEDGMENTS

The paper was supported by the National Key Research and Development Program (2017YFB0202200), the National Natural Science Foundation of China (61702568, U1711267), the Program for Guangdong Introducing Innovative and Entrepreneurial Teams (2017ZT07X355) and the Fundamental Research Funds for the Central Universities under Grant (17lgpy117).

REFERENCES

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* (2014).

[2] Shiyu Chang, Wei Han, Jiliang Tang, Guo-Jun Qi, Charu C Aggarwal, and Thomas S Huang. 2015. Heterogeneous network embedding via deep architectures. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 119–128.

[3] Jingyuan Chen, Hanwang Zhang, Xiangnan He, Liqiang Nie, Wei Liu, and Tat-Seng Chua. 2017. Attentive collaborative filtering: Multimedia recommendation with item-and component-level attention. In *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 335–344.

[4] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. 2016. Wide & deep learning for recommender systems. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*. ACM, 7–10.

[5] Yuxiao Dong, Nitesh V Chawla, and Ananthram Swami. 2017. metapath2vec: Scalable representation learning for heterogeneous networks. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 135–144.

[6] Tao-yang Fu, Wang-Chien Lee, and Zhen Lei. 2017. HIN2Vec: Explore Meta-paths in Heterogeneous Information Networks for Representation Learning. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, 1797–1806.

[7] Guibing Guo, Jie Zhang, Zhu Sun, and Neil Yorke-Smith. [n. d.]. LibRec: A Java Library for Recommender Systems.

[8] Xiangnan He and Tat-Seng Chua. 2017. Neural factorization machines for sparse predictive analytics. In *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 355–364.

[9] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 173–182.

[10] Shifu Hou, Yanfang Ye, Yangqiu Song, and Melih Abdulhayoglu. 2017. Hindroid: An intelligent android malware detection system based on structured heterogeneous information network. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1507–1515.

[11] Zhipeng Huang, Yudian Zheng, Reynold Cheng, Yizhou Sun, Nikos Mamoulis, and Xiang Li. 2016. Meta structure: Computing relevance in large heterogeneous information networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1595–1604.

[12] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[13] Xiangnan Kong, Bokai Cao, and Philip S Yu. 2013. Multi-label classification by mining label and instance correlations from heterogeneous information networks. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 614–622.

[14] Yehuda Koren. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 426–434.

[15] Ni Lao and William W Cohen. 2010. Relational retrieval using a combination of path-constrained random walks. *Machine learning* 81, 1 (2010), 53–67.

[16] Jing Li, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tao Lian, and Jun Ma. 2017. Neural Attentive Session-based Recommendation. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, 1419–1428.

[17] Xiang Li, Ben Kao, Yudian Zheng, and Zhipeng Huang. 2016. On transductive classification in heterogeneous information networks. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. ACM, 811–820.

[18] Andriy Mnih and Ruslan R Salakhutdinov. 2008. Probabilistic matrix factorization. In *Advances in neural information processing systems*. 1257–1264.

[19] Steffen Rendle. 2010. Factorization machines. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*. IEEE, 995–1000.

[20] Steffen Rendle. 2012. Factorization machines with libfm. *ACM Transactions on Intelligent Systems and Technology (TIST)* 3, 3 (2012), 57.

[21] Chuan Shi, Xiangnan Kong, Yue Huang, S Yu Philip, and Bin Wu. 2014. Hetesim: A general framework for relevance measure in heterogeneous networks. *IEEE Transactions on Knowledge and Data Engineering* 26, 10 (2014), 2479–2492.

[22] Chuan Shi, Zhiqiang Zhang, Ping Luo, Philip S Yu, Yading Yue, and Bin Wu. 2015. Semantic path based personalized recommendation on weighted heterogeneous information networks. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. ACM, 453–462.

[23] Yizhou Sun, Jiawei Han, Xifeng Yan, Philip S Yu, and Tianyi Wu. 2011. Pathsim: Meta path-based top-k similarity search in heterogeneous information networks. *Proceedings of the VLDB Endowment* 4, 11 (2011), 992–1003.

[24] Aaron Van den Oord, Sander Dieleman, and Benjamin Schrauwen. 2013. Deep content-based music recommendation. In *Advances in neural information processing systems*. 2643–2651.

[25] Hao Wang, Naiyan Wang, and Dit-Yan Yeung. 2015. Collaborative deep learning for recommender systems. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1235–1244.

[26] Min Xie, Hongzhi Yin, Hao Wang, Fanjiang Xu, Weitong Chen, and Sen Wang. 2016. Learning graph-based poi embedding for location-based recommendation. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. ACM, 15–24.

[27] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *International Conference on Machine Learning*. 2048–2057.

[28] Hong-Jian Xue, Xin-Yu Dai, Jianbing Zhang, Shujian Huang, and Jiajun Chen. 2017. Deep matrix factorization models for recommender systems. *static. ijcai. org* (2017).

[29] Quanzeng You, Hailin Jin, Zhaowen Wang, Chen Fang, and Jiebo Luo. 2016. Image captioning with semantic attention. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 4651–4659.

[30] Xiao Yu, Xiang Ren, Yizhou Sun, Quanquan Gu, Bradley Sturt, Urvashi Khandelwal, Brandon Norick, and Jiawei Han. 2014. Personalized entity recommendation: A heterogeneous information network approach. In *Proceedings of the 7th ACM international conference on Web search and data mining*. ACM, 283–292.

[31] Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Wei-Ying Ma. 2016. Collaborative knowledge base embedding for recommender systems. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. ACM, 353–362.

[32] Huan Zhao, Quanming Yao, Jianda Li, Yangqiu Song, and Dik Lun Lee. 2017. Meta-graph based recommendation fusion over heterogeneous information networks. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 635–644.