Deep Global and Local Generative Model for Recommendation

Huafeng Liu, Liping Jing, Jingxuan Wen, Zhicheng Wu, Xiaoyi Sun, Jiaqi Wang, Lin Xiao, Jian Yu
Beijing Key Lab of Traffic Data Analysis and Mining
Beijing Jiaotong University
Beijing, China
{huafeng,lpjing,jingxuan,wuzhicheng,xiaoyisun,jiaqi.wang,xiaolin,jianyu}@bjtu.edu.cn

ABSTRACT

Deep generative model, especially variational auto-encoder (VAE), has been successfully employed by more and more recommendation systems. The reason is that it combines the flexibility of probabilistic generative model with the powerful non-linear feature representation ability of deep neural networks. The existing VAE-based recommendation models are usually proposed under global assumption by incorporating simple priors, e.g., a single *Gaussian*, to regularize the latent variables. This strategy, however, is ineffective when the user is simultaneously interested in different kinds of items, i.e., the user's preference may be highly diverse. In this paper, thus, we propose a Deep Global and Local Generative Model for recommendation to consider both local and global structure among users (**DGLGM**) under the Wasserstein auto-encoder framework. Besides keeping the global structure like the existing model, **DGLGM** adopts a non-parametric *Mixture Gaussian* distribution with several components to capture the diversity of the users' preferences. Each component is corresponding to one local structure and its optimal size can be determined via the automatic relevance determination technique. These two parts can be seamlessly integrated and enhance each other. The proposed **DGLGM** can be efficiently inferred by minimizing its penalized upper bound with the aid of local variational optimization technique. Meanwhile, we theoretically analyze its generalization error bounds to guarantee its performance in sparse feedback data with diversity. By comparing with the state-of-the-art methods, the experimental results demonstrate that **DGLGM** consistently benefits the recommendation system in top-N recommendation task.

CCS CONCEPTS

• **Information systems** → **Recommender systems**; **Clustering and classification**.

KEYWORDS

Personalized Recommendation, Deep Generative Model, Bayesian Graphical Model

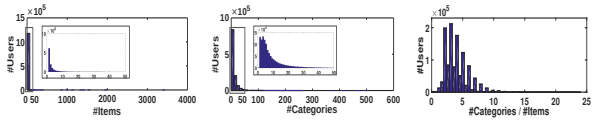
ACM Reference Format:
Huafeng Liu, Liping Jing, Jingxuan Wen, Zhicheng Wu, Xiaoyi Sun, Jiaqi Wang, Lin Xiao, Jian Yu. 2020. Deep Global and Local Generative Model for Recommendation. In *Proceedings of The Web Conference 2020 (WWW '20), April 20–24, 2020, Taipei, Taiwan.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3366423.3380138

WWW '20, April 20–24, 2020, Taipei, Taiwan

ACM ISBN 978-1-4503-7023-3/20/04.
https://doi.org/10.1145/3366423.3380138

1 INTRODUCTION

A common task of recommendation systems is to improve customer experience by mining knowledge from prior implicit or explicit feedback. The system will be more satisfied if it can provide personalized recommendation. It aims to predict a personalized ranking on a set of items that users may like the most. In real-world scenarios, feedback data is usually limited because of the limited user responses and a large scale of users and items. Meanwhile, one user might be interested in different kinds of items, especially when the system contains a large set of items with various types. Thus, it is a challenging issue to accurately capture the personalized characteristics for each user.



(a) # items (b) #categories (c) category ratio

Figure 1: Data distribution in *Yelp* dataset.

Taking *Yelp*[1] dataset as an example, all *Yelp* businesses belong to 1240 categories. Figure 1 (a) shows the distribution about the number of rated items along all users. Most users only rated few items, which leads to a power-law distribution. This statistical result confirms that the feedback data is extremely sparse. Figure 1 (b) demonstrates the size of categories corresponding to items which are rated by each user. It can be found that most of the users are interested in more than one categories. To further demonstrate that users have diverse interests, we calculate the ratio between category size and item size for each user, and its distribution along all users is shown in Figure 1 (c). Although a large number of users enrolled few items and a small number of categories, the corresponding ratio is large, which indicates that the diversity of user interests exists in the preference data.

To capture the diversity of user preferences, recently, researchers proposed several local matrix approximation methods [18, 35, 36]. The main idea is to divide the whole rating matrix into several submatrices so that each submatrix contains a set of like-minded users and the items that they are interested in. Then, the latent user/item factors are determined by applying low-rank matrix approximation technique [16, 26] on each submatrix, which are used to estimate the missing rates. Unfortunately, the limited representation ability of matrix approximation hinders these methods to achieve a comparably good result. To be excited, deep neural networks have been revolutionizing the recommendation architectures

[1] https://www.yelp.com/dataset/challenge

dramatically and bring more opportunities in reinventing the user experiences for better customer satisfaction. One typical strategy is adding the non-linear deep features into the probabilistic generative models [11, 19, 20] for collaborative filtering to estimate the unseen and complex distribution of recommendation data. However, those deep generative models (e.g., with *Multinomial* distribution) are originally proposed for dense data [15, 17], which may lead to overfitting or be stuck in a poor local optimal for sparse recommendation data. Furthermore, they use one prior distribution (e.g., *standard Gaussian*) to model the latent variables for all users, which can not sufficiently characterize the diversity of user preferences.

In this paper, thus, we focus on recommendation task containing sparse feedback data with diverse user preferences. A new Deep Global and Local Generative Model (**DGLGM**) is proposed to characterize both global and local structure among users. On the one hand, **DGLGM** introduces *Beta-Bernoulli* distribution as the likelihood to model the implicit feedback of all users, so that the global structure can be effectively determined from the sparse recommendation data. On the other hand, **DGLGM** introduces *Mixture Gaussian* distribution as the prior of the latent variables and is comprised of several *Beta-Bernoulli* components to sufficiently characterize the diversity of user preferences, where each component corresponds to one subset of users who have similar preferences). These two parts are seamlessly integrated together under the *Wasserstein* auto-encoder framework, so that they can enhance each other. **DGLGM** can be efficiently inferred via the local variational optimization technique. Moreover, we theoretically analyze the generalization error bounds of the proposed method, which is able to guarantee the performance on sparse feedback data with diversity. A series of experiments are conducted on four real-world datasets (*ML 20M*, *Netflix*, *Epinions*, and *Yelp*), and the results have demonstrated that **DGLGM** outperforms the state-of-the-art baselines in terms of several popular evaluation metrics.

2 PROBLEM FORMULATION

Let $\mathbf{X} \in \mathbb{N}^{n \times m}$ denote the binary implicit feedback among n users and m items. Its element $\mathbf{x}_{ij} \in \{0, 1\}$ indicates whether the j-th item is interacted by the i-th user. $\mathbf{x}_i = [\mathbf{x}_{i1}, \cdots, \mathbf{x}_{im}]^\top \in \mathbb{N}^m$ is a binary vector demonstrating the click history of user i on all items. The general goal of deep generative recommendation method is to determine the latent representation $\mathbf{z}_i \in \mathbb{R}^d$ by defining some generative process from latent representation to preference feedback data, *i.e.*, $p(\mathbf{x}_i|\mathbf{z}_i)$. The dimensionality d, in literatures, is usually fixed with a small value ($d \ll \min\{n, m\}$).

Table 1: The Recall@10 and NDCG@10 of Mult-VAE for users (*Yelp* dataset) with different numbers of feedback when latent dimensionality d is 50 and 150. $N(i)$ is the set of items user i rated.

d	50		150	
Metrics	Recall@10	NDCG@10	Recall@10	NDCG@10
$\|N(i)\| < 5$	0.5432	0.7201	0.5361	0.7123
$\|N(i)\| > 50$	0.5659	0.7431	0.5889	0.7763
All	0.5566	0.7246	0.5824	0.7651

In real-world recommendation scenarios, the user preferences are always diverse. For example, *Movielens* and *Yelp*, the number of interacted items are different among users, and the items belong to a large set of categories. However, the user representation vector with a low-dimensional latent space will be a bottleneck to express user's diverse interests. To demonstrate this point, a case study is conducted on the *Yelp* dataset. The state-of-the-art method Multi-VAE [19] (a variational auto-encoder recommendation with *Multinomial* likelihood) is adopted to handle the data with different latent space sizes. Table 1 lists the recommendation performances under two cases ($d = 50$ and $d = 150$). We focus on two subsets of users, one group with less than 5 items and the other with more than 50 items. The results show that Multi-VAE with $d = 150$ can not work well for users with less than 5 feedback. The main reason, we believe, is that the complex model (including much more parameters) may result in overfitting problem. For the overall users, although Multi-VAE with $d = 150$ achieves higher accuracy than Multi-VAE with $d = 50$, the improvement is obtained by destroying the performance on the near-cold-start users. Unfortunately, such set of near-cold-start users is usually the key group in real-world recommendation system. This study demonstrates that deep generative recommendation model with fixed latent dimensionality cannot perfectly model the internal mixture-preference structure of user preferences. Thus, it is desirable to model users preference by considering their diverse interests.

3 THE PROPOSED MODEL

To address the challenges mentioned previously, the main aim of this work is to actually capture the personalized property and simultaneously cover the diversity of user preferences. In this section, we first present a deep global generative recommendation model (**DGGM**) to model the user's personalized property from the whole implicit feedback data. Then we extend it by introducing a deep local generative process to capture the diverse user preferences. Thus, the whole model is called as deep global and local generative recommendation model (**DGLGM**), as shown in Figure 2 To efficiently infer the proposed deep generative model, a local variational optimization algorithm is designed. Meanwhile, we formally define a generalization of **DGLGM** model and theoretically demonstrate it has a bounded generalization error.

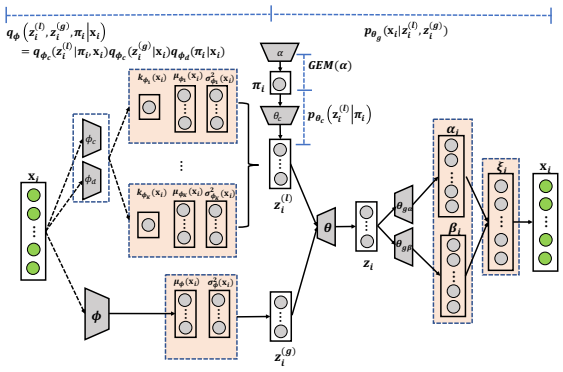


Figure 2: Architecture of the proposed Deep Global and Local Generative recommendation Model (DGLGM).

3.1 Global Model (DGGM)

To sufficiently capture the personalized property for each user, we adopt the auto-encoder structure to learn the user's latent representation. Mathematically, it can be described as follows. The feedback data $\mathbf{x}_i \in \mathbb{N}^m$ for user i is reconstructed by a latent vector $\mathbf{z}_i \in \mathbb{R}^d$. For convenient computing, $\mathbf{z}_i$ is assumed to be sampled from a standard *Gaussian* prior, i.e., $\mathbf{z}_i \sim \mathcal{N}(0, \mathbf{I}_d)$. Simultaneously, a vector $\boldsymbol{\xi}_i \in \mathbb{R}^m$ is introduced to characterize the extent to which the user i likes each item. To properly model the preference of each user i (in the range of $[0, 1]$) to the corresponding items, the *Beta* distribution is adopted to sample $\boldsymbol{\xi}_i$. Among it, there are two positive shape parameters, $\boldsymbol{\alpha}_i \in \mathbb{R}_+^m$ and $\boldsymbol{\beta}_i \in \mathbb{R}_+^m$ over m items, which are produced via two non-linear functions $f_{\theta_\alpha}(\cdot)$ and $f_{\theta_\beta}(\cdot)$ from $\mathbf{z}_i$. In this work, we use a three-layer neural networks activated by *tanh* function as the non-linear function and transfer the output via *ReLu* function. Since the feedback data $\mathbf{x}_i$ is binary, the *Bernoulli* distribution is used to generate it according to the strength $\boldsymbol{\xi}_i$. Then, the whole generative process can be summarized as follows.

$$\begin{aligned} &\mathbf{z}_i^{(g)} \sim \mathcal{N}(0, \mathbf{I}_d), \qquad \boldsymbol{\alpha}_i = f_{\theta_\alpha}(\mathbf{z}_i) \text{ and } \boldsymbol{\beta}_i = f_{\theta_\beta}(\mathbf{z}_i) \\ &\boldsymbol{\xi}_i \sim \prod_{j=1}^{m} \mathcal{B}eta(\boldsymbol{\alpha}_{ij}, \boldsymbol{\beta}_{ij}), \qquad \mathbf{x}_i \sim \prod_{j=1}^{m} \mathcal{B}(\boldsymbol{\xi}_{ij}) \end{aligned} \tag{1}$$

In other words, the implicit feedback data can be generated under the *Beta-Bernoulli* distribution. In this case, the global generative model can be formulated as:

$$p_G^{(g)}(\mathbf{x}_i) = \int_{\mathbf{z}_i^{(g)}} p_\theta(\mathbf{x}_i|\mathbf{z}_i^{(g)}) p(\mathbf{z}_i^{(g)}) d\mathbf{z}_i^{(g)} \tag{2}$$

where the global latent variable $\mathbf{z}_i^{(g)}$ has the ability to capture the associations among all users and items.

According to the property of *Bernoulli* distribution and *Beta* distribution, we can generate sparse or dense feedback data by setting the range of α_i and β_i. For example, when α_i and β_i are greater than 1 and α_i is less than β_i, $\boldsymbol{\xi}_i$ will be small and then the generated $\mathbf{x}_i$ will be sparse. User i only has one latent representation $\mathbf{z}_i^{(g)}$ over all items and the preference score $\boldsymbol{\xi}_{ij}$ of user i on item j is independently generated. Although the proposed model may put the preference mass on the non-zero entries of $\mathbf{x}_i$, it simultaneously considers the non-observed information. This strategy is consistent with the assumption of *missing not at random (MNAR)* [21]. Thus, it is expected to be more suitable to generate sparse implicit feedback data.

3.2 Global and Local Model (DGLGM)

Mixture model has proved to be a universal approximator for any continuous density function, which could be exploited to better characterize the data from different spaces using multiple probability distributions. This idea has recently been pursued for recommendation [7, 32] and obtains good performance. Nevertheless, they have to predefine the number of mixture components (K). This hyper-parameter definitely affects the final recommendation performance. If K is too small, the mixture model may not be able to well capture the complicated local structure from wide-ranging sources. If K is too large, however, it will be time-consuming to learn all components even though most of them may make few contribution. Moreover, without a proper prior assumption for the mixing coefficient, the mixture model may be unstable and result in overfitting [7].

To capture the diverse users' preferences, we propose a deep mixture generative recommendation model. In this case, the feedback data of each user is generated by a mixture *Gaussian* distribution.

$$p_G^{(l)}(\mathbf{x}_i) = \sum_{k=1}^{K} \boldsymbol{\pi}_{ik} \int_{\mathbf{z}_i^{(l)}} p_\theta(\mathbf{x}_i|\mathbf{z}_i^{(l)}) p_{\theta_c}(\mathbf{z}_i^{(l)}|\boldsymbol{\pi}_{ik}) d\mathbf{z}_i^{(l)} \tag{3}$$

This is a typical parametric Bayesian model due to we do not exactly the number of components. If we pre-specify K, the model capacity may be limited in real applications, since it may be difficult or even counterproductive to fix the number of parameters a prior. To automatically determine the number of components, we exploit Non-parametric Bayesian technique which provides an elegant solution on automatic adaptation of model capacity. Such adaptivity can be obtained by defining stochastic processes on rich measure spaces such as Dirichlet process (DP), Indian buffet process (IBP) and etc. More specifically, the latent variable $\mathbf{z}_i^{(l)}$ is generated via a hierarchical structure with two layers according to the continuous density function p_{θ_c} (c indicates continuous) and user-component membership $\boldsymbol{\pi}_i$. Among it, $\boldsymbol{\pi}_i$ is sampled from *GEM* distribution [24] as follows.

$$\boldsymbol{\pi}_i \sim GEM(\alpha) \text{ with } \sum_{k=1}^{\infty} \boldsymbol{\pi}_{ik} = 1 \tag{4}$$

Here *GEM* distribution is adopted because it has ability to construct a Dirichlet process. A DP is parameterized by a concentration parameter $\eta > 0$ and a base distribution G_0 over a measure space. A random variable drawn from a $DP(\eta, G0)$, is itself a distribution over the measure space. It was shown that the random distributions drawn from a DP are discrete almost surely. That is, they place the probability mass on a countably infinite collection of atoms, i.e. $G(\cdot) = \sum_{k=1}^{\infty} \boldsymbol{\pi}_{ik} \delta_{\eta_{ik}}$, where η_{ik} is the value of the k-th atom independently drawn from the base distribution G_0 and $\boldsymbol{\pi}_{ik}$ is the probability assigned to the k-th atom.

An efficient way to construct a DP is stick-breaking process. Considering a stick with unit length, we try to break it into an infinite number of segments $\boldsymbol{\pi}_{ik}$ by the following process with $v_k \sim \mathcal{B}eta(1, \alpha)$:

$$\boldsymbol{\pi}_{ik} = \begin{cases} \boldsymbol{v}_1 & \text{if } k = 1, \\ \boldsymbol{v}_k \prod_{l=1}^{k-1}(1 - \boldsymbol{v}_l) & \text{for } k > 1. \end{cases} \tag{5}$$

That is, we first choose a $\mathcal{B}eta$ variable v_1 and break v_1 of the stick. Then, for the remaining segment, we draw another $\mathcal{B}eta$ variable and break off that proportion of the remainder of the stick. Such a representation of DP provides insights for developing variational approximate inference algorithms [4]. It is easy to show that $\sum_{k=1}^{\infty} \boldsymbol{\pi}_{ik} = 1$ with probability own. Following the construction above, the stick-breaking distribution over $\boldsymbol{\pi}_i$ is written as $\boldsymbol{\pi}_i \sim GEM(\alpha)$.

Then, the generative model with local representation can be formulated as:

$$\begin{aligned} &p_G^{(l)}(\mathbf{x}_i) = \sum_{k=1}^{\infty} \boldsymbol{\pi}_{ik} \int_{\mathbf{z}_i^{(l)}} p_\theta(\mathbf{x}_i|\mathbf{z}_i^{(l)}) p_{\theta_c}(\mathbf{z}_i^{(l)}|\boldsymbol{\pi}_{ik}) d\mathbf{z}_i^{(l)} \\ &\boldsymbol{\pi}_i \sim GEM(\alpha) \text{ with } \sum_{k=1}^{\infty} \boldsymbol{\pi}_{ik} = 1 \end{aligned} \tag{6}$$

Similar to the generative process in **DGGM** model, the *Beta-Bernoulli* distribution is used to generate the feedback data. Note that the generation is built on the truncated stick-breaking process at K, which has been proved to closely approximate a true Dirichlet process as long as K is chosen to be large enough. Empirically K may be initialized to some value from tens to hundreds based on the model complexity. The useless dimensions will gradually be pruned automatically. For our case, small $\boldsymbol{\pi}_{ik}$ indicates that it is very unlike for some entries belonging to the corresponding cluster, thus, we can prune such clusters because they do not play an important role. As a consequence, the users can be clustered into K groups without a complicated model selection procedure.

By combining the local latent variable $\mathbf{z}_i^{(l)}$ and global representation $\mathbf{z}_i^{(g)}$ together, we obtain the final generative model (**DGLGM**) as follows.

$$\begin{aligned} p_G(\mathbf{x}_i) = \sum_{k=1}^{\infty} \boldsymbol{\pi}_{ik} \int_{\mathbf{z}_i^{(l)}} \int_{\mathbf{z}_i^{(g)}} p_\theta(\mathbf{x}_i | \xi_{\theta_z}(\mathbf{z}_i^{(l)}, \mathbf{z}_i^{(g)})) \\ p_{\theta_c}(\mathbf{z}_i^{(l)} | \boldsymbol{\pi}_{ik}) p(\mathbf{z}_i^{(g)}) d\mathbf{z}_i^{(l)} d\mathbf{z}_i^{(g)} \\ \boldsymbol{\pi}_i \sim GEM(\alpha) \text{ with } \sum_{k=1}^{\infty} \boldsymbol{\pi}_{ik} = 1 \end{aligned} \tag{7}$$

here the final latent representation $\mathbf{z}_i$ is obtained via local and global representations together ($\xi_{\theta_z}(\mathbf{z}_i^{(l)}, \mathbf{z}_i^{(g)})$), where $\xi_{\theta_z}(\cdot)$ is the transform function from $\mathbf{z}_i^{(g)}$ and $\mathbf{z}_i^{(l)}$ to $\mathbf{z}_i$.

For convenient computing, the latent features in $\mathbf{z}^{(g)}$ or $\mathbf{z}^{(l)}$ are assumed independent to each other. Taking the i-th user belonging to the k-th component as an example, its local latent representation $\mathbf{z}_i^{(l)}$ can be modeled with the following *Gaussian* distribution $\mathbf{z}_i^{(l)} \sim \prod_{r=1}^{d^{(k)}} \mathcal{N}(0, \boldsymbol{\lambda}_r^{(k)}) = \mathcal{N}(0, \Lambda^{(k)})$ where $\Lambda^{(k)} = diag(\boldsymbol{\lambda}^{(k)}) \in \mathbb{R}^{d^{(k)} \times d^{(k)}}$ is the covariance matrix of latent representation. To determine the optimal dimensionality $d^{(k)}$, we take advantage of the automatic relevance determination (*ARD*) technique [22, 33]. Since each latent feature (r) is assumed having zero mean, the feature with small variance ($\boldsymbol{\lambda}_r^{(k)}$) will shrink to zero. In this case, the latent features with small variance can not contribute to characterize users and items, thus, we can remove them. In other words, only the latent features with larger variance (empirically greater than 0.001) are useful to form the latent space. A good by-product is that different component can contain its own latent space size.

3.3 Inference Learning for DGLGM

One way to look at deep generative models is to postulate that they are trying to minimize certain discrepancy measures between the true data distribution and the generative distribution [10, 15]. However, recommendation data is characterized by few frequently occurring items and a long-tail of rare items, which results in those data exists in a lower dimensional manifold. Thus, we identify a problem with standard deep generative model by utilizing Wasserstein distance for sparse and high-dimensional implicit feedback data [31]. Here, Wasserstein distance is adopted rather than KL-divergence or JS-divergence, because it has the ability to measure the similarity between the distributions well and preserve the transitivity in latent space due to the much weaker topology.

Following [1, 30], the generative modeling of **DGLGM** from the optimal transport point of view can be implemented under the auto-encoder framework. The parameters in model **DGLGM** can be determined by minimizing certain discrepancy measures between the data distribution $p_{\mathbf{x}_i}$ and model distribution $p_G(\mathbf{x}_i)$ with the aid of Wasserstein distance [31]. It has been proved that minimizing the Wasserstein distance is equal to optimizing the following upper bound:

$$\begin{aligned} \mathcal{W}_G(\mathbf{x}_i, p_G(\mathbf{x}_i)) = \inf_{q_\phi(\mathbf{z}_i^{(l)}, \mathbf{z}_i^{(g)}, \boldsymbol{\pi}_i | \mathbf{x}_i) \in \mathcal{Q}_{\mathcal{Z}^{(l)} \times \mathcal{Z}^{(g)} \times \Pi}} \mathbb{E}_{p_{\mathbf{x}_i}} \mathbb{E}_{q_{\phi_c}(\mathbf{z}_i^{(l)}, \boldsymbol{\pi}_i | \mathbf{x}_i)} \\ \mathbb{E}_{q_{\phi_c}(\mathbf{z}_i^{(g)} | \mathbf{x}_i)} \mathbb{E}_{p_{\theta_g}(\mathbf{x}_i | \mathbf{z}_i^{(l)}, \mathbf{z}_i^{(g)})} [c(\mathbf{x}_i, p_G(\mathbf{x}_i))] \end{aligned}$$

where $\mathcal{Q}_{\mathcal{Z}^{(l)} \times \mathcal{Z}^{(g)} \times \Pi}$ is the set of all joint distribution over $\mathbf{z}_i^{(l)}$, $\mathbf{z}_i^{(g)}$ and $\boldsymbol{\pi}_i$, such that variational distribution $q_\phi(\mathbf{z}_i^{(l)}, \mathbf{z}_i^{(g)}, \boldsymbol{\pi}_i | \mathbf{x}_i)$ in the inference model can be factorized as $q_{\phi_c}(\mathbf{z}_i^{(l)} | \pi_i, \mathbf{x}_i)$ $q_{\phi_c}(\mathbf{z}_i^{(g)} | \mathbf{x}_i)$ $q_{\phi_d}(\boldsymbol{\pi}_i | \mathbf{x}_i)$. Any parametrization of $q_\phi(\mathbf{z}_i^{(l)}, \mathbf{z}_i^{(g)}, \boldsymbol{\pi}_i | \mathbf{x}_i)$ reduces the search space of the infimum, thus, the above objective function is in fact an upper bound of the true Wasserstein distance. $c(x, y)$ is the cost function to measure the distance between x and y, here the squared cost function $c(x, y) = ||x - y||_2^2$ is used.

To penalize the discrepancy related to $\{\mathbf{z}_i^{(l)}, \mathbf{z}_i^{(g)}, \boldsymbol{\pi}_i\}$ between prior $\sum_k \boldsymbol{\pi}_{ik} p_{\theta_c}(\mathbf{z}_i^{(g)} | \boldsymbol{\pi}_{ik}) p(\mathbf{z}_i^{(g)})$ and posterior $\mathbb{E}_{p(\mathbf{x}_i)}[q_\phi(\mathbf{z}_i^{(l)}, \mathbf{z}_i^{(g)}, \boldsymbol{\pi}_i | \mathbf{x}_i)]$, we introduce a regularizer

$$D(\mathbb{E}_{p(\mathbf{x}_i)}[q_\phi(\mathbf{z}_i^{(l)}, \mathbf{z}_i^{(g)}, \boldsymbol{\pi}_i | \mathbf{x}_i)] || \sum_k \boldsymbol{\pi}_{ik} p_{\theta_c}(\mathbf{z}_i^{(l)} | \boldsymbol{\pi}_{ik}) p(\mathbf{z}_i^{(g)}))$$

Here Maximum Mean Discrepancy (MMD), a distance on the space of densities, is adopted since it shares the properties of divergence functions and has the ability to form an unbiased U-estimator [9]. The other reason is that it is conjuncted with stochastic gradient descent (SGD) methods. For a positive-definite reproducing kernel $\mathcal{K}$, the MMD is defined by

$$\begin{aligned} \text{MMD}(q||p) = &\mathbb{E}_{z_1, z_2 \sim q}[\mathcal{K}(z_1, z_2)] + \mathbb{E}_{z_1, z_2 \sim p}[\mathcal{K}(z_1, z_2)] \\ &- 2\mathbb{E}_{z_1 \sim q, z_2 \sim p}[\mathcal{K}(z_1, z_2)] \end{aligned}$$

where the kernel $\mathcal{K}(z_1, z_2)$ is computed by the inverse multiquadratic (IMQ) kernel $\mathcal{K}(x, y) = C/(C + ||x - y||_2^2)$. The main reason is that IMQ has ability to characterize the heavier tails than classic radial basis function kernels. In this case, IMQ kernel has the ability to provide more useful information in the early training process to fasten the learning process to match the aggregated posterior with the prior.

Meanwhile, since constrained infimum is intractable, we relax it by combining with the penalization regularizer as follows.

$$\begin{aligned} &\mathcal{L}_G(\mathbf{x}_i; \theta, \phi) = \\ &\inf_{q_\phi(\mathbf{z}_i^{(l)}, \mathbf{z}_i^{(g)}, \boldsymbol{\pi}_i | \mathbf{x}_i) \in \mathcal{Q}_{\mathcal{Z}^{(l)} \times \mathcal{Z}^{(g)} \times \Pi}} \mathbb{E}_{p_{\mathbf{x}_i}} \mathbb{E}_{q_{\phi_c}(\mathbf{z}_i^{(l)}, \boldsymbol{\pi}_i | \mathbf{x}_i)} \\ &\mathbb{E}_{q_{\phi_c}(\mathbf{z}_i^{(g)} | \mathbf{x}_i)} \mathbb{E}_{p_{\theta_g}(\mathbf{x}_i | \mathbf{z}_i^{(l)}, \mathbf{z}_i^{(g)})} [c(\mathbf{x}_i, p_G(\mathbf{x}_i))] \\ &+ \lambda D(\mathbb{E}_{p(\mathbf{x}_i)}[q_\phi(\mathbf{z}_i^{(l)}, \mathbf{z}_i^{(g)}, \boldsymbol{\pi}_i | \mathbf{x}_i)] || \sum_k \boldsymbol{\pi}_{ik} p_{\theta_c}(\mathbf{z}_i^{(l)} | \boldsymbol{\pi}_{ik}) p(\mathbf{z}_i^{(g)})) \end{aligned} \tag{8}$$

Note that in feedback data generation process, *Beta* distribution is used as as prior of *Bernoulli* distribution to generate probability $\boldsymbol{\xi}_{ij}$ and stick-breaking construction weight $\boldsymbol{\pi}_{ik}$. To efficiently implement inference learning, we adopt *reparameterization trick* for sampling from *Beta* distribution. A typical way is to transfer the sampling process for *Beta* distribution via a generalized reparameterization method. Specifically we represent the *Beta* distribution

as *Kumaraswamy* distribution since *Kumaraswamy* distribution is closely related to *Beta* distribution and its cumulative distribution function (CDF) has a simple form. *Kumaraswamy* distribution is a family of continuous probability distributions defined on the interval $[0, 1]$ with positive parameters a and b, which is defined as

$$Kumaraswamy(x; a, b) = abx^{a-1}(1 - x^a)^{b-1} \tag{9}$$

when we set $a = 1$ or $b = 1$, the *Kumaraswamy* distribution and *Beta* distribution are equivalent, and for equivalent parameter settings, the *Kumaraswamy* resembles the *Beta* albeit with higher entropy. The most convenient property of the *Kumaraswamy* distribution is that its inverse CDF has a closed-form, where samples can be drawn via the inverse transform. Specifically, we sampling $u \sim Uniform(0, 1)$ and reparametrize $x = (1 - u^{1/b})^{1/a}$. The stochasticity of the sampling process is isolated, and the gradient with respect to ϕ and θ can be back-propagated through sampled *Beta* random variables.

3.4 Local Variational Optimization for DGLGM

The model parameters $\{\theta, \phi\}$ can be updated jointly by minimizing (8). Usually, the diagonal *Gaussian* distribution $\mathcal{N}(\mu_{\phi_k}(\mathbf{x}_i), diag(\sigma^2_{\phi_k}(\mathbf{x}_i)))$ is employed as posterior $q_\phi(\mathbf{z}_i^{(l)}, \boldsymbol{\pi}_{ik}|\mathbf{x}_i)$ and $q_\phi(\mathbf{z}_i^{(g)}|\mathbf{x}_i)$, which can be parametrized by a neural network (e.g., we sampling $\epsilon \sim \mathcal{N}(0, \mathbf{I}_k)$ and reparameterize $\mathbf{z}_i^{(l)} = \mu_\phi(\mathbf{x}_i) + \epsilon \odot \sigma_\phi(\mathbf{x}_i)$). Thus, the problem becomes to find the good variational parameters $\psi(\mathbf{x}_i) = \{\mu_\phi(\mathbf{x}_i), \sigma^2_\phi(\mathbf{x}_i)\}$ for a given $\mathbf{x}_i$, where $\psi(\mathbf{x}_i)$ is the output of neural network with parameters ϕ that are trained to maximize $\mathcal{L}_G(\mathbf{x}_i; \theta, \phi, \psi(\mathbf{x}_i))$.

However, standard learning algorithm directly uses the random $\psi(\mathbf{x}_i)$ to estimate (8), which can not guarantee obtaining the optimal variational parameters. Moreover, random $\psi(\mathbf{x}_i)$ may yield a much looser upper bound of the true Wasserstein distance if the inference network is either not sufficiently powerful or its parameters ϕ are not well tuned. Motivated by the stochastic variational inference [17], we try to automatically determine the optimal variational parameters ψ^* and guarantee the tightest upper bound of (3.3) for a data point $\mathbf{x}_i$

$$\mathcal{L}_G(\mathbf{x}_i; \theta, \phi) \geq \mathcal{L}_G(\mathbf{x}_i; \theta, \phi, \psi^*(\mathbf{x}_i)) \geq \mathcal{W}_G(\mathbf{x}_i, p_G(\mathbf{x}_i)) \tag{10}$$

Algorithm 1 Learning with local variational optimization for **DGLGM**

Input: Implicit preference matrix $\mathbf{X} = [\mathbf{x}_1, \cdots \mathbf{x}_n]$ with n users. Randomly initialize θ, ϕ.

1: **for** $t = 1$ to T **do**
2: Sample a user $\mathbf{x}_i$ and set $\psi_0 = \psi(\mathbf{x}_i)$
3: Approximate $\psi^N \approx \psi^*(\mathbf{x}_i) = \arg\min_\psi \mathcal{L}_G(\mathbf{x}_i; \theta^t, \phi^t, \psi(\mathbf{x}_i))$
4: **for** $\tau = 0, \cdots, N-1$ **do**
5: $\psi^{\tau+1} = \psi^\tau + \eta_\psi \nabla_{\psi_\tau} \mathcal{L}_G(\mathbf{x}_i; \theta^t, \phi^t, \psi^\tau)$
6: **end for**
7: Update θ: $\theta^{t+1} \leftarrow \theta^t + \eta_\theta \nabla_{\theta_t} \mathcal{L}_G(\mathbf{x}_i; \theta^t, \phi^t, \psi^N)$
8: Update ϕ: $\phi^{t+1} \leftarrow \phi^t + \eta_\phi \nabla_{\phi_t} \mathcal{L}_G(\mathbf{x}_i; \theta^{t+1}, \phi^t, \psi(\mathbf{x}_i))$
9: Reduce certain dimension of $\mathbf{z}_i$ if corresponding *ARD* variance is less than the predefined threshold ϵ_λ
10: **end for**

To obtain the optimal ψ^* for further updating model parameters $\{\theta, \phi\}$, we propose a local variational optimization (LVO) method to estimate parameters $\psi = \psi(\mathbf{x}_i)$ with the aid of the inference network. Its main idea is to optimize ψ via gradient descent, where one initializes with ψ^0 and takes successive steps of $\psi^{\tau+1} = \psi^\tau + \eta_\psi \nabla_{\psi_\tau} \mathcal{L}_G(\mathbf{x}_i; \theta, \phi, \psi^\tau)$, where the gradient with respect to ψ can be approximated via Monte Carlo sampling. As shown in line 4-6 of Algorithm 1, the resulting ψ^N can approach the optimal variational parameters. Then, it can be used to calculate the gradients of $\mathcal{L}_G(\mathbf{x}_i; \theta, \phi, \psi^N)$ on θ and ϕ respectively. Finally, these two parameters will be iteratively updated via stochastic back-propogation, as shown in line 7 and line 8. Note that N in inner optimization is a predefined value, usually few steps are enough to find the optimal ψ. Model parameters ψ, θ and ϕ are updated using stochastic gradient descent with learning rates $\eta_\psi, \eta_\theta, \eta_\phi$ obtained via ADAM [14].

3.5 Error Bound Analysis

In this subsection, we will theoretically analyze the generalization error bounds of the proposed **DGLGM** model. Motivated by [2], we can define the generalization of feedback data generation process by measuring the difference between generative data distribution $\mathcal{X}_G$ and real data distribution $\mathcal{X}_{real}$, so that the population distance between the true and generative distributions ($\mathcal{X}_{real}$ and $\mathcal{X}_G$) is close to the empirical distance between the empirical distributions ($\hat{\mathcal{X}}_{real}$ and $\hat{\mathcal{X}}_G$).

Definition 3.1. For the empirical version of the true distribution ($\hat{\mathcal{X}}_{real}$) with n training examples, a generated distribution ($\hat{\mathcal{X}}_G$) generalizes under the distance $d(\cdot, \cdot)$ between distributions with generalization error $\delta_1 > 0$ if the following holds with high probability,

$$|E(\mathbf{X}) - E(\hat{\mathbf{X}})| \leq \delta_1$$

where

$$E(\mathbf{X}) = \mathbb{E}_{\mathbf{x}^{(real)} \sim \mathcal{X}_{real}, \mathbf{x}^{(G)} \sim \mathcal{X}_G} d(\mathbf{x}^{(real)}, \mathbf{x}^{(G)}),$$

$$E(\hat{\mathbf{X}}) = \frac{1}{mn} \sum_{i=1}^{n} \sum_{j=1}^{m} d(\hat{\mathbf{x}}_{ij}^{(real)}, \hat{\mathbf{x}}_{ij}^{(G)}) | \hat{\mathbf{x}}_{ij}^{(real)} \sim \hat{\mathcal{X}}_{real}, \hat{\mathbf{x}}_{ij}^{(G)} \sim \hat{\mathcal{X}}_G$$

$E(\mathbf{X})$ indicates the population distance between the true and generated distributions ($\mathcal{X}_{real}$ and $\mathcal{X}_G$). $E(\hat{\mathbf{X}})$ is the empirical distance between the empirical distributions $\hat{\mathcal{X}}_{real}$ and $\hat{\mathcal{X}}_G$.

Since the proposed **DGGM** can be seem as a generalized matrix completion model, the *Frobenius norm* between the ground truth feedback data ($\mathbf{X}$) and the generative feedback data ($\hat{\mathbf{X}}$) with n users and m items, is used as the metric to establish the error bound of the proposed method, i.e. $d(\mathbf{x}_{ij}^{(real)}, \hat{\mathbf{x}}_{ij}^{(G)}) = ||\mathbf{x}_{ij}^{(real)} - \hat{\mathbf{x}}_{ij}^{(G)}||_F$. We will use $||\mathbf{x}_{ij} - \hat{\mathbf{x}}_{ij}||_F$ as shorthand for $||\mathbf{x}_{ij}^{(real)} - \hat{\mathbf{x}}_{ij}^{(G)}||_F$.

According to the generative process of **DGLGM**, it can be seen that the whole feedback matrix is generated by $K + 1$ **DGGM**s with *ARD* prior (K local matrices with local properties and one global matrix with global property). Here we define $\mathbf{X}^{(k)}$ as k-th submatrix, $n^{(k)}$ is the number of users in submatrix $\mathbf{X}^{(k)}$. Firstly, we establish the error bound of the proposed **DGLGM**, i.e., the generative error of the deep global and local generative recommendation model is bounded, so that we can still find optimal generative model for recommendation by optimizing the proposed optimization problem.

Theorem 3.2. *For any* $\mathbf{X} \in \mathbb{N}^{n \times m}$ $(m, n > 2)$,

$$E(\mathbf{X}) \geq E(\hat{\mathbf{X}}) + \sqrt{\frac{\log \delta_1}{-2mn} (\max_{(i,j)} d_{ij})^2}$$

holds with probability at least $1-\delta_1$ *over uniformly choosing an empirical version of* $\mathbf{X}$. *Here,* $d_{ij} = d(\hat{\mathbf{x}}_{ij}^{(real)}, \hat{\mathbf{x}}_{ij}^{(G)})$.

Proof. Since the entries in $\mathbf{X}$ are chosen independently and uniformly, it is reasonable to assume each $d_{ij} = d(\hat{\mathbf{x}}_{ij}^{(real)}, \hat{\mathbf{x}}_{ij}^{(G)})$ is a random variable and satisfies $p(\zeta \geq d_{ij} \geq 0) = 1$, where $\zeta = \max_{ij} d_{ij}$. Hence, based on the Hoeffding Inequality, we have $p(|E(\mathbf{X}) - E(\hat{\mathbf{X}})| \geq \epsilon) \leq \exp(\frac{-2mn\epsilon^2}{\zeta^2})$. By setting $\epsilon = \sqrt{\frac{\log \delta_1}{-2n}\zeta^2}$, we have $p\Big(|E(\mathbf{X}) - E(\hat{\mathbf{X}})| \leq \sqrt{\frac{\log \delta_1}{-2mn}\zeta^2}\Big) \geq 1-\delta_1$, i.e.,

$$p\Big(E(\mathbf{X}) \leq E(\hat{\mathbf{X}})| + \sqrt{\frac{\log \delta_1}{-2mn}(\max_{(ij)} d_{ij})^2}\Big) \geq 1-\delta_1$$

Therefore, the errors of the deep global generative recommendation model(**DGGM**) are bounded. □

Then, we theoretically prove the generalization error bounds of a single **DGGM** related to a single mixture component. To make the theorems more convincing, we make several standard assumptions: (1) each submatrix $\mathbf{X}^{(k)}$ related to k-th mixture component is incoherent[6, 29], (2) $\mathbf{X}^{(k)}$ is well-conditioned[6], and (3) the number of users is larger than items in each submatrix ($n^{(k)} \geq m^{(k)} = m$). As shown in Theorem 7 in [5], a theoretical bound to generalized matrix completion model (**DGGM**) is existing as follows.

Theorem 3.3. *If* $\mathbf{X}^{(k)}$ *is well-conditioned and incoherent such that* $|\Omega| \geq C\mu^2 m d^{(k)} log^6 m$, *then with high probability* $1-m^{-3}$, $\mathbf{X}^{(k)}$ *satisfies*

$$||\mathbf{X}^{(k)} - \hat{\mathbf{X}}^{(k)}||_F \leq 4\varepsilon\sqrt{\frac{(2+\rho)m}{\rho}} + 2\varepsilon = 4\sqrt{\frac{(2+\rho)m}{\rho}} + 2$$

where $\rho = |\mathbf{X}^{(k)}|/(n^{(k)}m)$, $\varepsilon = \max(\hat{\mathbf{X}}) - \min(\hat{\mathbf{X}}) = 1$ *and* $d^{(k)}$ *is the optimal dimensionality of latent representation for* k*-th submatrix,* Ω *is the set of observed feedback.*

This theorem indicates that the generative error of each mixture component $\hat{\mathbf{X}}^{(k)}$ is bounded. Then, based on Theorem 3.3, we can analyze the generalization error bound of the proposed **DGLGM** method on whole rating matrix via the following theorem.

Theorem 3.4. *If each submatrix related to a specific mixture component satisfied Theorem 3.3, then with high probability* $1-\delta$, $\hat{\mathbf{X}}$ *is divided into* K *submatrices satisfies*

$$P\Big(||\mathbf{X} - \hat{\mathbf{X}}||_F \leq \frac{\omega}{\sqrt{mn}}(4\sqrt{\frac{(2+\rho)nK}{\rho}} + 2K)\Big) \geq 1-\delta$$

where $\delta = (2Km)^{-3}$ *and* ω *indicates the average mixture weight.*

Proof. For every user-item pair (i, j), an observed implicit feedback $\mathbf{x}_{ij}$ is equal to $\hat{\mathbf{x}}_{ij} + z$ where z is a random variable whose absolute error is bounded by

$$||W \otimes (\mathbf{X} - \hat{\mathbf{X}})||_\infty \leq \omega(\max(\mathbf{X}) - \min(\mathbf{X})) = \omega\varepsilon = \omega \tag{11}$$

where W indicates the mixture weights and. $\varepsilon = \max(\mathbf{X}) - \min(\mathbf{X}) = 1$ since we focus on implicit feedback data. By applying Theorem 3.3 to implicit preference reconstruction problem with bounded noise, we get with probability greater than $1-v^{-3}$ that every mixture model to approximate $\mathbf{X}$ will satisfy

$$||W \otimes (\mathbf{X} - \hat{\mathbf{X}})||_F \leq \frac{\omega}{\sqrt{mn}}(4\sqrt{\frac{\gamma(2+\rho)}{\rho}} + 2) \tag{12}$$

where $v = \max(m, n)$, $\gamma = \min(m, n)$. For a mixture model, there are K different mixture components $\mathbf{X}^{(k)}$ and obviously $\sum_k \gamma^{(k)} \leq n$. Using Cauchy-Schwarz inequality, we get $\sum_k \sqrt{\gamma^{(k)}} \leq \sqrt{K\sum_k \gamma^{(k)}} \leq \sqrt{Kn}$. Therefore, we can bound the reconstruction error as follow:

$$\begin{aligned} ||W \otimes (\mathbf{X} - \hat{\mathbf{X}})||_F &\overset{(a)}{\leq} \sum_k ||W^{(k)} \otimes (\mathbf{X}^{(k)} - \hat{\mathbf{X}}^{(k)})||_F \\ &\overset{(b)}{\leq} \sum_k \frac{\omega}{\sqrt{mn}}(4\sqrt{\frac{\gamma^{(k)}(2+\rho)}{\rho}} + 2) \\ &\overset{(c)}{\leq} \frac{\omega}{\sqrt{mn}}(4\sqrt{\frac{2Kn(2+\rho)}{\rho}} + 2K) \end{aligned} \tag{13}$$

in which (a) holds due to the triangle inequality of Frobenius norm; and (b) holds due to (12). Since for all (i, j) pairs, we have

$$E(\hat{\mathbf{X}}) = \frac{||\mathbf{X} - \hat{\mathbf{X}}||_F}{\sqrt{mn}} \leq \frac{||W \otimes (\mathbf{X} - \hat{\mathbf{X}})||_F}{\sqrt{mn}} \tag{14}$$

Combining (13) and (14), we established the error bound of $\hat{\mathbf{X}}$ as stated above. In order to adjust the confidence level, we take a union bound of events $||W \otimes (\mathbf{X} - \hat{\mathbf{X}})||_F \geq \frac{\omega}{\sqrt{mn}}(4\sqrt{\frac{\gamma(2+\rho)}{\rho}} + 2)$ for each mixture component, then we have $\sum_{(k)}(v^{(k)})^{-3} \geq (K\sum_{(k)} v^{(k)})^{-3} \geq (2Km)^{-3}$. Thus, the error bound holds with probabilities at least $1-(2Km)^{-3}$ □

Theorem 3.5. *If Theorem 3.4 holds, then with probability of at least* $1-\delta$, *the reconstructed feedback matrix* $\hat{\mathbf{X}}$ *based on local and global representations setting satisfies:*

$$P\Big(||\mathbf{X} - \hat{\mathbf{X}}||_F \leq \frac{\omega}{\sqrt{mn}}(4\sqrt{\frac{2(K+1)n(2+\rho)}{\rho}} + 2(K+1))\Big) \geq 1-\delta$$

where $\delta = (2(K+1)m)^{-3}$

Proof. By Theorem 3.4, we bound the error of $\hat{\mathbf{X}}$ related to both local and global representations as follows:

$$\begin{aligned} ||\mathbf{X} - \hat{\mathbf{X}}||_F &\overset{(a)}{\leq} ||Q^{(l)} \otimes (\mathbf{X} - \hat{\mathbf{X}})||_F + ||Q^{(g)} \otimes (\mathbf{X} - \hat{\mathbf{X}})||_F \\ &\overset{(b)}{\leq} \frac{1}{K+1}\sum_{k=1}^{K+1} ||Q^{(k)} \otimes (\mathbf{X}^{(k)} - \hat{\mathbf{X}}^{(k)})||_F \\ &\overset{(c)}{\leq} \frac{1}{K+1}\sum_{k=1}^{K+1} \frac{\omega}{\sqrt{mn}}(4\sqrt{\frac{\gamma^{(k)}(2+\rho)}{\rho}} + 2) \\ &\overset{(d)}{\leq} \frac{\omega}{\sqrt{mn}}(4\sqrt{\frac{2(K+1)n(2+\rho)}{\rho}} + 2(K+1)) \end{aligned} \tag{15}$$

where $Q^{(l)}$ and $Q^{(g)}$ indicate the weights of local and global representations modeling via neural networks. $Q^{(k)}$ is the weight for k-th local matrix and $Q^{(K+1)}$ is equals to $Q^{(g)}$. Inequality (b) holds due to the triangle inequality of Frobenius norm. (c) holds due to inequation (12). Finally, applying the same strategy in Therorem 3.4, we conclude the proof. The confidence level is adjusted to $(2(K+1)m)^{-3}$ using the union bound property as in Theorem 3.5. □

4 RELATED WORK

The existing research on VAE based deep generative model mainly focuses on the following three points. The first one is proposed by introducing different likelihood for specific task, e.g., by replacing original *Gaussian* likelihood in VAE with *Multinomial* likelihood, Mult-VAE is proposed to model the generative process of implicit feedback data for recommendation [19]. The second one focus on

applying different distance to measure the difference between the generative distribution and the true data distribution, Wasserstein distance is utilized in Wasserstein auto-encoder [30] from the optimal transport point of view, and aWAE [37] extends Wasserstein auto-encoder (WAE) with the aid of mutual information and sparse regularization for collaborative filtering. The third one is to introduce proper prior for modeling latent variables, e.g., SVAE [12] introduces *Gaussian* mixture model to fit data with different clusters under the VAE architecture. FactorVAE [13] introduces disentangled *Gaussian* prior to encourage the distribution of representations to be factorial end independent across the dimensions. Although above methods are proposed, they still suffer from several challenging problems, such as sparsity of data, uncertainty of optimization process, and improper prior assumption.

Our approach focuses on modeling the generative process of implicit feedback to solve the challenges mentioned above by fully considering the above three potential points. Considering the property of implicit feedback, we introduce a *Beta-Bernoulli* likelihood for modeling sparse/dense implicit feedback. Faced with the users' diverse preference, a nonparametric *Dirichlet* process based mixture prior with ARD technique is introduced to model the prior of latent representation, which has the ability to capture the users' interests adaptively and automatically. From inference learning perspective, the VAE reconstruction error is based on Wasserstein distance instead of traditional KL divergence, which has the ability to modeling high-dimensional recommendation data.

5 EXPERIMENTS

In this section, we evaluate the proposed deep generative model on four datasets by comparing with the state-of-the-art recommendation methods.

5.1 Experimental Setting

Table 2: Summary of experimental datasets

	ML 20M	*Netflix*	*Epinions*	*Yelp*
♯ users (n)	138,493	480,189	49,290	1,182,626
♯ items (m)	26,744	17,770	139,738	156,638
♯ ratings	20,000,263	100,000,000	664,824	4,731,265
RDensity	0.54%	1.17%	0.010%	0.0026%
$\overline{m}_i$	144	208	14	4
$\overline{n}_j$	748	5,627	5	30

$\overline{m}_i$: the average number of items rated by each user
$\overline{n}_j$: the average number of users interested in each item

Datasets: In experiments, four widely used recommendation datasets, *MovieLens 20M (ML 20M)* [2], *Netflix* [3], *Epinions* [4] and *Yelp* [5], are used to validate the recommendation performance. Among them, *ML 20M* and *Netflix* come from movie domain, *Epinions* belongs to online product domain, and *Yelp* is related to local business domain. The preference scores in *ML 20M* are 10 discrete numerical values in the range of [0.5,5] with step 0.5, while the ratings in other datasets are ordinal values on the scale 1 to 5. Following [19], we binarize explicit data by keeping ratings of four or higher. More detailed information is summarized in Table 2.

[2]https://grouplens.org/datasets/movielens/
[3]https://www.netflixprize.com
[4]http://www.trustlet.org/downloaded_epinions.html
[5]https://www.yelp.com/dataset/challenge

Metrics for ranking estimation: Recall ($Recall@K(i) = |Re(i) \cap T(i)|/|T(i)|$) is used as ranking estimation, where $Re(i)$ denotes the set of recommended items to user i and $T(i)$ denotes the set of favorite items of user i. Meanwhile, the normalized discounted cumulative gain (NDCG) is adopted to measure the item ranking accuracy, which can be computed by: $NDCG@K(i) = \frac{DCG@K(i)}{IDCG@K(i)}$ (where $DCG@K(i) = \sum_{r=1}^{K}(2^{\mathbb{I}(w(r)\in\mathcal{I}_i)}-1)/\log_2(r+1)$, and $IDCG$ is the DCG value with perfect ranking). $\mathbb{I}(\cdot)$ is the indicator function and $w(r)$ is the item at rank j. $\mathcal{I}_i$ is the set of items that user i clicked on. Five-fold cross-validation technique is used and their averaged results are reported.

Baselines: We choose three kinds of recommendation methods as baselines. The details are summarized as follows: Traditional recommendation methods: BPR [25], SLIM [23] and GLOMA [8]. Deep neural network recommendation models: NCF [11], CDAE [34] and NCR [27]. Deep generative recommendation models: Mult-VAE [19], aWAE [37].

Parameter setting: The parameters of all algorithms we compared with are either adopted from their original papers or determined by experiments. For **DGLGM**, the architecture for the neural networks of encoder and decoder are symmetrical, and we adopt multilayer perceptron with three layers, and the best overall architecture is $[m \rightarrow \{600\}_{k=1}^{K} \rightarrow d \rightarrow 600 \rightarrow m]$ where m is the number of items. Moreover, we find that going deeper does not improve recommendation performance. We adopt *reparameterization trick* introduced in section 3.3 for sampling from *Beta* distribution. The dropout technique [28] is adopted at the input layer with probability 0.5 and we do not apply the weight decay for any parts. In SLIM, the regularization parameters are tuned in 0.1, 0.5, 1, 5. For GLOMA, π_1 and π_2 are 0.3 and 0.5. For AE-based deep methods (CDAE, Mult-VAE and our method), the hyper-parameters are automatically tuned according to the TPE method [3].

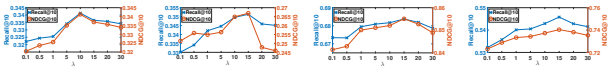
The held-out users strategy and five-fold cross validation are used to evaluate the recommendation performance. The 20% users are taken as held-out users and we use the same number of users for validation and test. For each held-out user, his/her feedback data is randomly split into five equal sized subsets. Among them, four subsets are used to obtain the latent representation, and the rest subset is for evaluation in each round. Finally, the averaged results of five rounds are reported.

5.2 Results and Discussion

In this section, we investigate **DGLGM** from six views. Firstly, a series of experiments are conducted to test the effect of model parameters. Secondly, the effect of optimal dimensionality determination (*ARD* prior) is analyzed. Thirdly, we analyze the effect of global representations an local representations on **DGLGM**. Next, **DGLGM** is compared with baselines from *All Users* view and *Near-cold-start Users* view. Finally, we analyze the generalization ability and convergence of the proposed model.

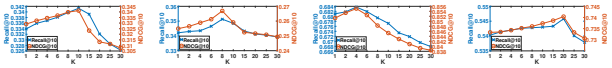
5.2.1 Effect of parameters. The first experiment is designed to investigate the effect of regularizer coefficient λ for recommendation performance. Figure 3 shows the effect of regularizer parameter λ on four datasets in terms of Recall@10 and NDCG@10. The results demonstrate that **DGLGM** performs better as λ increases, reaches the best value at around $\lambda = 10$ for *ML 20M* and $\lambda = 15$ for *Netflix*, *Epinions* and *Yelp*, and then decreases in performance as λ grows

larger. Sine MMD regularization with IMQ kernel can provide more useful information from prior, larger λ may put too much attention on capacity limitation, thus decrease the efficacy of the recommendation, while a smaller λ may lower the prior dependency and reduce the effect of prior information.



(a) *ML 20M* (b) *Netflix* (c) *Epinions* (d) *Yelp*

Figure 3: Effect of λ (regularizer coefficient).

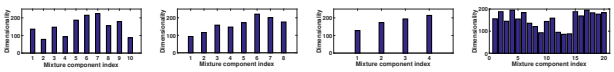


(a) *ML 20M* (b) *Netflix* (c) *Epinions* (d) *Yelp*

Figure 4: Effect of K (the number of mixture components).

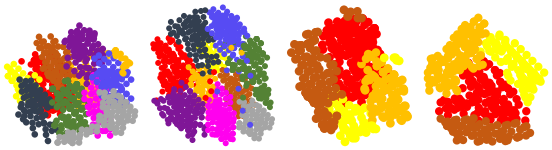
Additionally, the number of mixture components K is the key parameter for the proposed **DGLGM**, which controls the diversity of user preference. The second experiment is conducted to evaluate the effect of the number of mixture components for recommendation performance. The value of K is tuned in {1, 2, 4, 6, 8, 10, 15, 20, 25, 30}. Figure 4 shows the effect of K in terms of Recall@10 and NDCG@10. It can be seen that the performance is improved when K increase. This is due to the fact that different mixture component may capture different user preference and further improve recommendation performance. After reaching the optimal value K (The optimal values are 10, 8, 4 and 20 on *ML 20M*, *Netflix*, *Epinions* and *Yelp*, respectively), the performance decrease with the increase of K. This implies that large K may overfit the training data. Furthermore, it is interesting to observe that the optimal K is large on the larger datasets since the number of users and items are large, which contains more diversities.

5.2.2 Effect of optimal dimensionality determination. Inspired by *ARD* technique, the proposed **DGLGM** model can adaptively determine the optimal dimensionality of local latent representation for each mixture component, which can be modeled in an optimal latent space. Note that we set the *ARD* threshold ϵ_λ to 0.001. Figure 5 demonstrates the optimal latent dimensionality for $\mathbf{z}_i^{(l)}$ in each mixture component determined by **DGLGM** on different datasets. It can be seen that the dimensionalities have different variance, and the minimal and maximal dimensionalities are {78, 224}, {93, 221}, {128, 214} and {86, 195} on *ML 20M*, *Netflix*, *Epinions* and *Yelp*, respectively. Furthermore, the optimal dimensionalities for global representation are 251, 234, 265 and 259 on *ML 20M*, *Netflix*, *Epinions* and *Yelp*, respectively. Obviously, the dimensionalities of global representation is greater than local representation, the reason is that global structure among users contains more preference information and it is better to represent each user in a higher dimensional latent space.



(a) *ML 20M* (b) *Netflix* (c) *Epinions* (d) *Yelp*

Figure 5: Demonstrating the optimal latent dimensionality d for $\mathbf{z}_i^{(l)}$ in each mixture component determined by DGLGM.



(a) *ML 20M*-DLGM (b) *ML 20M*-DGLGM (c) *Epinions*-DLGM (d) *Epinions*-DGLGM

Figure 6: Demonstrating the t-SNE projections of the local latent representations $\mathbf{z}_i^{(l)}$ learned by DLGM and DGLGM on dense dataset *ML 20M* and sparse dataset *Epinions*.

Additionally, we compare the recommendation performance of **DGLGM** against **DGGM** with different fixed latent dimensionality {50, 100, 150, 200}. Table 3 lists the optimal latent dimensionality in each mixture component obtained by **DGLGM** for different datasets. The best and second results are marked in bold and underline. As we can see, the performance of **DGGM** is unstable when the dimensionality increase from 50 to 200. Taking *Yelp* dataset as an example, **DGGM** with dimensionality 150 achieves better performance than dimensionality 200 but worse than dimensionality 50. The reason is that mapping users with preference diversity into a fixed latent space size cannot model all users well, so that some users are either under-fitted or over-fitted. Nevertheless, **DGGM** with all latent dimensionalities are inferior to **DGLGM**, because each user can be modeled in an optimal latent subspace, which can alleviate overfitting or underfitting problem to some extent.

5.2.3 Effect of global representation. The proposed **DGLGM** not only consider local structure among users but also capture the global structure in the whole preference space. In order to investigate the effect of global representation learned from the whole preference space, we remove the global model from the **DGLGM** and it degenerates into **DLGM**, i.e., deep generative model only considers mixture property. Table 4 lists the recommendation performance obtained via **DLGM** and **DGLGM** on four datasets in terms of Recall@10 and NDCG@10. Obviously, **DGLGM** performs better than **DLGM**, which indicates considering global structure among all users is useful for recommendation.

In order to further investigate the effect of global representation and demonstrate the properties of the learned latent representations, t-SNE was used to embed the local latent representations learned via **DGLGM** and **DLGM** into two dimensional space. Figure 6 demonstrates the t-SNE projections of the local latent representations $\hat{\mathbf{z}}_i$ learned by **DLGM** and **DGLGM** on dense dataset *ML 20M* and sparse dataset *Epinions*. Instance from different mixture components (user diversities) are represented by different colors. As shown in Figure 6, user diversities learned in the **DGLGM** are clustered with noticeably more cohesion and separation in both datasets, which indicates that the global representation is beneficial to the learning process of local latent representations and has contributed to capture more diverse user groups.

5.2.4 Diversity evaluation. One of the main contributions of **DGLGM** is exploit local mixture model to capture diverse user interests, and users assigned to the same mixture component have the same preference. To confirm this, taking *ML 20M* dataset as example, we investigate the corresponding semantical information of mixture component. In *ML 20M)*, each movie is marked by one or more genres labels, and all items belong to 18 genres, including

Table 3: Comparasion of DGLGM and DGGM with different latent dimensionality for z_i in terms of Recall@10 and NDCG@10.

Datasets	Metrics	DGGM-50	DGGM-100	DGGM-150	DGGM-200	DGLGM
ML 20M	Recall@10	0.3305	0.3318	0.3334	0.3325	**0.3413**
	NDCG@10	0.3259	0.3271	0.3289	0.3277	**0.3412**
Netflix	Recall@10	0.3411	0.3401	0.3421	0.3416	**0.3513**
	NDCG@10	0.2531	0.2515	0.2549	0.2537	**0.2669**
Epinions	Recall@10	0.6789	0.6801	0.6783	0.6779	**0.6937**
	NDCG@10	0.8492	0.8503	0.8484	0.8480	**0.8551**
Yelp	Recall@10	0.5388	0.5403	0.5379	0.5367	**0.5458**
	NDCG@10	0.7316	0.7331	0.7305	0.7301	**0.7404**

Table 4: Comparisons of DGLGM and DLGM.

Metrics	Methods	*ML 20M*	*Netflix*	*Epinions*	*Yelp*
Recall@10	**DLGM**	0.3397	0.3496	0.6824	0.5443
	DGLGM	**0.3413**	**0.3513**	**0.6837**	**0.5458**
NDCG@10	**DLGM**	0.3396	0.2649	0.8529	0.7390
	DGLGM	**0.3412**	**0.2669**	**0.8551**	**0.7404**

Action, *Adventure*, *Animation*, *Children*, *Comedy*, *Crime*, *Documentary*, *Drama*, *Fantasy*, *Film-Noir*, *Horror*, *Musical*, *Mystery*, *Romance*, *Sci-Fi*, *Thriller*, *War* and *Western*.

For the k-th mixture component obtained by **DGLGM**, let $m^{(k)}$ be the number of items that users related to the k-th mixture component rated, where the number of items belonging to the q-th genere is denoted as $m_q^{(k)}$ and $\sum_{q=1}^{n_I} m_q^{(k)} = m^{(k)}$. In this case, $\{m_q^{(k)}/m^{(k)}\}_{q=1}^{n_I}$ indicates the genres distribution in the k-th mixture component, where n_I is the number of genres. The proportion of item genres in each mixture component on *ML 20M* are shown in Figure. 7. Obviously, each mixture component has its most related genre, such as *Drama*, *Action*, *Comedy*, *Thriller*, *War*, *Children's*, *Romance*, *Adventure*, *Crime* and *Scf-Fi* for mixture component 1 to 10, which indicates that **DGLGM** has the ability capture diversity of user preference. Furthermore, we demonstrate the item proportion with specific genre for each mixture component. Due to the page limitation, we only show the results related to 6-th and 7-th mixture component, as shown in Figure 8. Most items ($> 70\%$) in one component are from the same genre (this property also exists in the other mixture components), which indicates each mixture component has its own focus and **DGLGM** has the ability to capture the diversity of user preference. This result can be further used to interpret the recommendation results.

5.2.5 Performance comparison. In order to deeply investigate the recommendation performance, the second experiment is conducted to compare the proposed **DGLGM** with several baselines from two views (*All Users* and *Near-cold-start Users*). *All Users* indicates that all users are used as the testing set. *Near-cold-start Users* view means that the users with less than five interacted items are involved in the testing set.

Table 5 shows the recommendation performance for *All Users* on four datasets. The best and second results are marked in bold and underline. Obviously, most of the deep methods significantly outperform traditional ranking approaches (BPR, SLIM, GLOMA), which indicates non-linear features are beneficial for improving recommendation quality. As we can see, in most cases, deep generative models (**DGLGM**, aWAE, Mult-VAE) are superior to other

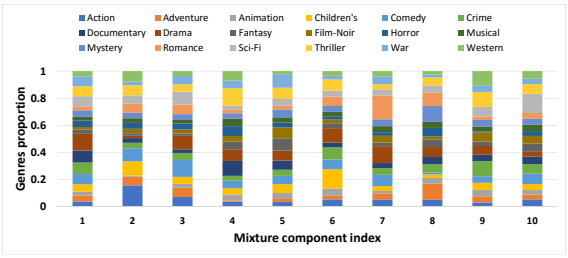


Figure 7: The distribution of item genres related to each mixture component for *ML 20M* dataset.

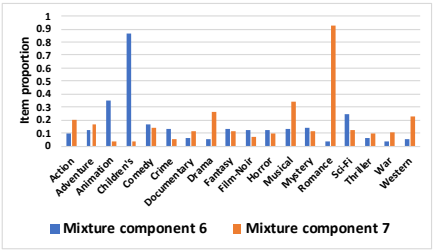


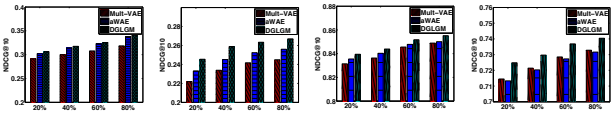
Figure 8: The item proportion with genre in the 6-th and 7-th mixture components for *ML 20M* dataset.

deep methods (CDAE, NCF, NCR), which indicates that proper generative process for implicit feedback data matching is helpful to learn more useful latent representation. **DGLGM** performs better than the state-of-the-art deep methods, which confirms that considering user diversity via mixture deep generative model is helpful to predict more precise recommended items.

As we known, sparser feedback data, more challenging the personalized recommendation task. In these four datasets, the average number of *Near-cold-start Users* are 1543, 2201, 34310 and 7439651 in *ML 20M*, *Netflix*, *Epinions* and *Yelp* respectively, and they are about 1.85%, 0.46%, 69.6%, 78.4% of all users in the corresponding datasets. It can be seen that *Epinions* and *Yelp* are extremely sparse, thus we further evaluate the recommendation performance on *Near-cold-start Users* for these two datasets. The results obtained by the proposed model and all baselines on *Epinions* and *Yelp* are listed in Table 5. As expected, **DGLGM** has the ability to handle sparse problem and is superior to all baselines. The reason is that *beta-bernoulli* distribution is suit to model sparse implicit feedback since it has the ability to consider both observed and non-observed feedback. Meanwhile MMD regularizer with IMQ kernel is characteristic and has much heavier tails, which is more suit for sparse data. Furthermore, the proposed iterative optimization method with updating local variational parameter provides a tighter upper bound for original Wasserstein auto-encoder objective and lower the final reconstruction error, which is beneficial to recommendation.

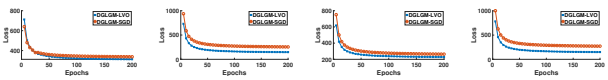
Table 5: Comparisons of different recommendation methods from *All Users* view and *Near-cold-start Users* view.

All Users										
Datasets	Metrics	BPR	SLIM	GLOMA	CDAE	NCF	NCR	Mult-VAE	aWAE	**DGLGM**
ML 20M	Recall@10	0.3045	0.3122	0.3276	0.3221	0.3332	0.3338	0.3320	0.3330	**0.3413**
	NDCG@10	0.3011	0.3033	0.3241	0.3224	0.3258	0.3279	0.3185	0.3380	**0.3412**
Netflix	Recall@10	0.2985	0.3003	0.3208	0.3177	0.3223	0.3261	0.3208	0.3410	**0.3513**
	NDCG@10	0.2044	0.2126	0.2385	0.2335	0.2456	0.2485	0.2449	0.2560	**0.2669**
Epinions	Recall@10	0.6321	0.6413	0.6659	0.6451	0.6785	0.6791	0.6812	0.6801	**0.6837**
	NDCG@10	0.8122	0.8257	0.8403	0.8322	0.8465	0.8451	0.8489	0.8501	**0.8551**
Yelp	Recall@10	0.4910	0.5027	0.5217	0.5133	0.5278	0.5391	0.5322	0.5353	**0.5458**
	NDCG@10	0.7082	0.7144	0.7298	0.7246	0.7314	0.7318	0.7328	0.7316	**0.7404**
Near-cold-start Users										
Datasets	Metrics	BPR	SLIM	GLOMA	CDAE	NCF	NCR	Mult-VAE	aWAE	**DGLGM**
Epinions	Recall@10	0.4136	0.4233	0.4344	0.4375	0.4544	0.4564	0.4587	0.4597	**0.4631**
	NDCG@10	0.7233	0.7285	0.7312	0.7344	0.7435	0.7441	0.7461	0.7477	**0.7499**
Yelp	Recall@10	0.2711	0.2833	0.2913	0.3012	0.3185	0.3105	0.3244	0.3252	**0.3293**
	NDCG@10	0.6812	0.6887	0.7041	0.6922	0.7033	0.7056	0.7123	0.7161	**0.7194**



(a) *ML 20M* (b) *Netflix* (c) *Epinions* (d) *Yelp*

Figure 9: NDCG@10 comparison of Mult-VAE, aWAE and DGLGM in terms of different training sets.



(a) *ML 20M* (b) *Netflix* (c) *Epinions* (d) *Yelp*

Figure 10: Comparison of reconstruction loss versus epochs between DGLGM-SGD and DGLGM-LVO.

5.2.6 Generalization ability and convergence analysis. In order to evaluate the generalization ability of the proposed model, we choose different training sets with different sizes {20%, 40%, 60%, 80%} and fix the same 20% as the testing set. We guarantee that small size training set is included in large size training set. Since Mult-VAE and aWAE obtains competitive performance among all baselines, the following comparisons and analysis focus on **DGLGM**, **DLGM**, Mult-VAE and aWAE. Figure 9 shows the NDCG@10 comparison in terms of different training set on four datasets. Since we have similar observations in terms of other metrics, we only show the results in NDCG@10. Obviously, the recommendation performance becomes better and better with the increasing of training data and **DGLGM** is superior to Mult-VAE and aWAE for all training set on four datasets. Note that *Yelp* is extremely sparse, Figure 9(d) shows that **DGLGM** significantly outperforms Mult-VAE and aWAE, especially for small size training sets (20%, 40%). This result demonstrates that **DGLGM** has a strong generalization ability.

Furthermore, we investigate the convergence property of the proposed local variational optimization LVO method. As mentioned before, the proposed LVO algorithm can obtain a tighter upper bound for **DGLGM** than original SGD optimization method. Firstly, We report the recommendation performance obtained via **DGLGM** with different optimization methods (SGD and LVO). As shown in Table 6, **DGLGM** with LVO outperforms **DGLGM** with SGD, which indicates LVO can estimate more precise model parameters and further improve recommendation performance. Furthermore, we visualize upper bounds on training and testing viewed as a function of epochs to analyze the influence of learning algorithm on convergence rate. As shown in Figure 10, we plot the reconstruct loss $c(\mathbf{x}_i, p_G(\mathbf{x}_i))$ versus epochs for LVO and SGD. Obviously, the proposed LVO method can obtain a lower upper bound than SGD and converges faster.

Table 6: Comparisons of DGLGM-SGD and DGLGM-LVO.

Metrics	Methods	*ML 20M*	*Netflix*	*Epinions*	*Yelp*
DGLGM-SGD	Recall@10	0.3376	0.3467	0.6789	0.5402
	NDCG@10	0.3384	0.2602	0.8503	0.7348
DGLGM-LVO	Recall@10	0.3397	0.3496	0.6824	0.5443
	NDCG@10	0.3396	0.2649	0.8529	0.7390

6 CONCLUSION

In this paper, we propose a deep generative model for recommendation considering both local and global properties of user preference (**DGLGM**) under the Wasserstein auto-encoder framework. To adaptively capture the complicated structures of the users' preference, the latent variables are characterized via local and global representations together, where the optimal dimensionality of each component is determined via the automatic relevance determination technique. (**DGLGM**) can be efficiently inferred by minimizing its penalized evidence lower bound with the aid of Stochastic Variational Inference technique. Meanwhile, we theoretically analyze its generalization error bounds to guarantee its performance in feedback data with diversity. The experiments have shown that **DGLGM** has the ability to output the high-quality recommended item.

ACKNOWLEDGMENTS

Liping Jing is the corresponding author. This work was supported in part by the National Natural Science Foundation of China under Grant 61822601, 61773050, and 61632004; the Beijing Natural Science Foundation under Grant Z180006; National Key Research and Development Program (2017YFC1703506); The Fundamental Research Funds for the Central Universities (2019JBZ110); Science and technology innovation planning foundation of colleges and Universities under the guidance of the Ministry of Education.

REFERENCES

[1] Martin Arjovsky, Soumith Chintala, and Léon Bottou. 2017. Wasserstein generative adversarial networks. In *Proceedings of ICML*. 214–223.

[2] Sanjeev Arora, Rong Ge, Yingyu Liang, Tengyu Ma, and Yi Zhang. 2017. Generalization and equilibrium in generative adversarial nets (gans). In *Proceeding of ICML*. JMLR. org, 224–232.

[3] James S. Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. 2011. Algorithms for hyper-parameter optimization. In *Proceedings of NIPS*. Curran Associates, Inc., 2546–2554.

[4] D.M. Blei and M.I. Jordan. 2006. Variational inference for Dirichlet process mixtures. *Bayesian Anal.* 42, 1 (2006), 121–144.

[5] Emmanuel J Candes and Yaniv Plan. 2011. Matrix completion with noise. *Proc. IEEE* 98, 6 (2011), 925–936.

[6] Emmanuel J Candès and Benjamin Recht. 2009. Exact matrix completion via convex optimization. *Foundations of Computational Mathematics* 9, 6 (2009), 717.

[7] Chao Chen, Dongsheng Li, Qin Lv, Junchi Yan, Stephen M Chu, and Li Shang. 2016. MPMA: mixture probabilistic matrix approximation for collaborative filtering.. In *Proceedings of the IJCAI*. 1382–1388.

[8] Chao Chen, Dongsheng Li, Qin Lv, Junchi Yan, Li Shang, and Stephen M Chu. 2017. GLOMA: Embedding global information in local matrix approximation models for collaborative filtering.. In *Proceedings of AAAI*. 1295–1301.

[9] Gintare Karolina Dziugaite, Daniel M. Roy, and Zoubin Ghahramani. 2015. Training generative neural networks via maximum mean discrepancy optimization. In *Proceedings of the UAI*. AUAI, 258–267.

[10] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Proceedings of NIPS*. 2672–2680.

[11] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of WWW*. 173–182.

[12] Matthew J Johnson, David K Duvenaud, Alex Wiltschko, Ryan P Adams, and Sandeep R Datta. 2016. Composing graphical models with neural networks for structured representations and fast inference. In *Proceeding of NIPS*. 2946–2954.

[13] Hyunjik Kim and Andriy Mnih. 2018. Disentangling by factorising. In *Proceedings of ICML (Proceedings of Machine Learning Research)*, Vol. 80. PMLR, 2649–2658.

[14] Diederik P Kingma and Jimmy Ba. 2014. Adam: a method for stochastic optimization. In *Proceedings of ICLR*.

[15] Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114* (2013).

[16] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 42, 8 (2009), 30–37.

[17] Rahul G Krishnan, Dawen Liang, and Matthew Hoffman. 2017. On the challenges of learning with inference networks on sparse, high-dimensional data. *Proceedings of AISTATS* (2017).

[18] Joonseok Lee, Seungyeon Kim, Guy Lebanon, and Yoram Singer. 2013. Local low-rank matrix approximation. In *Proceedings of ICML*. 82–90.

[19] Dawen Liang, Rahul G Krishnan, Matthew D Hoffman, and Tony Jebara. 2018. Variational Autoencoders for Collaborative Filtering. *Proceedings of WWW* (2018).

[20] Huafeng Liu, Jingxuan Wen, Liping Jing, and Jian Yu. 2019. Deep generative ranking for personalized recommendation. In *Proceedings of RecSys*. ACM, 34–42.

[21] Benjamin M. Marlin and Richard S. Zemel. 2009. Collaborative prediction and ranking with non-random missing data. In *Proceedings of RecSys*. 5–12.

[22] Radford M. Neal. 1996. *Bayesian learning for neural networks*. Springer. 456–456 pages.

[23] Xia Ning and George Karypis. 2012. SLIM: sparse linear methods for top-N recommender systems. In *Proceedings of ICDM*.

[24] Jim Pitman. 2002. *Combinatorial stochastic processes*. Technical Report. Technical Report.

[25] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of UAI*. AUAI Press, 452–461.

[26] Ruslan Salakhutdinov and Andriy Mnih. 2007. Probabilistic matrix factorization. In *Proceedings of International Conference on Machine Learning*. 880–887.

[27] Bo Song, Xin Yang, Yi Cao, and Congfu Xu. 2018. Neural collaborative ranking. In *Proceedings of CIKM*. ACM, 1353–1362.

[28] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research* 15, 1 (2014), 1929–1958.

[29] Ruoyu Sun and Zhi-Quan Luo. 2016. Guaranteed matrix completion via non-convex factorization. *IEEE Transactions on Information Theory* 62, 11 (2016), 6535–6579.

[30] Ilya Tolstikhin, Olivier Bousquet, Sylvain Gelly, and Bernhard Schoelkopf. 2018. Wasserstein auto-encoders. In *Proceedings of ICLR*.

[31] Cédric Villani. 2008. *Optimal transport: old and new*. Vol. 338. Springer Science & Business Media.

[32] Keqiang Wang, Wayne Xin Zhao, Hongwei Peng, and Xiaoling Wang. 2016. Bayesian probabilistic multi-topic matrix factorization for rating prediction.. In *Proceedings of the IJCAI*. 3910–3916.

[33] David P Wipf and Bhaskar D Rao. 2007. An empirical bayesian strategy for solving the simultaneous sparse approximation problem. *IEEE Transactions on Signal Processing* 55, 7 (2007), 3704–3716.

[34] Yao Wu, Christopher DuBois, Alice X Zheng, and Martin Ester. 2016. Collaborative denoising auto-encoders for top-n recommender systems. In *Proceedings of WSDM*. ACM, 153–162.

[35] Yao Wu, Xudong Liu, Min Xie, Martin Ester, and Qing Yang. 2016. CCCF: Improving collaborative filtering via scalable user-item co-clustering. In *Proceedings of WSDM*. ACM, 73–82.

[36] Menghao Zhang, Binbin Hu, Chuan Shi, and Bai Wang. 2017. Local low-rank matrix approximation with preference selection of anchor points. In *Proceedings of WWW*. 1395–1403.

[37] Jingbin Zhong and Xiaofeng Zhang. 2018. Wasserstein autoencoders for collaborative filtering. *arXiv preprint arXiv:1809.05662* (2018).