Applied Intelligence (2019) 49:3558–3569
https://doi.org/10.1007/s10489-019-01469-6

Neural variational matrix factorization for collaborative filtering in recommendation systems

Teng Xiao[1] · **Hong Shen**[1,2]

Published online: 22 April 2019

Abstract

Matrix factorization as a popular technique for collaborative filtering in recommendation systems computes the latent factors for users and items by decomposing a user-item rating matrix. Most matrix factorization methods including probabilistic matrix factorization that projects (parameterized) users and items probabilistic matrices to maximize their inner product suffer from data sparsity and result in poor latent representations of users and items. To alleviate these problems, we propose a novel deep generative model, namely Neural Variational Matrix Factorization, that incorporates side information (features) of both users and items to capture better latent representations of them for more effective collaborative-filtering recommendation. Our model consists of two end-to-end variational autoencoder neural networks, namely user neural network and item neural network respectively, that are capable of learning complex nonlinear distributed representations of users and items through our proposed variational inference. We present a Stochastic Gradient Variational Bayes estimator to estimate the intractable posterior distributions of latent factors of users and items and parameters of our model, and derive the variational evidence lower bounds of the model. Experiments conducted on three publicly available datasets show that our model significantly outperforms the state-of-the-art methods on recommendation accuracy measured by Hit Ratio and Normalized Discounted Cumulative Gain respectively.

Keywords Collaborative filtering · Recommendation · Matrix factorization · Deep generative process · Variational inference

1 Introduction

Recommendation systems (RS) are of paramount importance in social networks and e-commerce platforms. RS aims at inferring users' preferences over items by utilizing their previous interactions. Traditional methods for RS can be categorized into two classes [1]: content-based and collaborative filtering (CF). Content-based methods make use of the features of users and items and recommend items that are similar to the items that the users have liked before. CF methods utilize previous rating information obtained from users with similar interest to recommend items to the users, and have been widely used in RS due to their impressive performance.. Matrix factorization (MF) is one of the most successful and popular CF approaches that first infers users' and items' latent factors from a user-item rating matrix and then recommends items to users who share similar latent factors to the items [2]. Most matrix factorization methods including probabilistic matrix factorization (PMF) that projects (parameterized) users and items probabilistic matrices to maximize their inner product suffer from the problems of poor latent representations of users and items and data sparsity.

To overcome these problems, we propose a novel deep generative model, namely Neural Variational Matrix Factorization (NVMF), that incorporates side information (features) of both users and items to capture better latent representations of them for more effective collaborative-filtering recommendation. Our NVMF infers the posterior distribution (Bayesian estimation) of the latent factors of users and items through two end-to-end variational autoencoder neural networks, namely "user neural network" and "item neural network", consisting of a generative network (i.e., decoder) and a inference network (i.e., encoder) respectively,

✉ Hong Shen
hongsh01@gmail.com

Teng Xiao
cstengxiao@gmail.com

[1] School of Data and Computer Science, Sun Yat-sen University, Guangzhou, China

[2] School of Computer Science, The University of Adelaide, Adelaide, Australia

to model the generative process of user's latent factor and item's latent factor. The main contributions of this paper are:

- We propose a novel deep generative model, namely neural variational matrix factorization (NVMF), that incorporates side information via a novel deep generative process to capture more subtle relations between latent factors and side information. To the best of our knowledge, this is the first work which combines deep generative model with matrix factorization for collaborative filtering.
- To effectively infer NVMF, we propose an inference algorithm applying Stochastic Gradient Variational Bayes estimation on user and item neural networks to approximately compute parameter of our NVMF and infer latent factors of users and items, which improves the solution quality by following traditional inference methods that optimizes the variational parameters one by one. We derive the variational evidence lower bounds for our proposed model.
- We extensively conduct experiments and performance evaluation. The experiment results show that our proposed NVMF method outperforms major state-of-the-art CF methods on recommendation accuracy.

The paper is organized as follows: Section 2 introduces related work; Section 3 gives preliminaries and problem statement; Section 4 describes the deep generative process of our user and item neural networks; Section 5 presents the inference process for estimating the posterior probability distributions of latent variables of users and items; Section 6 shows the results of experiments and performance evaluation. Section 7 concludes the paper. A conference version containing some preliminary results has been accepted by PAKDD 2019 [35].

2 Related work

On matrix factorization (MF) for collaborative filtering (CF), a series of improved methods have been proposed, including non-negative matrix factorization (NNMF) [3], max-margin matrix factorization [4] and, most remarkably, probabilistic matrix factorization (PMF) [5] that projects (parameterized) users and items probabilistic matrices to maximize their inner product. To overcome the data sparsity problem suffered by these methods, hybrid methods that enjoy the advantages of different categories of CF methods have been proposed. Some hybrid methods [6–10] take the advantages of both content-based and PMF methods, and incorporate side information, such as demographics of a user, type of an item, etc., into PMF. Regardless of their better performance compared to the PMF, we find that most of hybrid methods apply Gaussian or Poisson distributions to model the generative process of users' ratings and may result in learning poor representations of users and items from complex data. Furthermore, most of these hybrid methods also incorporate side information via a linear regression way which hinders themselves from capturing the complex relations between latent factors and side information.

On the other hand, deep learning has achieved state-of-art results in various fields such as natural language processing [11], computer vision [12] and speech recognition [13, 14] due to its powerful ability of representation. Thus, some researchers have applied deep learning to the task of CF. Deep collaborative Filtering (DCF) [15] is a general deep architecture for hybrid CF by integrating PMF with deep neural networks. Wang et al. [16] proposed collaborative deep learning (CDL) to integrate stacked denoising autoencoder (SDAE) into probabilistic matrix factorization which get state-of-the-art result over real world datasets. In order to solve the matrix sparsity and cold start problems, Collaborative Filtering Neural network (CFN) [17] incorporates the side information to mitigate sparsity and cold start problems. Recently, Dong et al. [18] proposed the additional stacked denoising autoencoder (aSDAE). The aSDAE combines autoencoder with matrix factorization and incorporates side information into each layer of the neural network, which makes itself able to obtain the best possible result. However, these deep learning-based methods are all deterministic and unable to capture the uncertainties of the latent representations of users and items. For these models, using very deep stacked neural network makes the models themselves too deep to train and are easy to overfit the data. Thus, how to learn effective and robust latent representations of users and items needs to be further investigated.

Recently, the deep generative model [19–21] has attracted significant research efforts since it has both non-linearity of neural network, and can obtain more subtle latent representations of users and items due to its Bayesian nature. Li et al. [22] proposed Collaborative Variational Autoencoder (CVAE), which utilizes VAE to extract latent item information and incorporates it into matrix factorization. Liang et al. [23] proposed the VAE-CF model to the CF task. Chen et al. proposed a collective VAE [24] which incorporates side information into VAE-CF. However, these VAE-based methods do not consider side information of both users and items. They simply use the same Gaussian priors for all users and items, which is unrealistic for most cases and could lead to poor latent representation and suffer from the *Posterior-collapse* problem [25] of VAE.

In this paper, we solve the above problems by proposing a new CF method of neural-network based variational matrix factorization that models the relationship of latent factor and side information by a novel deep generative process so as

to alleviate the data sparsity problem and effectively learn latent representations. In addition, our proposed method as a variant of variational autoencoder for MF differs from the existing neural network based hybrid methods by taking the neural network as a complete Bayesian probabilistic framework to combine the advantages of both deep learning and probabilistic matrix factorization for the purpose of capturing more subtle latent representations and the uncertainties of latent representations.

3 Preliminaries

We first introduce Probabilistic Matrix Factorization and then give our problem definition.

3.1 Probabilistic matrix factorization

Our proposed neural variational matrix factorization method is built on the well-know probabilistic matrix factorization (PMF) [26] that can be viewed as parameterized projection of two given probabilistic matrices such that their inner product is maximized. The goal of PMF is to find two low-rank latent factor matrices $\boldsymbol{U}$ and $\boldsymbol{V}$ to approximate the user-item feedback matrix: $\boldsymbol{R} \approx \boldsymbol{U}^\top \boldsymbol{V}$. PMF gives a probabilistic framework to learn the latent factor matrices by assuming a linear model with Gaussian observation noise and Gaussian priors on the latent factors (see Fig. 1 left) to minimize the sum-of-squared-errors between observed feedback matrix $\boldsymbol{R}$ and the inner product of latent factors $(\boldsymbol{U}, \boldsymbol{V})$ with two quadratic regularization terms from the object function (3). Specifically, PMF considers the conditional distribution of the observed rating matrix $\boldsymbol{R}$ given latent factors $\boldsymbol{U}$ and $\boldsymbol{V}$, and the prior distribution of $\boldsymbol{U}$ and $\boldsymbol{V}$ as follows:

$$p(\boldsymbol{R}|\boldsymbol{U}, \boldsymbol{V}, \alpha) = \prod_{i=1}^{M}\prod_{j=1}^{N}\left[\mathcal{N}(R_{ij}|\boldsymbol{u_i}^\top \boldsymbol{v_j}, \alpha^{-1})\right]^{I_{ij}}, \quad (1)$$

$$p(\boldsymbol{U}|\alpha_1) = \prod_{i=1}^{M}\mathcal{N}(\boldsymbol{u_i}|0, \alpha_1^{-1}\boldsymbol{I}), \quad p(\boldsymbol{V}|\alpha_2) = \prod_{j=1}^{N}\mathcal{N}(\boldsymbol{v_j}|0, \alpha_2^{-1}\boldsymbol{I}), \quad (2)$$

where $\mathcal{N}(\cdot, \cdot)$ represents a Gaussian distribution with means and precision, and I_{ij} represents user i has rated on item j is not empty entities in $\boldsymbol{R}$. α, α_1, α_2 are parameters for precisions in the corresponding Gaussian distributions. PMF is learned by finding (Maximizing a posterior) MAP estimation. Maxing the posterior distribution is equivalent to minimizing the following loss function:

$$\arg\min_{\boldsymbol{U},\boldsymbol{V}} \mathcal{L}(\boldsymbol{U}, \boldsymbol{V}) = \frac{1}{2}\sum_{i=1}^{M}\sum_{j=1}^{N} I_{ij}(R_{ij} - \boldsymbol{u_i}^\top \boldsymbol{v_j})^2 + \frac{\lambda_1}{2}\sum_{i=1}^{M}\|\boldsymbol{u_i}\|_F^2 + \frac{\lambda_2}{2}\sum_{j=1}^{N}\|\boldsymbol{v_j}\|_F^2, \quad (3)$$

where $\|\cdot\|_F$ denotes the Frobenius norm, $\lambda_1 = \alpha_1/\alpha$ and $\lambda_2 = \alpha_2/\alpha$. Some improved methods based on PMF have been proposed such as Bayesian matrix factorization [8, 9] and Hierarchical Bayesian Matrix factorization [7] to incorporate side information into PMF and learn the full posterior of latent factors $\boldsymbol{U}$ and $\boldsymbol{V}$. However, these model incorporate side information via a linear regression ways which hinders them to capture subtle and non-linear relations between latent factors of users and items between their side information. The variational parameters in these models [7–9] are also too many to store or optimize local variational parameters. To address the drawbacks above, in the following section (Section 4), we propose a neural variational matrix factorization (NVMF) that integrates side information into probabilistic MF via a deep generative model and propose a Stochastic Gradient Variational Bayes [19] inference algorithm for our NVMF to avoid the optimization of the every variational parameters.

3.2 Problem definition and notations

The problem we address in this paper is to infer the posterior latent factors of users and items and predict the missing value in user-item feedback matrix $\boldsymbol{R}$ given $\boldsymbol{R}$, $\boldsymbol{F}$ and $\boldsymbol{G}$ via probabilistic matrix factorization. Accordingly, our neural-network based neural variational matrix factorization,

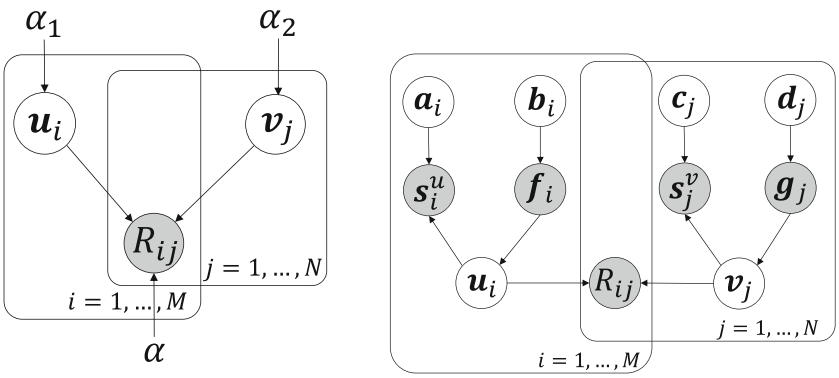

Fig. 1 Graphical models of Probabilistic Matrix Factorization (left) and our model NVMF (right), where shaded nodes are observed variables

NVMF, is essentially a function Ξ that satisfies the following:

$$\boldsymbol{R}, \boldsymbol{F}, \boldsymbol{G} \xrightarrow{\Xi} \boldsymbol{U}, \boldsymbol{V}, \tag{4}$$

where $\boldsymbol{R}$ is a feedback matrix such as rating matrix with M and N being the total number of users and items, respectively, $\boldsymbol{F} \in \mathbb{R}^{P \times M}$ and $\boldsymbol{G} \in \mathbb{R}^{Q \times N}$ are the side information matrix of all users and items, respectively, with P and Q being the dimensions of each user's and item's side information, respectively; $\boldsymbol{U} = [\boldsymbol{u}_1, ...\boldsymbol{u}_M] \in \mathbb{R}^{D \times M}$ and $\boldsymbol{V} = [\boldsymbol{v}_1, ...\boldsymbol{v}_N] \in \mathbb{R}^{D \times N}$ are the two rank matrices serving for users and items, respectively, with D denoting the dimensions of latent factor space..

There are two types of feedback matrix which are explicit feedback ($\boldsymbol{R} \in \mathbb{R}^{M \times N}$) and implicit feedback ($\boldsymbol{R} \in \{0, 1\}^{M \times N}$). Like recent recommendation methods [22, 27], we focus on implicit feedback in our paper as this is a more challenging situation. For convenient discussion, we represent each user i's rating scores including the missing/unobserved ones over all items as $\boldsymbol{s}_i^u = [R_{i1}, ..., R_{iN}] \in \mathbb{R}^{N \times 1}$, where R_{ij} is an element in $\boldsymbol{R}$. Similarly, we represent each item j's rating scores from all users including those who do not provide rating for j as $\boldsymbol{s}_j^v = [R_{1j}, ..., R_{Mj}] \in \mathbb{R}^{M \times 1}$. We call $\boldsymbol{s}_i^u$ and $\boldsymbol{s}_j^v$ as the collaborative information of user i and item j, respectively. Obviously, our task is to infer each user's and item's latent factors, $\boldsymbol{u}_i$ and $\boldsymbol{v}_j$ through $\boldsymbol{R}$, $\boldsymbol{F}$ and $\boldsymbol{G}$. Then we can use $\boldsymbol{u}_i$ and $\boldsymbol{v}_j$ to predict the missing R_{ij}.

Like most probabilistic models [7, 9, 10], our model requires three steps of implementation: 1) Define a generative model to describe the generative process of the observed variables ($\boldsymbol{R}$); (2) infer the posterior of the latent variables ($\boldsymbol{U}$ and $\boldsymbol{V}$) conditioned on the observed variables; 3) use the inferred posterior to predict the missing value in $\boldsymbol{R}$. Note that, unlike traditional MF methods [2] which gets the point vector of latent factors, the Bayesian matrix factorization mentioned above and our model is to infer the full posterior distributions of latent factors and then uses the inferred posteriors to predict missing value.

Table 1 is a list of mathematical symbols used in the paper:

Table 1 Glossary

Symbol	Description
$\boldsymbol{R}$	user-item rating (feedback) matrix
$\boldsymbol{U}$	user latent factor matrix
$\boldsymbol{V}$	item latent factor matrix
$\boldsymbol{F}$	user feature matrix
$\boldsymbol{G}$	item feature matrix
R_{ij}	the element in the i-th row and j-th column of $\boldsymbol{R}$
$\boldsymbol{s}_i^u$	the collaborative information of user i
$\boldsymbol{s}_j^v$	the collaborative information of item j
$\boldsymbol{a}_i$	the latent vector for $\boldsymbol{s}_i^u$
$\boldsymbol{b}_i$	the latent vector for $\boldsymbol{f}_i$
$\boldsymbol{c}_j$	the latent vector for $\boldsymbol{s}_j^v$
$\boldsymbol{d}_j$	the latent vector for $\boldsymbol{g}_j$
$\boldsymbol{f}_i$	the feature of user i
$\boldsymbol{g}_j$	the feature of item j
$\boldsymbol{A}$	the matrix serving for all latent vectors $\boldsymbol{a}_i$
$\boldsymbol{B}$	the matrix serving for all latent vectors $\boldsymbol{b}_i$
$\boldsymbol{C}$	the matrix serving for all latent vectors $\boldsymbol{c}_j$
$\boldsymbol{D}$	the matrix serving for all latent vectors $\boldsymbol{d}_j$
$\mathcal{Z}$	the set of all latent variables
$\mathcal{O}$	the set of all observed varibles
$\boldsymbol{u}_i$	the latent factor of user i
$\boldsymbol{v}_j$	the latent factor of item j
$\boldsymbol{\theta}$, $\boldsymbol{\alpha}$ and $\boldsymbol{\gamma}$	the parameters of the user network
$\boldsymbol{\tau}$, $\boldsymbol{\beta}$ and $\boldsymbol{\psi}$	the parameters of the item network

4 The generative process

The generative process of the probabilistic graphic model of our NVMF is shown in Fig. 1 (right). Unlike other matrix factorization models (Fig. 1 (left)) that directly utilize the rating value R_{ij} in user-item matrix $\boldsymbol{R}$ to infer the latent factors users and items, our NVMF model considers a more general situation that the i-th user latent factor $\boldsymbol{u}_i$ can be jointly inferred by both user's rating history on all items $\boldsymbol{s}_i^u$ (the collaborative information of user i) and user's features $\boldsymbol{f}_i$. Similarly, the j-th item latent factor $\boldsymbol{v}_j$ can be jointly inferred by both users' rating history on the item $\boldsymbol{s}_j^v$ (the collaborative information of item j) and its own features $\boldsymbol{g}_j$. Under this consideration, we first model the features of users and items through latent variable model. Although we do not know the real distributions of user features and item features, we know that any distribution can be generated by mapping the standard Gaussian through a sufficiently complicated function [28].

Thus for user i, given a standard Gaussian latent variable $\boldsymbol{b}_i$ assigned to the user, his features $\boldsymbol{f}_i$ are generated from its latent variable $\boldsymbol{b}_i$ through a neural network, which is called "user generative network" (see Fig. 2), and are governed by the parameter θ in the network such that we have:

$$\boldsymbol{b}_i \sim \mathcal{N}(0, \boldsymbol{I}_{K_b}), \quad \boldsymbol{f}_i \sim p_\theta(\boldsymbol{f}_i|\boldsymbol{b}_i), \tag{5}$$

where $\boldsymbol{I}_{K_b}$ is the covariance matrix, K_b is the dimension of $\boldsymbol{b}_i$, and the specific form of the probability of generating $\boldsymbol{f}_i$ given $\boldsymbol{b}_i$, $p_\theta(\boldsymbol{f}_i|\boldsymbol{b}_i)$, depends on the type of data. For instance, if $\boldsymbol{f}_i$ is binary vector, $p_\theta(\boldsymbol{f}_i|\boldsymbol{b}_i)$ can be a multivariate Bernoulli distribution $Ber(F_\theta(\boldsymbol{b}_i))$ with $F_\theta(\cdot)$ being the highly no-linear function parameterized by the parameter θ in the neural networks. Similarly, for item j,

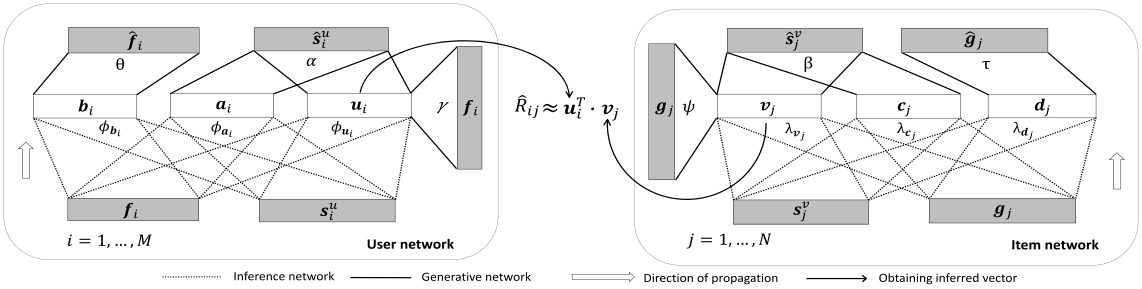

Fig. 2 The architecture of NVMF: user network (left) and item network (right) infer latent factors for users and items, shaded rectangles are observed vectors

its features $\boldsymbol{g}_j$ are modeled to be generated from a standard Gaussian latent variable $\boldsymbol{d}_j$ through another generative network, which is called "item generative network" (see Fig. 2), and are governed by a parameter τ in the network such that we have:

$$\boldsymbol{d}_j \sim \mathcal{N}(0, \boldsymbol{I}_{K_d}), \quad \boldsymbol{g}_j \sim p_\tau(\boldsymbol{g}_j|\boldsymbol{d}_j), \tag{6}$$

where $\boldsymbol{I}_{K_d}$ is the covariance matrix and K_d is the dimension of $\boldsymbol{d}_j$. Traditional PMF assumes the prior distributions of user latent factor $\boldsymbol{u}_i$ and item latent factor $\boldsymbol{v}_j$ are standard Gaussian distributions and predict rating only through collaborative information such as the user-item feedback matrix. In our model, to further enhance the performance, besides the collaborative information, we believe the user's features $\boldsymbol{f}_i$ can also positively contribute to the inference of his latent factor $\boldsymbol{u}_i$. Similarly, for better inferring the j-th item's latent factor $\boldsymbol{v}_j$, we also fully utilize user's features $\boldsymbol{g}_j$. Unlike most MF methods [7, 8, 29] that incorporate side information via linear regression, in order to get more subtle latent relations, we consider the conditional prior $p(\boldsymbol{u}_i|\boldsymbol{f}_i)$ and $p(\boldsymbol{v}_j|\boldsymbol{g}_j)$ are Gaussian distributions such that we have $p(\boldsymbol{u}_i|\boldsymbol{f}_i) = \mathcal{N}(\mu_u(\boldsymbol{f}_i), \Sigma_u(\boldsymbol{f}_i))$ and $p(\boldsymbol{v}_j|\boldsymbol{g}_j) = \mathcal{N}(\mu_v(\boldsymbol{g}_j), \Sigma_v(\boldsymbol{g}_j))$, where

$$\mu_u(\boldsymbol{f}_i) = F_{\mu_u}(\boldsymbol{f}_i), \qquad \Sigma_u(\boldsymbol{f}_i) = \mathrm{diag}(\exp(F_{\delta_u}(\boldsymbol{f}_i))), \tag{7}$$
$$\mu_v(\boldsymbol{g}_j) = G_{\mu_v}(\boldsymbol{g}_j), \qquad \Sigma_v(\boldsymbol{g}_j) = \mathrm{diag}(\exp(G_{\delta_v}(\boldsymbol{g}_j))), \tag{8}$$

where $F_{\mu_u}(\cdot)$, $F_{\delta_u}(\cdot)$, are the two highly non-linear functions parameterized by μ_u and δ_u in the neural network, i.e., the user prior network, serving for all users, and $G_{\mu_v}(\cdot)$ and $G_{\delta_v}(\cdot)$ are the two non-linear ones parameterized by μ_v and δ_v in another neural network, i.e., the item prior network, serving for all items, respectively. For simplicity, note that we set $\gamma = \{\mu_u, \delta_u\}$ and $\psi = \{\mu_v, \delta_v\}$. For the collaborative information of user i ($\boldsymbol{s}_i^u$), we assign a standard Gaussian latent variable $\boldsymbol{a}_i$ to it and believe user latent factor $\boldsymbol{u}_i$ can potentially affect user collaborative information. Then we consider $\boldsymbol{s}_i^u$ is generated from both a standard Gaussian latent variable $\boldsymbol{a}_i$ and user latent factor $\boldsymbol{u}_i$, and is governed by the parameter α in the generative network (see Fig. 2 and the caption in the figure) such that we have:

$$\boldsymbol{a}_i \sim \mathcal{N}(0, \boldsymbol{I}_{K_a}), \quad \boldsymbol{s}_i^u \sim p_\alpha(\boldsymbol{s}_i^u|\boldsymbol{a}_i, \boldsymbol{u}_i), \tag{9}$$

where $\boldsymbol{I}_{K_a}$ is the covariance matrix and K_a is the dimension of $\boldsymbol{a}_i$. Similarly, the j-th item's collaborative information, $\boldsymbol{s}_j^v$, is generated from its standard Gaussian latent variable $\boldsymbol{c}_j$ and item latent factor $\boldsymbol{v}_j$, and is governed by the parameter β in the generative network such that we have:

$$\boldsymbol{c}_j \sim \mathcal{N}(0, \boldsymbol{I}_{K_c}), \quad \boldsymbol{s}_j^v \sim p_\beta(\boldsymbol{s}_j^v|\boldsymbol{c}_j, \boldsymbol{v}_j), \tag{10}$$

where $\boldsymbol{I}_{K_c}$ is the covariance matrix and K_c is the dimension of $\boldsymbol{c}_j$. Similar to the form of the probability distribution, $p_\theta(\boldsymbol{f}_i|\boldsymbol{b}_i)$, in (5), the specific forms of the probability distributions in (6), (9) and (10) depend on the type of data. The rating R_{ij} is drawn from the Gaussian distribution whose mean is the inner product of the user i and item j latent factor representations such that we have:

$$p(R_{ij}|\boldsymbol{u}_i, \boldsymbol{v}_j) = \mathcal{N}(\boldsymbol{u}_i^\top \boldsymbol{v}_j, C_{ij}^{-1}). \tag{11}$$

where C_{ij}^{-1} is the precision of Gaussian distribution, and similar to the collaborative topic modeling [27], C_{ij} serves as a confidence parameter for rating R_{ij}, which is defined as:

$$C_{ij} = \begin{cases} \varphi_1 & \text{if} \quad R_{ij} \neq 0, \\ \varphi_2 & \text{if} \quad R_{ij} = 0, \end{cases} \tag{12}$$

where φ_1 and φ_2 are the parameters satisfying $\varphi_1 > \varphi_2 > 0$, the basic reason behind which is that if $R_{ij} = 0$ it means the user i is not interested in the item j or the user i is unaware of it. Figure 1 shows the graph model corresponding to the generative process defined above (i.e., (11), (10), (9), (8), (7), (6) and (5)). Based on the generative process, the joint

probability of all observation and the latent variables can be written as follows:

$$p(\boldsymbol{R}, \boldsymbol{F}, \boldsymbol{G}, \mathcal{Z}; \boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{\gamma}, \boldsymbol{\tau}, \boldsymbol{\beta}, \boldsymbol{\psi}) = \prod_{i=1}^{M}\prod_{j=1}^{N} \underbrace{p(\boldsymbol{a}_i)p(\boldsymbol{b}_i)p_{\boldsymbol{\theta}}(\boldsymbol{f}_i|\boldsymbol{b}_i)p_{\boldsymbol{\gamma}}(\boldsymbol{u}_i|\boldsymbol{f}_i)p_{\boldsymbol{\alpha}}(\boldsymbol{s}_i^u|\boldsymbol{a}_i, \boldsymbol{u}_i)}_{\text{for users}} \cdot \underbrace{p(\boldsymbol{c}_j)p(\boldsymbol{d}_j)p_{\boldsymbol{\tau}}(\boldsymbol{g}_j|\boldsymbol{d}_j)p_{\boldsymbol{\psi}}(\boldsymbol{v}_j|\boldsymbol{g}_j)p_{\boldsymbol{\beta}}(\boldsymbol{s}_j^v|\boldsymbol{c}_j, \boldsymbol{v}_j)}_{\text{for items}} p(R_{ij}|\boldsymbol{u}_i, \boldsymbol{v}_j), \tag{13}$$

Thus the marginal distribution of all observed variables ($\boldsymbol{R}$, $\boldsymbol{F}$ and $\boldsymbol{G}$) is:

$$p(\boldsymbol{R}, \boldsymbol{F}, \boldsymbol{G}; \boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{\gamma}, \boldsymbol{\tau}, \boldsymbol{\beta}, \boldsymbol{\psi}) = \int \prod_{i=1}^{M}\prod_{j=1}^{N} \underbrace{p(\boldsymbol{a}_i)p(\boldsymbol{b}_i)p_{\boldsymbol{\theta}}(\boldsymbol{f}_i|\boldsymbol{b}_i)p_{\boldsymbol{\gamma}}(\boldsymbol{u}_i|\boldsymbol{f}_i)p_{\boldsymbol{\alpha}}(\boldsymbol{s}_i^u|\boldsymbol{a}_i, \boldsymbol{u}_i)}_{\text{for users}} \cdot \underbrace{p(\boldsymbol{c}_j)p(\boldsymbol{d}_j)p_{\boldsymbol{\tau}}(\boldsymbol{g}_j|\boldsymbol{d}_j)p_{\boldsymbol{\psi}}(\boldsymbol{v}_j|\boldsymbol{g}_j)p_{\boldsymbol{\beta}}(\boldsymbol{s}_j^v|\boldsymbol{c}_j, \boldsymbol{v}_j)}_{\text{for items}} p(R_{ij}|\boldsymbol{u}_i, \boldsymbol{v}_j) d\mathcal{Z}, \tag{14}$$

where $\mathcal{Z} = \{\boldsymbol{U}, \boldsymbol{V}, \boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, \boldsymbol{D}\}$ is the set of all latent variables in (15) that need to be inferred, and $\mathcal{Z}_{ij} = \{\boldsymbol{u}_i, \boldsymbol{v}_j, \boldsymbol{a}_i, \boldsymbol{b}_i, \boldsymbol{c}_j, \boldsymbol{d}_j\}$.

5 The inference process

We now show how to infer the posterior distributions over the latent variables $\mathcal{Z}$ and parameters $(\boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{\gamma}, \boldsymbol{\tau}, \boldsymbol{\beta}, \boldsymbol{\psi})$ to maximize the marginal distribution in (16), optimize parameters, and predict rating of R.

5.1 Estimation of latent variables and parameters

Based on the generative process defined in last section, the joint probability of all observation and the latent variables can be written as follows:

$$p(\boldsymbol{R}, \boldsymbol{F}, \boldsymbol{G}, \mathcal{Z}; \boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{\gamma}, \boldsymbol{\tau}, \boldsymbol{\beta}, \boldsymbol{\psi}) = \prod_{i=1}^{M}\prod_{j=1}^{N} \underbrace{p(\boldsymbol{a}_i)p(\boldsymbol{b}_i)p_{\boldsymbol{\theta}}(\boldsymbol{f}_i|\boldsymbol{b}_i)p_{\boldsymbol{\gamma}}(\boldsymbol{u}_i|\boldsymbol{f}_i)p_{\boldsymbol{\alpha}}(\boldsymbol{s}_i^u|\boldsymbol{a}_i, \boldsymbol{u}_i)}_{\text{user part}} \cdot \underbrace{p(\boldsymbol{c}_j)p(\boldsymbol{d}_j)p_{\boldsymbol{\tau}}(\boldsymbol{g}_j|\boldsymbol{d}_j)p_{\boldsymbol{\psi}}(\boldsymbol{v}_j|\boldsymbol{g}_j)p_{\boldsymbol{\beta}}(\boldsymbol{s}_j^v|\boldsymbol{c}_j, \boldsymbol{v}_j)}_{\text{item part}} \underbrace{p(R_{ij}|\boldsymbol{u}_i, \boldsymbol{v}_j)}_{\text{PMF part}}, \tag{15}$$

From (15), we can view our NVMF as three parts, i.e., the user part, the item part and the PMF part. The user part and item part mainly model the inner relations between side information and latent factors in users and items, respectively. The PMF part models the across interactions between latent factors of users and items. Based on the joint distribution of all variables (15), the marginal distribution of all observed variables ($\boldsymbol{R}$, $\boldsymbol{F}$ and $\boldsymbol{G}$) is:

$$p(\boldsymbol{R}, \boldsymbol{F}, \boldsymbol{G}; \boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{\gamma}, \boldsymbol{\tau}, \boldsymbol{\beta}, \boldsymbol{\psi}) = \int \prod_{i=1}^{M}\prod_{j=1}^{N} p(\boldsymbol{a}_i)p(\boldsymbol{b}_i)p_{\boldsymbol{\theta}} \times (\boldsymbol{f}_i|\boldsymbol{b}_i)p_{\boldsymbol{\gamma}}(\boldsymbol{u}_i|\boldsymbol{f}_i)p_{\boldsymbol{\alpha}}(\boldsymbol{s}_i^u|\boldsymbol{a}_i, \boldsymbol{u}_i) \cdot p(\boldsymbol{c}_j)p(\boldsymbol{d}_j)p_{\boldsymbol{\tau}}(\boldsymbol{g}_j|\boldsymbol{d}_j)p_{\boldsymbol{\psi}}(\boldsymbol{v}_j|\boldsymbol{g}_j)p_{\boldsymbol{\beta}}(\boldsymbol{s}_j^v|\boldsymbol{c}_j, \boldsymbol{v}_j)p(R_{ij}|\boldsymbol{u}_i, \boldsymbol{v}_j)d\mathcal{Z}, \tag{16}$$

where $\mathcal{Z} = \{\boldsymbol{U}, \boldsymbol{V}, \boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, \boldsymbol{D}\}$ is the set of all latent variables in (15) that need to be inferred, and $\mathcal{Z}_{ij} = \{\boldsymbol{u}_i, \boldsymbol{v}_j, \boldsymbol{a}_i, \boldsymbol{b}_i, \boldsymbol{c}_j, \boldsymbol{d}_j\}$. We focus on inferring the posterior distributions of the latent variables ($\boldsymbol{U}$ and $\boldsymbol{V}$) in $\mathcal{Z}$ and find parameters $(\boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{\gamma}, \boldsymbol{\tau}, \boldsymbol{\beta}, \boldsymbol{\psi})$ to maximize the marginal distribution of observed variables ($\boldsymbol{R}$, $\boldsymbol{F}$ and $\boldsymbol{G}$) in (16).

However, it is difficult to infer latent variables in $\mathcal{Z}$ by using traditional mean-field approximation since we do not have any conjugate probability distribution in our model which requires by the traditional mean-field approach [30] and the integral in marginal distribution of observed variables ($\boldsymbol{R}$, $\boldsymbol{G}$ and $\boldsymbol{F}$) in (16) is also intractable. Inspired by VAE [19], we use Stochastic Gradient Variational Bayes

(SGVB) estimator to estimate posteriors of the latent variables related to user ($\boldsymbol{a}_i$, $\boldsymbol{b}_i$, $\boldsymbol{u}_i$) and latent variables related to item ($\boldsymbol{c}_j$, $\boldsymbol{d}_j$, $\boldsymbol{v}_j$) by introducing two inference networks, i.e., the user inference network and the item inference network (see Fig. 2), parameterized by ϕ and λ, respectively.

To do this, we first decompose the variational distribution q of latent variables in $\mathcal{Z}$ into two categories of variational distributions used in the two networks in our NVMF model — user inference network and item inference network (see Fig. 2), q_ϕ and q_λ, by assuming the conditional independence:

$$q(\mathcal{Z}_{ij}|\mathcal{X}_i, \mathcal{Y}_j, R_{ij}) = \underbrace{q_\phi(\boldsymbol{u}_i|\mathcal{X}_i)q_\phi(\boldsymbol{a}_i|\mathcal{X}_i)q_\phi(\boldsymbol{b}_i|\mathcal{X}_i)}_{\text{for users}} \cdot \underbrace{q_\lambda(\boldsymbol{v}_j|\mathcal{Y}_j)q_\lambda(\boldsymbol{c}_j|\mathcal{Y}_j)q_\lambda(\boldsymbol{d}_j|\mathcal{Y}_j)}_{\text{for items}}, \quad (17)$$

where $\mathcal{X}_i = (\boldsymbol{s}_i^u, \boldsymbol{f}_i)$ represents the set of user observed variables, and $\mathcal{Y}_j = (\boldsymbol{s}_j^v, \boldsymbol{g}_j)$ represents the set of item observed variables.

Like VAE [19], the variational distributions are chosen to be a Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, whose mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$ are the output of the inference network. Thus, in our NVMF, for latent variables related to the i-th user, we set:

$$q_\phi(\boldsymbol{u}_i|\mathcal{X}_i) = \mathcal{N}(\mu_{\phi_{u_i}}(\mathcal{X}_i), \text{diag}(\exp(\delta_{\phi_{u_i}}(\mathcal{X}_i)))), \quad (18)$$
$$q_\phi(\boldsymbol{a}_i|\mathcal{X}_i) = \mathcal{N}(\mu_{\phi_{a_i}}(\mathcal{X}_i), \text{diag}(\exp(\delta_{\phi_{a_i}}(\mathcal{X}_i)))), \quad (19)$$
$$q_\phi(\boldsymbol{b}_i|\mathcal{X}_i) = \mathcal{N}(\mu_{\phi_{b_i}}(\mathcal{X}_i), \text{diag}(\exp(\delta_{\phi_{b_i}}(\mathcal{X}_i)))), \quad (20)$$

where the subscripts of μ and δ indicate the parameters in our user inference network corresponding to $\boldsymbol{u}_i$, $\boldsymbol{a}_i$ and $\boldsymbol{b}_i$, respectively. Similarly, for j-th item:

$$q_\lambda(\boldsymbol{v}_j|\mathcal{Y}_j) = \mathcal{N}(\mu_{\lambda_{v_j}}(\mathcal{Y}_j), \text{diag}(\exp(\delta_{\lambda_{v_j}}(\mathcal{Y}_j)))), \quad (21)$$
$$q_\lambda(\boldsymbol{c}_j|\mathcal{Y}_j) = \mathcal{N}(\mu_{\lambda_{c_j}}(\mathcal{Y}_j), \text{diag}(\exp(\delta_{\lambda_{c_j}}(\mathcal{Y}_j)))), \quad (22)$$
$$q_\lambda(\boldsymbol{d}_j|\mathcal{Y}_j) = \mathcal{N}(\mu_{\lambda_{d_j}}(\mathcal{Y}_j), \text{diag}(\exp(\delta_{\lambda_{d_j}}(\mathcal{Y}_j)))), \quad (23)$$

where the subscripts of μ and δ indicate the parameters in item inference network corresponding to $\boldsymbol{v}_j$, $\boldsymbol{c}_j$ and $\boldsymbol{d}_j$, respectively.

Thus, the tractable standard evidence lower bound (ELBO) of variational distribution $q(\mathcal{Z}_{ij}|\mathcal{X}_i, \mathcal{Y}_j, R_{ij})$ for the inference can be computed as follows:

$$\begin{aligned}\mathcal{L}(q) &= \mathbb{E}_q[\log p(\mathcal{O}, \mathcal{Z}) - \log q(\mathcal{Z}|\mathcal{O})] \\ &= \sum_{i=1}^{M}\sum_{j=1}^{N}(\mathcal{L}_i(q_\phi) + \mathcal{L}_j(q_\lambda) + \mathbb{E}_q[\log p(R_{ij}|\boldsymbol{u}_i, \boldsymbol{v}_j)]), \quad (24)\end{aligned}$$

where $\mathcal{O} = (\boldsymbol{F}, \boldsymbol{G}, \boldsymbol{R})$ is a set of all observed variables. q_ϕ and q_λ are user term and item term in (17), respectively. For user i and item j, we have:

$$\begin{aligned}\mathcal{L}_i \quad (q_\phi) &= \mathcal{L}(\phi, \alpha, \theta, \gamma; \mathcal{X}_i) = \mathbb{E}_{q_\phi(\boldsymbol{a}_i, \boldsymbol{u}_i|\mathcal{X}_i)}[\log p_\alpha(\boldsymbol{s}_i^u|\boldsymbol{a}_i, \boldsymbol{u}_i)] \\ &+\mathbb{E}_{q_\phi(\boldsymbol{b}_i|\mathcal{X}_i)}[\log p_\theta(\boldsymbol{f}_i|\boldsymbol{b}_i)] - \text{KL}(q_\phi(\boldsymbol{a}_i|\mathcal{X}_i)||p(\boldsymbol{a}_i)) \quad (25) \\ &-\text{KL}(q_\phi(\boldsymbol{b}_i|\mathcal{X}_i)||p(\boldsymbol{b}_i)) - \omega_1 \text{KL}(q_\phi(\boldsymbol{u}_i|\mathcal{X}_i)||p_\gamma(\boldsymbol{u}_i|\boldsymbol{f}_i)),\end{aligned}$$

$$\begin{aligned}\mathcal{L}_j \quad (q_\lambda) &= \mathcal{L}(\lambda, \beta, \tau, \psi; \mathcal{Y}_j) = \mathbb{E}_{q_\lambda(\boldsymbol{c}_j, \boldsymbol{v}_j|\mathcal{Y}_j)}[\log p_\beta(\boldsymbol{s}_j^v|\boldsymbol{c}_j, \boldsymbol{v}_j)] \\ &+\mathbb{E}_{q_\lambda(\boldsymbol{d}_j|\mathcal{Y}_j)}[\log p_\tau(\boldsymbol{g}_j|\boldsymbol{d}_j)] - \text{KL}(q_\lambda(\boldsymbol{c}_j|\mathcal{Y}_j)||p(\boldsymbol{c}_j)) \quad (26) \\ &-\text{KL}(q_\lambda(\boldsymbol{d}_j|\mathcal{Y}_j)||p(\boldsymbol{d}_j)) - \omega_2 \text{KL}(q_\lambda(\boldsymbol{v}_j|\mathcal{Y}_j)||p_\psi(\boldsymbol{v}_j|\boldsymbol{g}_j)),\end{aligned}$$

where in the standard ELBO, the free parameters ω_1 and ω_2 are 1, and $\text{KL}(q_\phi(\boldsymbol{u}_i|\mathcal{X}_i)\ ||p_\gamma(\boldsymbol{u}_i|\boldsymbol{f}_i))$ is the Kullback-Leibler divergence between the approximate posterior distribution $q_\phi(\boldsymbol{u}_i|\mathcal{X}_i)$ and the prior $p_\gamma(\boldsymbol{u}_i|\boldsymbol{f}_i)$. The variational distribution $q_\phi(\boldsymbol{u}_i|\mathcal{X}_i)$ acts as an approximation to the true posterior $p(\boldsymbol{u}_i|\mathcal{O})$ when maximizing (25). Similarly, $q_\lambda(\boldsymbol{v}_j|\mathcal{Y}_j)$ acts as an approximation to the true posterior $p(v_j|\mathcal{O})$ when maximizing (26). And like discussed in VAE [19], maximizing the ELBO also means maximizing the marginal distribution of observed variables ($\mathcal{O}$).

Inspired by β-VAE and the previous work [23, 31], we use two free trade-off parameters ω_1 and ω_2 for the last terms in (25) and (26), respectively, in the ELBO to control the KL regularization instead of directly applying $\omega_1 = \omega_2 = 1$ (other KL terms in (25) and (26) do not need to apply the trade-off parameters as they are not related to our final latent factors $\boldsymbol{u}_i$ and $\boldsymbol{v}_j$). Since we assume the posteriors of all latent variables in $\mathcal{Z}$ are all Gaussian distribution, the KL terms in (25) and (26) have analytical forms. However, for the expectations terms in (25) and (26), we can not compute them analytically. To handle this problem, we use Monte Carlo method [20] to approximate the expectations by drawing samples from the posterior distribution of latent variables $\mathcal{Z}$. By using the reparameterization trick [20], the ELBO for user network is given:

$$\begin{aligned}\mathcal{L} \quad (\phi, \alpha, \theta, \gamma; \mathcal{X}_i) &\approx \frac{1}{K}\sum_{k=1}^{K}(\log p_\alpha(\boldsymbol{s}_i^u|\boldsymbol{a}_i^k, \boldsymbol{u}_i^k) + \log p_\theta(\boldsymbol{f}_i|\boldsymbol{b}_i^k)) \quad (27) \\ &-\text{KL}(q_\phi(\boldsymbol{a}_i|\mathcal{X}_i)||p(\boldsymbol{a}_i)) - \text{KL}(q_\phi(\boldsymbol{b}_i|\mathcal{X}_i)||p(\boldsymbol{b}_i)) \\ &-\omega_1 \text{KL}(q_\phi(\boldsymbol{u}_i|\mathcal{X}_i)||p_\gamma(\boldsymbol{u}_i|\boldsymbol{f}_i)),\end{aligned}$$

where K is the size of the samplings, $\boldsymbol{a}_i^k = \boldsymbol{\mu}_a + \boldsymbol{\delta}_a \odot \boldsymbol{\epsilon}_a^k$, $\boldsymbol{b}_i^k = \boldsymbol{\mu}_b + \boldsymbol{\delta}_b \odot \boldsymbol{\epsilon}_b^k$, $\boldsymbol{u}_i^k = \boldsymbol{\mu}_u + \boldsymbol{\delta}_u \odot \boldsymbol{\epsilon}_u^k$, $\odot$ is an element-wise multiplication and $\boldsymbol{\epsilon}_a^k$, $\boldsymbol{\epsilon}_b^k$, $\boldsymbol{\epsilon}_u^k$ are samples drawn from standard multivariate normal distribution $\mathcal{N}(0, \boldsymbol{I})$. The superscript k denotes the k-th sample. The ELBO for item

network, $\mathcal{L}(\lambda, \beta, \tau, \psi; \mathcal{Y}_j)$, can be derived similarly, and thus we omit it here.

Algorithm 1 Overview of the training for NVMF.

Input: user-item feedback matrix $\boldsymbol{R}$, user feature matrix $\boldsymbol{F}$, item feature matrix $\boldsymbol{G}$, parameters ω_1 and ω_2.

1: Randomly initialize $\phi, \alpha, \theta, \gamma, \lambda, \beta, \tau, \psi$.
2: **while** not converged **do**
3: Sample a R_{ij} from $\boldsymbol{R}$.
4: For R_{ij}, extracting the i-th user's features $\boldsymbol{f}_i$, and the j-th item's features $\boldsymbol{g}_j$, the i-th user's and the j-th item's collaborative information, $\boldsymbol{s}_i^u$ and $\boldsymbol{s}_j^v$.
5: Compute the KL terms in (25) and (26), the matrix factorization loss (29) by the output of user and item inference networks.
6: For every expectation terms in (25) and (26), sample ϵ from standard multivariate normal distribution and compute them via reparameterization trick.
7: Update $\phi, \alpha, \theta, \gamma, \lambda, \beta, \tau, \psi$ by back-propagation.
8: **end while**
9: return $\phi, \alpha, \theta, \gamma, \lambda, \beta, \tau, \psi$.

5.2 Optimizing parameters

Since minimizing the objection function is equivalent to maximizing the log likelihood of the observed data. Based on $\mathcal{L}(\phi, \alpha, \theta, \gamma; \mathcal{X}_i)$ in (28) and $\mathcal{L}(\lambda, \beta, \tau, \psi; \mathcal{Y}_j)$, the objective function is:

$$\mathcal{L} = -\sum_{i=1}^{M}\sum_{j=1}^{N}(\mathcal{L}(\phi, \alpha, \theta, \gamma; \mathcal{X}_i) + \mathcal{L}(\lambda, \beta, \tau, \psi; \mathcal{Y}_j) + \frac{C_{ij}}{2}\mathbb{E}_{q_\phi(\boldsymbol{u}_i|\mathcal{X}_i)q_\lambda(\boldsymbol{v}_j|\mathcal{Y}_j)}[(R_{ij} - \boldsymbol{u}_i^\top \boldsymbol{v}_j)^2]), \quad (28)$$

where the expectation term is given by:

$$\mathbb{E}_{q_\phi(\boldsymbol{u}_i|\mathcal{X}_i)q_\lambda(\boldsymbol{v}_j|\mathcal{Y}_j)}[(R_{ij} - \boldsymbol{u}_i^\top \boldsymbol{v}_j)^2] = R_{ij}^2 - 2R_{ij}\mathbb{E}[\boldsymbol{u}_i]^\top \mathbb{E}[\boldsymbol{v}_j] + \mathrm{tr}((\mathbb{E}[\boldsymbol{v}_j]\mathbb{E}[\boldsymbol{v}_j]^\top + \Sigma_v)\Sigma_u) + \mathbb{E}[\boldsymbol{u}_i]^\top(\mathbb{E}[\boldsymbol{v}_j]\mathbb{E}[\boldsymbol{v}_j]^\top + \Sigma_v)\mathbb{E}[\boldsymbol{u}_i], \quad (29)$$

where $\mathrm{tr}(\cdot)$ denotes the trace of a matrix. Since NVMF is a fully end-to-end neural network, the whole parameters of the model are the weight matrix of entire network, we can use back-propagation algorithm to optimize the weights of the user network and the item network.

5.3 Prediction of ratings

After the training is converged, we can get the posterior distributions of $\boldsymbol{u}_i$ and $\boldsymbol{v}_j$ through the user and item inference networks, respectively. So the prediction R_{ij} can be made by:

$$\mathbb{E}[R_{ij}|\mathcal{O}] = \mathbb{E}[\boldsymbol{u}_i|\mathcal{O}]^\top \mathbb{E}[\boldsymbol{v}_j|\mathcal{O}]. \quad (30)$$

Specifically, given a user, an item and the corresponding observed collaborative information and the features, $\boldsymbol{s}_i^u$, $\boldsymbol{f}_i$, $\boldsymbol{s}_j^v$ and $\boldsymbol{g}_j$, our NVMF makes the rating prediction for the user on the item, R_{ij}, as $R_{ij} = \mathbb{E}[R_{ij}|\mathcal{O}]$ with $\mathbb{E}[\boldsymbol{u}_i|\mathcal{O}] = \mu_{\phi_{u_i}}(\mathcal{X}_i)$ (see (18)) and $\mathbb{E}[\boldsymbol{v}_j|\mathcal{O}] = \mu_{\phi_{v_j}}(\mathcal{Y}_j)$ (see (21)), respectively. For ranking prediction, we rank a list of items based on these prediction ratings.

5.4 Discussion

The computational complexity of training NVMF in each iteration is $O(2(N+P)L + PL + 2(M+Q)L + Q \cdot L + J \cdot L^2 + D^2)$, where J is the total number of layers in user and item networks, L is the average dimensions of these layers, $O(2(N+P)L)$, $O(PL)$, $O(2(M+Q)L)$, $O(QL)$, $O(J \cdot L^2)$ and $O(D^2)$ are the complexities of the encoder input layer and the decoder output layer in the user network, the input layer in the user prior network, the encoder input layer and the decoder output layer in the item network, the input layer in the item prior network, all the latent layers in NVMF, and the matrix factorization, respectively. Thus, the complexity of proposed NVMF is at the same level as the previous recommendation methods [15, 16].

6 Performance evaluation

We first give our experimental environment settings and then present the results of performance evaluation of experiments.

6.1 Experimental setting

6.1.1 Dataset

We use three benchmark datasets in our experiments which are commonly used to previous recommendation algorithms.

- **MovieLens-100K**[1] (ML-100K) : Similar to [15, 18], we extract the features of users and movies provided by the dataset to construct our side information matrices F and G respectively. The user's feature contains user's id, age, gender, occupation and zipcode, corresponding to the movie's feature contains movie's title, release data and 18 categories of movie genre.

[1] https://grouplens.org/datasets/movielens/100k/

- **MovieLens-1M**[2] (ML-1M): Similar to ML-100K, we can get side information of users and items.
- **Bookcrossing**[3]: For this dataset, we also extract the features of users and book provided by the dataset. We encoded the user and book feature into binary vector of length 30 and 30 respectively. Since we will evaluate our model performance on implicit feedback. Thus, following to [15, 18], we interpret three datasets above as implicit feedback.

6.1.2 Baselines and experimental settings

For implicit feedback, as demonstrated in [16, 22], the hybrid collaborative filtering model incorporating side information outperforms the other method without side information. So the most baselines we choose are hybrid models. The baselines we use to compare our proposed method are listed as follows:

1) **CMF** [32]: This model is a MF model which decomposes the user-item matrix $\boldsymbol{R}$, user's side information matrix $\boldsymbol{F}$, and item's side information matrix $\boldsymbol{G}$ to get the consistent latent factor of user and item.
2) **CDL** [16]: This model is a hierarchical Bayesian model for joint learning of stacked denoising autoencoder and collaborative filtering.
3) **CVAE** [22]: This model is a Bayesian probabilistic model which unifies item feature and collaborative filtering through stochastic deep learning and probabilistic model.
4) **aSDAE** [18]: This model is a deep learning model which can integrate the side information into matrix factorization via additional stack denoising autoencoder.
5) **NeuMF** [33]: This model is a state-of-the-art collaborative filtering method for implicit feedback.

Since the implicit feedback matrix $\boldsymbol{R} \in \{0, 1\}^{M \times N}$, we set $p_\theta(\boldsymbol{f}_i|\boldsymbol{b}_i)$ and $p_\theta(\boldsymbol{g}_j|\boldsymbol{c}_j)$ as multivariate Bernoulli distribution. We set ω_1, ω_2 and the learning rate η to 0.05, 0.05 and 0.0001. The value of φ_1, φ_2 and λ_w are set to 1, 0.01 and 0.0001, respectively. The dimensions K_a, K_b, K_c, K_d are all chosen to be 20. The inference and generative networks are both two-layer network architectures and the last layer of generative network is a softmax layer. Similar to explicit feedback, the prior network of user and item are both set to one layer. We also set the dimensions of use and item latent factor D as 30. We use 80% ratings of dataset to train each method, 10% as validation, and the remaining 10% for testing. We repeat this procedure five times and report the average performance.

[2] https://grouplens.org/datasets/movielens/1m/

[3] https://grouplens.org/datasets/book-crossing/

6.1.3 Evaluation metrics

For evaluation, we use the Hit Ratio (HR) [34] and the Normalized Discounted Cumulative Gain (NDCG) [33] as our evaluation as metrics. For each user, we sort the top-K items based on the predicted ratings.

$$\text{HR@}K = \frac{\#\text{hits@}K}{|\text{GT}|}, \qquad \text{NDCG@}K = Z_k \sum_{i=1}^{K} \frac{2^{\text{rel}_i} - 1}{\log_2(i+1)}, \quad (31)$$

where GT denotes the Ground Truth in test list set, rel_i is the graded relevance value of the item at position i and Z_k is the normalization. For top-K recommendation, $\text{rel}_i \in \{0, 1\}$. We report the average recall scores over all users in our experimental analysis.

6.2 Results and analysis

We now show the results of performance comparisons between our proposed model NVMF and baselines on different experimental settings.

6.2.1 Performance comparison with baselines

Table 2 shows the experiment result that compare CMF, CDL, CVAE, aSDAE and NeuMF using three datasets. As we can see, our proposed methods NVMF significantly outperform the compared methods in all cases on both ML-100K, ML-1M and BookCrossing datasets. Compared with other methods that have a deep learning structure, CMF achieves the worst performance. This demonstrates the deep learning structure can learn more subtle and complex representation than the tradition MF method. We also observe that although CDL and CVAE both have a deep learning structure, CVAE achieves better performance than CDL. This is because CDL is based on denoising autoencoder which can be seen as point estimation, however CVAE is fully deep probabilistic model which make it hard to overfit the data. From Table 2, we observe the strongest baseline in our experiment is aSDAE which outperforms the other baselines. Although aSDAE is not a probabilistic model, it incorporates user's side information (user's features) into matrix factorization which CDL and CVAE does not. By incorporating both user's feature and item's feature and applying deep generative model, our model NVMF outperform all the baselines. Specifically, the average improvements of NVMF over the state-of-the-art method. Compared to other datasets, our model has the greatest performance improvement on Bookcrossing which is most sparse matrix among three datasets. This shows NVMF can more effectively alleviate the sparsity problem on implicit feedback than aSDAE.

Table 2 Recommendation performance comparison between our NVMF and baselines

Datasets	Metrics	CMF	CDL	CVAE	aSDAE	NeuMF	NVMF(ours)
ML-100K	HR@5	0.4121	0.4564	0.4721	0.4981	0.4942	0.5083
	NDCG@5	0.2124	0.2991	0.3012	0.3156	0.3351	0.3417
	HR@10	0.5587	0.6123	0.6421	0.6871	0.6692	0.6982
	NDCG@10	0.3387	0.3654	0.3871	0.4231	0.4103	0.4358
ML-1M	HR@5	0.4237	0.5011	0.5141	0.5411	0.5211	0.5681
	NDCG@5	0.2578	0.3362	0.3621	0.4124	0.4011	0.4325
	HR@10	0.5921	0.6557	0.6874	0.7321	0.7202	0.7412
	NDCG@10	0.3328	0.3547	0.3864	0.4121	0.4025	0.4205
Bookcrossing	HR@5	0.1565	0.1714	0.1921	0.2234	0.2123	0.2354
	NDCG@5	0.0523	0.0717	0.0921	0.1121	0.1024	0.1244
	HR@10	0.2347	0.2654	0.2876	0.3097	0.3012	0.3142
	NDCG@10	0.1024	0.1451	0.1612	0.1911	0.1876	0.2087

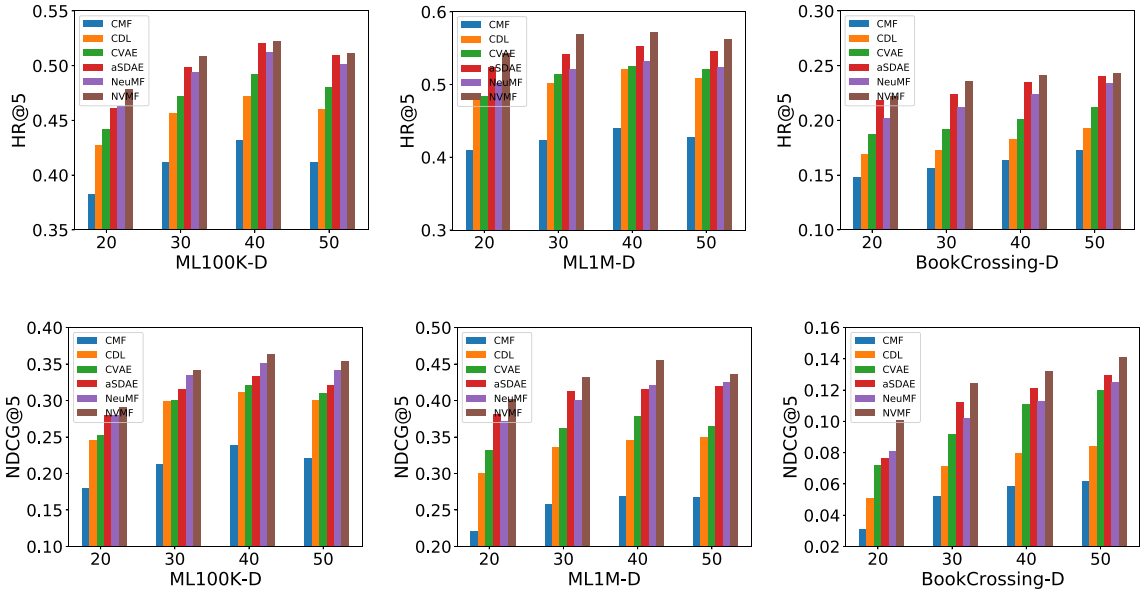

Fig. 3 Performance comparison between NVMF and the baselines by varying latent dimension D on three datasets

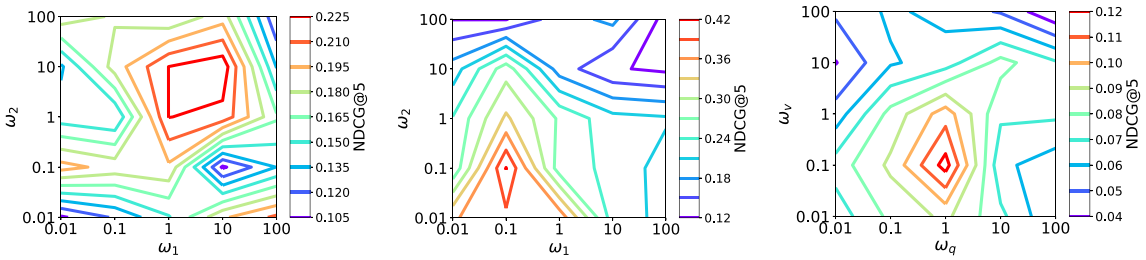

Fig. 4 The NDCG@5 for NVMF by varying ω_1 and ω_2 on three datasets

Table 3 Recommendation performance comparison with different percentages of training dat on ML-100K

Datasets	Percentages	Metrics	CMF	CDL	CVAE	aSDAE	NeuMF	NVMF(ours)
ML-100K	30%	HR@5	0.2125	0.2367	0.2431	0.2679	0.2565	0.2768
		NDCG@5	0.0834	0.1012	0.1123	0.1335	0.1231	0.1621
	50%	HR@5	0.3132	0. 3421	0.3517	0.3865	0.3721	0.3921
		NDCG@5	0.1452	0.1617	0.1762	0.1891	0.1832	0.2076
	80%	HR@5	0.4121	0.4564	0.4721	0.4981	0.4942	0.5083
		NDCG@5	0.2124	0.2991	0.3012	0.3156	0.3351	0.3417

6.2.2 Performance comparison on different model configurations

Figure 3 shows performance comparison in term of different dimension D. We can observe that larger dimension leads to better performance. According to Fig. 3 , our NVMF outperforms other baselines on different dimensions. We also can find our NVMF achieve best performance when D=40 on ML100K and ML100M datasets, D=50 on BookCrossing. Figure 4 shows the contours of NDCG@5 for NVMF on three datasets. When the parameters equals 1, i.e $\omega_1 = 1$ and $\omega_2 = 1$, it means we directly optimize the standard ELBO -which has a degraded performance and this confirmed in Fig. 4b. When we decrease ω_1 to 0.1 at fixed ω_2, we find NVMF's performance improves(small ω_1 implies we want condense more user's collaborative information into user's latent factor). Similar observation can be made for varying ω_2 at fixed ω_1. Moreover, It can be observed that there is a region of values of ω_1 and ω_2 (near (0.1,0.1)), around which NVMF provides the best performance in terms of recall. Altogether, Fig. 4 shows treating ω_1 and ω_2 as trade-off parameters can yields significant improvement in performance of the recommendation.

6.3 Performance comparison on different sparsity conditions

To further evaluate our proposed NVMF on different sparsity conditions, we train each compared baseline method with different percentages (30%, 50%, 80%) of feedbacks on ML-100K datasets. Note that we do not conduct the experiment on ML-1M and BoorCrossing Datasets, since they are already very sparse (95.8% and 99.9% sparsity respectively). Table 3 shows the results. From Table 3, we can find: 1) Our proposed model outperforms all baselines on all sparsity conditions, which demonstrates our model can effectively handle the data sparsity problem. 2) The models which incorporate both uses' and items' side information (i.e., NVMF and aSDAE) outperform others (i.e., CDL and CVAE), which demonstrates again that incorporating both users' and items' side information can effectively handle data sparsity problem. 3) Our model NVMF outperforms many baselines (i.e., aSDAE and NeuMF) at extreme sparsity condition (30%) by a large margin.

7 Conclusion

In this paper, we study the problem of how to learn subtle and complex latent factors from the feedback matrix with side information for collaborative filtering in recommendation systems. We propose a neural variational matrix factorization model NVMF which is a novel deep generative model to learn the latent factors of users and items. NVMF possesses the merits of both deep learning and probabilistic generative models. It can learn more subtle and complex representations than traditional probabilistic matrix factorization, and is more robust than other deep neural network models such as autoencoder (AE) and denoising autoencoder (DAE). In addition, NVMF incorporates the features of users and items into matrix factorization through a novel generative process, which enables it to effectively handle the data sparsity and cold start problems. We present a complete end-to-end network architecture so that back-propagation can be applied for efficient parameter estimation, and a Stochastic Gradient Variational Bayes (SGVB) estimator to infer latent factors and parameters in our model. We also derive a variational lower bound to show the quality guarantee of the model. Experiments conducted on explicit and implicit (user-item rating) feedbacks have demonstrated our model's effectiveness in learning the latent factors and its outperformance on recommendation accuracy over the state-of-the-art methods for recommendation.

The complete architecture of model generation and parameter inference makes the proposed model easy to be extended to handle other types of data such as images and videos.

Acknowledgements This work is supported by National Key R & D Program of China Project #2017YFB0203201 and Australian Research Council Discovery Project DP150104871. The corresponding author is Hong Shen.

References

1. Zhang S, Yao L, Sun A Deep learning based recommender system: a survey and new perspectives
2. Koren Y, Bell R, Volinsky C Matrix factorization techniques for recommender systems. Computer
3. Lee DD, Seung HS (2001) Algorithms for non-negative matrix factorization. In: NIPS, pp 556–562
4. Xu M, Zhu J, Zhang B (2013) Fast max-margin matrix factorization with data augmentation. In: ICML, pp 978–986
5. Salakhutdinov R, Mnih A (2008) Bayesian probabilistic matrix factorization using Markov chain Monte Carlo. In: ICML, pp 880–887
6. Adams RP, Dahl GE, Murray I Incorporating side information in probabilistic matrix factorization with gaussian processes. arXiv:1003.4944
7. Park S, Kim Y-D, Choi S (2013) Hierarchical Bayesian matrix factorization with side information. In: IJCAI, pp 1593–1599
8. Kim YD, Choi S (2014) Scalable variational Bayesian matrix factorization with side information, 493–502
9. Porteous I, Asuncion A, Welling M (2010) Bayesian matrix factorization with side information and Dirichlet process mixtures. In: AAAI, pp 563–568
10. Singh A, Gordon GJ (2010) A Bayesian matrix factorization model for relational data. In: UAI, pp 556–563
11. Mikolov T, Sutskever I, Chen K, Corrado GS, Dean J (2013) Distributed representations of words and phrases and their compositionality. In: NIPS, pp 3111–3119
12. He K, Gkioxari G, Dollar P, Girshick R (2017) Mask r-cnn. In: ICCV, pp 2980–2988
13. Hinton G, Deng L, Yu D, Dahl GE, Mohamed A, Jaitly N, Senior A, Vanhoucke V, Nguyen P, Sainath TN (2012) Deep neural networks for acoustic modeling in speech recognition: the shared views of four research groups. IEEE Signal Process Mag 29(6):82–97
14. Graves A, Jaitly N (2014) Towards end-to-end speech recognition with recurrent neural networks. In: ICML, pp 1764–1772
15. Li S, Kawale J, Fu Y (2015) Deep collaborative filtering via marginalized denoising auto-encoder. In: CIKM, pp 811–820
16. Wang H, Wang N, Yeung DY (2014) Collaborative deep learning for recommender systems, 1235–1244
17. Strub F, Gaudel R, Mar J (2016) Hybrid recommender system based on autoencoders. In: Proceedings of the 1st workshop on deep learning for recommender systems. ACM, pp 11–16
18. Dong X, Yu L, Wu Z, Sun Y, Yuan L, Zhang F (2017) A hybrid collaborative filtering model with deep structure for recommender systems. In: AAAI, pp 1309–1315
19. Kingma DP, Welling M Auto-encoding variational Bayes. arXiv:1312.6114
20. Rezende DJ, Mohamed S, Wierstra D Stochastic backpropagation and approximate inference in deep generative models. arXiv:1401.4082
21. Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, Bengio Y (2014) Generative adversarial nets. In: NIPS, pp 2672–2680
22. Li X, She J (2017) Collaborative variational autoencoder for recommender systems. In: KDD, pp 305–314
23. Liang D, Krishnan RG, Hoffman MD, Jebara T Variational autoencoders for collaborative filtering. arXiv:1802.05814
24. Chen Y, de Rijke M (2018) A collective variational autoencoder for top-n recommendation with side information. In: Proceedings of the 3rd workshop on deep learning for recommender systems. ACM, pp 3–9
25. Bowman SR, Vilnis L, Vinyals O, Dai A, Jozefowicz R, Bengio S (2016) Generating sentences from a continuous space. In: Proceedings of the 20th SIGNLL conference on computational natural language learning, pp 10–21
26. Mnih A, Salakhutdinov RR (2008) Probabilistic matrix factorization. In: NIPS, pp 1257–1264
27. Wang C, Blei DM (2011) Collaborative topic modeling for recommending scientific articles. In: KDD, pp 448–456
28. Doersch C Tutorial on variational autoencoders
29. Agarwal D, Chen BC (2009) Regression-based latent factor models. In: KDD, pp 19–28
30. Wainwright MJ, Jordan MI et al (2008) Graphical models, exponential families, and variational inference. Found Trends® Mach Learn 1(1–2):1–305
31. Higgins I, Matthey L, Pal A, Burgess C, Glorot X, Botvinick M, Mohamed S, Lerchner A beta-vae: learning basic visual concepts with a constrained variational framework
32. Singh AP, Gordon GJ (2008) Relational learning via collective matrix factorization. In: KDD, pp 650–658
33. He X, Liao L, Zhang H, Nie L, Hu X, Chua T-S (2017) Neural collaborative filtering. In: WWW, pp 173–182
34. Liang D, Charlin L, McInerney J, Blei DM (2016) Modeling user exposure in recommendation. In: Proceedings of the 25th international conference on World Wide Web. International World Wide Web Conferences Steering Committee, pp 951–961
35. Xiao T, Shen H (2019) Neural Variational Matrix Factorization with Side Information for Collaborative Filtering. In: PAKDD 2019 (to appear)

Teng Xiao received Bachelor of Engineering degree in Internet of Things form Jiangsu University. He is currently graduate student in the School of Data and Computer Science, Sun Yat-sen University, China. His research interests include deep learning, social networks and applied intelligence.

Hong Shen is s currently a specially-appointed professor at Sun Yat-sen University, China, and a tenured Professor (Chair) of Computer Science at University of Adelaide, Australia. He received the B.Eng. degree from Beijing University of Science and Technology, M.Eng. degree from University of Science and Technology of China, Ph.Lic. and Ph.D. degrees from Abo Akademi University, Finland, all in Computer Science. He was Professor and Chair of the Computer Networks Laboratory in Japan Advanced Institute of Science and Technology (JAIST) during 2001-2006, and Professor (Chair) of Compute Science at Griffith University, Australia, where he taught 9 years since 1992. With main research interests in parallel and distributed computing, algorithms, data mining, privacy preserving computing, high performance networks and multimedia systems, he has published more than 300 papers including over 100 papers in international journals such as a variety of IEEE and ACM transactions. Prof. Shen received many honours and awards.