Neural Networks 126 (2020) 52–64

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

Recommendation via Collaborative Autoregressive Flows

Fan Zhou [a,*], Yuhua Mo [a], Goce Trajcevski [b], Kunpeng Zhang [c], Jin Wu [a], Ting Zhong [a]

[a] School of Information and Software Engineering, University of Electronic Science and Technology of China, China
[b] Department of Electrical and Computer Engineering, Iowa State University, Ames IA, United States of America
[c] Department of Decision, Operations & Information Technologies, University of Maryland, College Park MD, United States of America

ARTICLE INFO

Article history:
Received 14 August 2019
Received in revised form 12 January 2020
Accepted 9 March 2020
Available online 13 March 2020

Keywords:
Collaborative recommendation
Variational inference
Normalizing flows
Autoregressive flows
Generative models

ABSTRACT

Although it is one of the most widely used methods in recommender systems, Collaborative Filtering (CF) still has difficulties in modeling non-linear user–item interactions. Complementary to this, recently developed deep generative model variants (e.g., Variational Autoencoder (VAE)) allowing Bayesian inference and approximation of the variational posterior distributions in these models, have achieved promising performance improvement in many areas. However, the choices of variation distribution – e.g., the popular diagonal-covariance Gaussians – are insufficient to recover the true distributions, often resulting in biased maximum likelihood estimates of the model parameters.

Aiming at more tractable and expressive variational families, in this work we extend the flow-based generative model to CF for modeling implicit feedbacks. We present the Collaborative Autoregressive Flows (CAF) for the recommender system, transforming a simple initial density into more complex ones via a sequence of invertible transformations, until a desired level of complexity is attained. CAF is a non-linear probabilistic approach allowing uncertainty representation and exact tractability of latent-variable inference in item recommendations. Compared to the agnostic-presumed prior approximation used in existing deep generative recommendation approaches, CAF is more effective in estimating the probabilistic posterior and achieves better recommendation accuracy. We conducted extensive experimental evaluations demonstrating that CAF can capture more effective representation of latent factors, resulting in a substantial gain on recommendation compared to the state-of-the-art approaches.

1. Introduction

The large volumes of user–item interaction data facilitate personalized recommendations by presenting to users a set of unseen items (e.g., POIs, articles, movies, videos, etc.) they may like. Aiming at efficiently attracting more interests and clicks for various online services, several popular techniques emerged – e.g., Collaborative Filtering (CF) and Matrix Factorization (MF) (Koren, Bell, & Volinsky, 2009) – providing models for capturing similarity between items and users. While being capable of discovering latent features encoded in the user–item interactions, MF and its variants (e.g., probabilistic MF Salakhutdinov & Mnih, 2008) mainly focus on capturing the linear relations and may not fully leverage more intricate user–item interactions (Zhang, Yao, Sun, & Tay, 2019).

Recently, deep learning based recommendation models have shown great potential for learning effective representations, achieving state-of-the-art performance (He et al., 2017; Zhang et al., 2019). However, while desirable, the accuracy improvements are not enough for assuring high-quality recommender systems (and may even lead to over-specialization) — which shifted the researchers foci towards providing holistic recommendation experiences for the users, including factors like diversity, novelty, serendipity, fairness, and interpretability (Zhang et al., 2019).

Two recent paradigms have shown promises in improving recommendation experience:
(1) *Generative models* (e.g., Variational Autoencoder Kingma & Welling, 2014 and Generative Adversarial Nets (GANs) Goodfellow et al., 2014), used in computer vision and natural language processing have been introduced to recommender systems, producing models with promising results: Collaborative VAEs (Lee, Song, & Moon, 2017; Li & She, 2017; Liang, Krishnan, Hoffman, & Jebara, 2018) and IRGAN (Wang et al., 2017). While GAN-based models can synthesize large and realistic images and may assist in certain aspects of recommendations – e.g., improving ranking performance (Wang et al., 2017) and model robustness (He, Du, He, & Chua, 2018) and addressing the discrete sampling problem from the generator (Chae, Kang, Kim, & Lee, 2018) – they typically lack characterization of an explicit density, making it difficult to directly model user–item interactions.

* Corresponding author.
E-mail addresses: fan.zhou@uestc.edu.cn (F. Zhou), moyuhua95@gmail.com (Y. Mo), gocet25@iastate.edu (G. Trajcevski), kpzhang@umd.edu (K. Zhang), wj@uestc.edu.cn (J. Wu), zhongting@uestc.edu.cn (T. Zhong).

https://doi.org/10.1016/j.neunet.2020.03.010
0893-6080/© 2020 Elsevier Ltd. All rights reserved.

(2) *Likelihood-based methods* such as autoregressive models (e.g., LSTM Hochreiter & Schmidhuber, 1997 and GRU Chung, Gulcehre, Cho, & Bengio, 2014) and VAEs, arguably, are also promising for modeling implicit user–item interactions. While often used in capturing *sequential* user-item interactions (Hidasi, Karatzoglou, Baltrunas, & Tikk, 2016; Manotumruksa, Macdonald, & Ounis, 2018), they do not allow stochastic inference and thus fail to model interaction uncertainty. In contrast, VAE-based collaborative models can learn the complicated interactions of sparse implicit feedback and auxiliary information due to inherent capability of Bayesian inference in approximating the variational posterior distribution by leveraging the flexibility of neural networks architectures. CVAE (Li & She, 2017) was the first model to combine VAE and MF for learning probabilistic latent representation of context and implicit relationships between items and users, and several extensions (Chen & de Rijke, 2018; Karamanolakis et al., 2018; Lee et al., 2017; Liang et al., 2018) have proposed augmentations by incorporating auxiliary information, e.g., ratings, profiles, etc., that are associated with users and items.

Challenges: Despite incorporating Bayesian inference and uncertainty representation, collaborative VAE models are hard to optimize, largely due to the inherent biased variational inference. The models usually make a strong assumption that the posterior can be decomposed into multiple independent factors — while the variational inference needs to search the optimal posterior approximation within a parametric family of distributions which are usually specified as the prior. However, unless an exact family of distributions is chosen, the models are not flexible enough to match the true posterior and uncertainty of the recommendation. Such issues spurred the interest in enriching the variational posterior distribution by *normalizing flow* (NF) (Rezende & Mohamed, 2015) — a series of invertible transformations to desired variables with a simple initial distribution. Compared to VAEs and GANs, flow-based generative models have so far gained less attention — however, they have unique merits such as exact latent-variable inference and analytical likelihood evaluation (Kingma & Dhariwal, 2018).

Our approach: In this work, we introduce the *Collaborative Autoregressive Flows (CAF)* for modeling and estimating implicit feedbacks of user–item interactions. CAF extends the variational autoencoders with the capability of stochastic and amortized inference, and is more effective in producing flexible and tractable distributions, enabling better variational approximation and better recommendation performance. Our main contributions are:

- We propose a novel collaborative filtering-like model that leverages the Bayesian inference and autoregressive flows for item recommendation. While VAE-based networks can capture non-linear user–item interactions and generalize the data representation, autoregressive flows in our model are able to improve and interpret the representation learning of latent factors.
- The proposed model allows flexible and tractable probabilistic density estimation by exploiting the flows to approximate the true posterior of stochastic latent factors, largely alleviating the inference bias in existing Bayesian recommendation models and improving the recommendation accuracy.
- Hybridizing two autoregressive flows endows CAF with the benefits of both components, i.e., the efficiency on variational inference and sampling of data. We demonstrate that CAF bridges the gap between the latent factors with simple base distribution and the real data with complex distribution.
- We conduct extensive experiments and demonstrate that our CAF model achieves superiority over previous deep (generative) learning based models on three real-world datasets. We analyze the performance gain of our model by investigating the learned latent variables, showing that the number of transformations is a strong indicator interpreting the recommendation performance.

To our knowledge, CAF is the first work integrating Bayesian recommender with flow-based generative models, leveraging tractable posterior approximation. The rest of this paper is organized as follows. We discuss the related work in Section 2, and introduce the problem and provide necessary background in Section 3. The details of the proposed method are presented in Section 4, followed by comprehensive experimental evaluations in Section 5. We present concluding remarks in Section 6.

2. Related work

We now position our work in the context of related literature, from three main perspectives.

2.1. Deep recommendation models

神经网络在推荐系统中的应用

Recent advances in neural networks (NN) (Schmidhuber, 2015) have spurred research results introducing various deep learning techniques into recommender systems. Collaborative deep learning (CDL) and its variant (Wang, Wang and Yeung, 2015; Zhuang et al., 2017) integrate stacked denoising autoencoder (DAE) into probabilistic matrix factorization to learn item representations from side information. On another line, Neural collaborative filtering (NCF) (He et al., 2017) generalizes MF for collaborative filtering by a neural network tackling the limitation of linear interaction between MF. Thereafter, neural networks have been successfully applied to different recommendation scenarios such as session-based/sequential recommendation (Hidasi et al., 2016; Zhou, Wen, Zhang, Trajcevski, & Zhong, 2019), group recommendation (Yin et al., 2019), POI/trip recommendation (He, Qi, & Ramamohanarao, 2019; Zhou et al., 2019), social recommendation (Fan et al., 2019), TV Show recommendation (Cho, Lee, Han, Choi, & Kim, 2019), mentionee recommendation (Wang, Meng, Bian, Li and Yang et al., 2018) and so on, whereby various deep learning techniques such as attention mechanisms, RNNs and adversarial learning are widely employed for enhancing recommendation performance according to the contexts and tasks. Despite these NN-based and autoencoder-based models show promising performance, they are primarily restricted to learning representation of items, and thus are difficult for Bayesian inference due to lack of Bayesian nature.

2.2. Deep generative recommendation

生成式模型在推荐系统中的应用

Owing to their capabilities for learning joint density of data and learning meaningful features from large *unlabeled* datasets, deep generative models have gained a significant popularity in recent years. Part of their appeal is due to the fact that they are not necessarily task-specific but benefit many downstream solutions — including recommendation. Two representative deep generative models are Generative Adversarial Nets (GAN) (Goodfellow et al., 2014) and Variational AutoEncoders (VAE) (Kingma & Welling, 2014).

GAN has been widely used in recommender systems but are limited to assisting ranking or matching without density estimation. One of the most impressive works is IRGAN (Wang et al., 2017), which leverages adversarial learning to approximate the data distribution with the generator by estimating the relevance of item pairs given a particular user, and to distinguish whether the item pairs are from real data or generated samples with the discriminator. Later on, many GAN-based models (Chae et al., 2018; Fan et al., 2019; He et al., 2018; Wang et al., 2018; Xie, Li, Chen, Xu, & Zheng, 2019; Zhou et al., 2019) have been proposed focusing on different aspects of recommendation, e.g., improving the model robustness via perturbation with adversarial samples (He et al., 2018) or optimizing model parameter inference (Wang, Yin et al., 2018) with adversarial learning.

VAE (Kingma & Welling, 2014), in contrast, is a latent variable model that consists of an encoder and a decoder both parameterized by neural networks. Instead of directly performing maximum likelihood estimation on the intractable marginaAEl log-V, training is done by optimizing the tractable ELBO through amortized variational inference which can be made efficiently through the reparameterization trick (Kingma & Welling, 2014). The first VAE-based deep generative recommender system called collaborative variational autoencoder (CVAE) was proposed in Li and She (2017), which jointly models the generation of content and the rating information using vanilla VAE (Kingma & Welling, 2014) in a collaborative filtering (CF) setting. Subsequently, Lee et al. (2017) augmented CF with ladder VAE (Sønderby et al., 2016) and leveraged adversarial learning to regularize their proposed collaborative recommendation models. Liang et al. (2018) found that VAE model suffers from underfitting when modeling large, sparse, high-dimensional data (Krishnan, Liang, & Hoffman, 2018), and presented a multinomial conditional likelihood based VAE framework (Liang et al., 2018). Several recent works (Chen & de Rijke, 2018; Karamanolakis et al., 2018) extend the ideas of applying VAEs to CF-based recommendation but primarily focus on combining various auxiliary features and/or improving the latent factor representation learning. It is worthwhile to mention that a latest work (Vo & Soh, 2018) leverages VAE to learn users' latent interest space and generate plausible appealing new items that *do not exist* in the training set, although its main topic is out of the scope of this work.

2.3. Flow-based generative models

Flow-based generative models have gained less attention compared to GANs and VAEs — however, they enable tractable density estimation and exact objective optimization, in contrast to the intractable density estimation of VAE and the implicit losses of GANs. Two main families of existing flow-based models are *normalizing flows* (NF) and *autoregressive flows* (AF). Rezende and Mohamed (2015) was the first work introducing NF to the variational posterior estimation, enabling to represent a richer family of distribution and retains the log-likelihood of data tractable. Along these lines, NICE (Dinh, Krueger, & Bengio, 2015), Real-NVP (Dinh, Sohl-Dickstein, & Bengio, 2017) and Glow (Kingma & Dhariwal, 2018) are also NF-based models. The recently proposed Glow model has been shown to be very successful in generating high-quality images (Kingma & Dhariwal, 2018) and speech synthesis (Prenger, Valle, & Catanzaro, 2019). If a flow transformation in a NF is constructed with an *autoregressive* model, where each dimension of a vector variable is conditioned on its preceding dimensions, one can obtain a particular NF with autoregressive transformation. IAF (Kingma et al., 2016) and MAF (Papamakarios, Pavlakou, & Murray, 2017) can be categorized as AF-based density estimators, while PxielRNN (van den Oord, Kalchbrenner and Kavukcuoglu, 2016) and WaveNet (van den Oord et al., 2016) (and its improved version Parallel WaveNet van den Oord et al., 2018) are also examples of applying AF on high-quality image generation and high-fidelity audio synthesis, respectively.

Key differences: Our work differs from the above mentioned works in several ways. Compared to conventional CF-based recommendation models, we model the problem within a probabilistic recommendation setting which allows for Bayesian inference and capturing non-linear user–item interactions. Furthermore, our model has at least three key differences with the previous collaborative VAE recommendation models: (1) we derived a novel ELBO for item recommendation task and proposed a collaborative autoregressive flow model for improving the ELBO approximation; (2) we introduced a training method combining the effectiveness of inverse AF and masked AF to address the bias inference problem and efficiency issues in the settings of item recommendation; and (3) the interpretation of the model performance was presented, where we explained the working mechanisms and the learned representation of the proposed CAF model.

Table 1
Notations. $*$ indicates u or v.

Notation	Description
$\mathbf{R} \in \mathbb{R}^{m\times n}$	Matrix of user–item interaction.
$\mathbf{U} \in \mathbb{R}^{m\times d}$	Matrix of user representation.
$\mathbf{V} \in \mathbb{R}^{n\times d}$	Matrix of user representation.
m/n	The number of users/items.
d	The latent dimensionality of $\mathbf{U}$ and $\mathbf{V}$.
$\mathbf{u}^i/\mathbf{v}^j$	User/item representation vector.
$\mathbf{z}_u^K/\mathbf{z}_v^K$	User/item latent variable representation.
$\mathbf{u}_c/\mathbf{v}_c$	User/item collaborative information.
r^{ij}	The rating of user i over item j.
$\mathbf{r}^{i\cdot}$	The rating of user i over all items.
$\mathbf{r}^{\cdot j}$	Item j's rating scores from all users.
$\mathbf{z}_*$ and $p(\mathbf{z}_*)$	Latent factor and its prior.
θ and ϕ	Parameters of decoder and encoder.
$\mathcal{L}(*;\theta,\phi)$	Evidence lower bound (ELBO).
$p_\theta(*\|\mathbf{z}_*)$	Generative networks (decoder).
$q_\phi(\mathbf{z}_*\|*)$	Inference networks (encoder).
K	The number of transformations.
f_k	Invertible transformation function.

3. Preliminaries

We now introduce the basic terminology used throughout the paper, and preliminary concepts of normalizing flows.

Notation: Let the matrix $\mathbf{R}_{m\times n}$ denote the user–item interactions, where each entry r^{ij} represents the preference (i.e., rating score) of user i on item j. For implicit feedbacks, simply indicating whether user i has interacted with the item j or not — $\mathbf{R}$ becomes binary, i.e., each r^{ij} is either 1 or 0. We denote the user and item representation matrices respectively by $\mathbf{U} \in \mathbb{R}^{m\times d}$ and $\mathbf{V} \in \mathbb{R}^{n\times d}$, where d is the dimension of the latent space. We use a scalar u^i (v^j) to refer a particular user (item) and vector $\mathbf{u}^i \in \mathbf{U}$ ($\mathbf{v}^j \in \mathbf{V}$) to denote its latent factor representation. Following previous works (Lee et al., 2017; Li & She, 2017), we use auxiliary prior information associated with users (e.g., social, demographics, etc.) and items (e.g., spatio-temporal information, category, etc.) in the model training. Thus, the recommendation task becomes to predict the missing values in $\mathbf{R}$ by optimizing the representation learning of latent factor matrices $\mathbf{U}$ and $\mathbf{V}$ together with the known values in $\mathbf{R}$.

The notation used in this paper is summarized in Table 1.

Normalizing Flow (NF Rezende & Mohamed, 2015): is a framework for building flexible posterior distributions through an iterative procedure. The key idea is to transform a simple base distribution into a complex one by stacking a series of invertible mappings which, in theory, can approximate any complex distribution. Given a variable $\mathbf{z}^0$ with known probability distribution $p_0(\mathbf{z}^0)$ (e.g., Gaussian) and a chain of invertible transformations $f = [f_1, \ldots, f_K]$, $\mathbf{z}^K$ can be calculated by composing the transformations from f as:

$$\mathbf{z}^K = f_K(\mathbf{z}^{K-1}) = f_K(f_{K-1}(\mathbf{z}^{K-2})) \\ = f_K(f_{K-1}(\cdots f_1(\mathbf{z}^0))). \quad (1)$$

Since each $f_k \in f$ is invertible (i.e., $\mathbf{z}^{k-1} = f_k^{-1}(\mathbf{z}^k)$) and, according to the definition of probability, $\int p_k(\mathbf{z}^k) d\mathbf{z}^k = \int p_{k-1}(\mathbf{z}^{k-1}) d\mathbf{z}^{k-1} = 1$ — for a collection of variables $\mathbf{z}^0, \ldots, \mathbf{z}^K$ we can obtain the distribution $p_K(\mathbf{z}^K)$ in a more flexible manner as:

$$
\begin{aligned}
p_K(\mathbf{z}^K) &= p_{K-1}(\mathbf{z}^{K-1}) \left| \det \frac{d f_K^{-1}(\mathbf{z}^K)}{d\mathbf{z}^K} \right| \\
&= p_0(\mathbf{z}^0) \left| \det \frac{d\mathbf{z}^K}{d\mathbf{z}^0} \right|^{-1}, \qquad (2)
\end{aligned}
$$

where $\det \frac{df}{d\mathbf{z}}$ is the Jacobian determinant of f, and $p_0(\mathbf{z}^0)$ is the base (initial) distribution (e.g., Gaussian). When there is no ambiguity, for conciseness we will drop the subscript when denoting the distribution.

The path traversed by the random variables $\mathbf{z}^k = f_k(\mathbf{z}^{k-1})$ with initial distribution $p(\mathbf{z}^0)$ is called the *flow*, and the whole path formed by the successive distributions $p(\mathbf{z}^K)$ is the *normalizing flow*. To ensure tractability of Eq. (2): (1) the transformation f_k must be easy to invert; and (2) the determinant of its Jacobian must be easy to compute (Rezende & Mohamed, 2015). These constraints allow the transformation to go deeper by composing multiple instances of it, and the result will still be a valid normalizing flow. The log-likelihood of a distribution $q(\mathbf{z}^K)$ can be computed iteratively as:

$$
\log q(\mathbf{z}^K) = \log q(\mathbf{z}^0) - \sum_{k=1}^{K} \log \det \left| \frac{d\mathbf{z}^k}{d\mathbf{z}^{k-1}} \right|. \qquad (3)
$$

In Rezende and Mohamed (2015), a family of transformations have been introduced, among which the *planar flow* is defined as:

$$
f(\mathbf{z}) = \mathbf{z} + \mathbf{y}\delta(\mathbf{w}^\intercal \mathbf{z} + b), \qquad (4)
$$

where $\mathbf{y}, \mathbf{w} \in \mathbb{R}^d$ and $b \in \mathbb{R}$ are parameters, and δ is a suitable smooth non-linear activation function (e.g., tanh). According to the *Matrix determinant lemma* (Harville, 1998; Symeonidis & Zioupos, 2016), the Jacobian of this transformation is:

$$
\left| \det \frac{\partial f}{\partial \mathbf{z}} \right| = \left| 1 + \mathbf{y}^\intercal \delta'(\mathbf{w}^\intercal \mathbf{z} + b)\mathbf{w} \right|, \qquad (5)
$$

where δ' is the derivative activation and the Jacobian can be computed in $O(d)$ time — where d is the dimensionality of $\mathbf{z}$.

4. Methodologies

In this section we present the details of the proposed model — Collaborative Autoregressive Flows (CAF) for recommender systems. CAF is a generative latent variable model unifying collaborative information and auxiliary information generated by latent variables. It follows the main architecture of VAE-based recommendation models (Lee et al., 2017; Li & She, 2017; Liang et al., 2018) — however, we also introduce flow-based generative networks into the variational inference. As we will demonstrate, CAF allows more expressive and flexible posterior distribution than the existing works.

4.1. Model basics

Fig. 1 depicts the main components of CAF. As shown, there are two variational autoencoders in CAF for learning user and item representation, respectively.

In the previous collaborative VAE models (Lee et al., 2017; Li & She, 2017), the only auxiliary information considered was associated with items, in addition to the collaborative information. For instance, related works include spatio-temporal information of POIs (Ma, Zhang, Wang, & Liu, 2018), the word

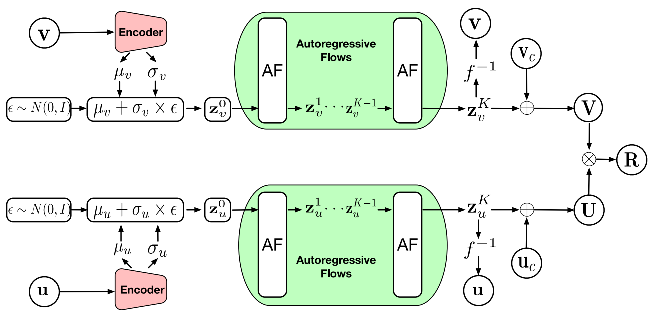


Fig. 1. Overview of our proposed model CAF.

frequency of scientific articles (Li & She, 2017), or the ratings and movie genres (Liang et al., 2018) in their models.

In this work, inspired by the most recent result in Chen and de Rijke (2018), we also incorporate information from the user's side, such as the visit-frequency for items or user-rating (i.e., the transpose of item-rating matrix) for item recommendation — cf. Fig. 1.

More specifically, the proposed CAF model constructs generative latent variable models for the auxiliary information and assigns latent variables $\mathbf{z}_u^i$ and $\mathbf{z}_v^j$ to each user and item, respectively. The side information associated with each user $\mathbf{u}^i$ and each item $\mathbf{v}^j$ are generated from their respective latent variables via the generation neural networks parameterized by θ:

$$
\mathbf{u}^i \sim p_\theta(\mathbf{u}^i | \mathbf{z}_u^i), \qquad \mathbf{v}^j \sim p_\theta(\mathbf{v}^j | \mathbf{z}_v^j), \qquad (6)
$$

where $\mathbf{u}^i$ and $\mathbf{v}^j$ are real-valued data (e.g., spatio-temporal information, ratings, demographics or word-frequency, etc.). These data are generated based on an autoregressive flow-based posterior distribution parameterized by the generation networks. In addition, we incorporate the collaborative information associated with both users $\mathbf{u}_c$ and items $\mathbf{v}_c$, which are drawn from the Gaussian distribution:

$$
\mathbf{u}_c \sim N(0, I), \qquad \mathbf{v}_c \sim N(0, I) \qquad (7)
$$

which are combined with latent variables to represent the user and the item:

$$
\mathbf{u}^i = \mathbf{z}_u^K + \mathbf{u}_c, \qquad \mathbf{v}^j = \mathbf{z}_v^K + \mathbf{v}_c. \qquad (8)
$$

Note that the above assumptions are similar to previous VAE-based recommendation models (Chen & de Rijke, 2018; Karamanolakis et al., 2018; Lee et al., 2017; Li & She, 2017; Liang et al., 2018; Xiao, Liang, Shen, & Meng, 2018). However, these works are different from each other in the choices of the auxiliary information. For example, Liang et al. (2018) only consider the rating information while Li and She (2017) include the item-side auxiliary information. Other recent works (e.g., Chen & de Rijke, 2018; Xiao et al., 2018) incorporate the user-side information based on the main architecture of CVAE (Li & She, 2017).

In accordance with the model in Fig. 1, the joint probability of CAF is factorized as:

$$
p(\mathbf{U}, \mathbf{V}, \mathbf{R}, \mathbf{Z}_u, \mathbf{Z}_v) = \prod_{i=1}^{m} \prod_{j=1}^{n} p(r^{ij} | \mathbf{u}^i, \mathbf{v}^j) \cdot p(\mathbf{u}^i | \mathbf{z}_u^i) p(\mathbf{v}^j | \mathbf{z}_v^j) p(\mathbf{z}_u^i) p(\mathbf{z}_v^j) \qquad (9)
$$

where $p(r^{ij} | \mathbf{u}^i, \mathbf{v}^j)$ defines the rating of user u^i on item v^j and is generated by the two decoder networks. Following previous works (Chen & de Rijke, 2018; Liang et al., 2018; Xiao et al., 2018), we assume that $p(r^{ij} | \mathbf{u}^i, \mathbf{v}^j)$ follows a Bernoulli distribution, which allows us to model the loss function of training user ratings as the logistic log-likelihood:

$$p(\mathbf{r}^{i\cdot}|\mathbf{u}^i, \mathbf{v}^j) = \sum_{j=1}^{n}(r^{ij}\log \hat{r}^{ij} + (1-r^{ij})\log(1-\hat{r}^{ij})) \tag{10}$$

where $\mathbf{r}^{i\cdot}$ is the rating of user u^i over all items and $\hat{r}^{ij}$ is the predicted results by the model — a multiple-layer perceptron (MLP) network in this case.

The marginal log-likelihood of representations for a user and an item is intractable to compute or differentiate directly for flexible generative models, especially for high-dimensional latent variables. Thus, we resort to variational inference by defining a parametric distribution over the latent variables, e.g., $q_\phi(\mathbf{z}_v|\mathbf{v})$ for items, and maximizing the evidence lower bound (ELBO) on the marginal log-likelihood of each observation (we omit the subscripts for simplicity):

$$\begin{aligned}\log p_\theta(\mathbf{v}) &= \mathbb{E}_{q_\phi(\mathbf{z}_v|\mathbf{v})}\log\left[\frac{p_\theta(\mathbf{v},\mathbf{z}_v)}{q_\phi(\mathbf{z}_v|\mathbf{v})}\right] + \mathrm{KL}\left[q_\phi(\mathbf{z}_v|\mathbf{v})\|p_\theta(\mathbf{z}_v|\mathbf{v})\right]\\ &\geq \mathbb{E}_{q_\phi(\mathbf{z}_v|\mathbf{v})}\left[\log p_\theta(\mathbf{v},\mathbf{z}_v) - \log q_\phi(\mathbf{z}_v|\mathbf{v})\right]\\ &\triangleq \mathcal{L}(\mathbf{v};\theta,\phi)\end{aligned} \tag{11}$$

There are numerous ways to optimize the ELBO, among which VAEs (Kingma & Welling, 2014) use a parametric inference network and reparameterization of $q_\phi(\mathbf{z}_v|\mathbf{v})$ to alternatively maximize the following reformulation: 损失函数

$$\mathcal{L}(\mathbf{v};\theta,\phi) = \mathbb{E}_{q_\phi}[\log p_\theta(\mathbf{v}|\mathbf{z}_v)] - \mathrm{KL}\left[q_\phi(\mathbf{z}_v|\mathbf{v})\|p(\mathbf{z}_v)\right] \tag{12}$$

where the expectation term is the one-by-one data reconstruction. Therefore, the objective of maximizing ELBO $\mathcal{L}(\mathbf{v};\theta,\phi)$ of $\log p_\theta(\mathbf{v})$ becomes to minimize the Kullback–Leibler (KL) divergence between the variational distribution $q_\phi(\mathbf{z}_v|\mathbf{v})$ (also referred to as the *encoder*) and the prior $p(\mathbf{z}_v)$ (which is always ≥ 0). Note that the above process is similar to the representation estimation for users, which is therefore omitted due to the lack of space. Also note that, for clarity, we will sometimes omit the parameters ϕ and θ in the subsequent formulae.

While the first term in Eq. (12) can be estimated with a Monte Carlo approximation, the KL divergence is usually computed with the reparameterization trick (Kingma & Welling, 2014) in previous VAE-based recommendation models (Chen & de Rijke, 2018; Karamanolakis et al., 2018; Lee et al., 2017; Li & She, 2017; Liang et al., 2018; Xiao et al., 2018). On the other hand, the prior $p(\mathbf{z}_v)$ is usually assumed to be a diagonal covariance Gaussian distribution. As a result, the encoder density $q_\phi(\mathbf{z}_v|\mathbf{v})$ would be regularized to be a Gaussian when $q_\phi(\mathbf{z}_v|\mathbf{v}) = p(\mathbf{z}_v)$ for all $\mathbf{z}_v$, which greatly limits the capability of the variational inference, due to the real posterior $p_\theta(\mathbf{z}_v|\mathbf{v})$ is much more complicated than a Gaussian in real case. Therefore, a poor variational approximation to the posterior can fail to reflect the right amount of uncertainty (Huang, Krueger, Lacoste, & Courville, 2018), resulting in inaccurate and unreliable recommendations. Our main purpose is therefore to optimize the proposed posterior $q_\phi(\mathbf{z}_*|*)$ (where $*$ refers to either u or v), in order to match the prior $p(\mathbf{z}_*)$, which is not assumed to be a user-agnostic distribution anymore, but is alternatively estimated with the flows in order to approximate the true posterior $p_\theta(\mathbf{z}_*|*)$.

4.2. Inference with autoregressive flows

Our CAF model differs from previous works (Lee et al., 2017; Li & She, 2017; Liang et al., 2018) in that we introduce a more flexible and effective posterior estimation module rather than the simple Gaussian assumption of the prior $p(\mathbf{z}_u)$ and $p(\mathbf{z}_v)$. This

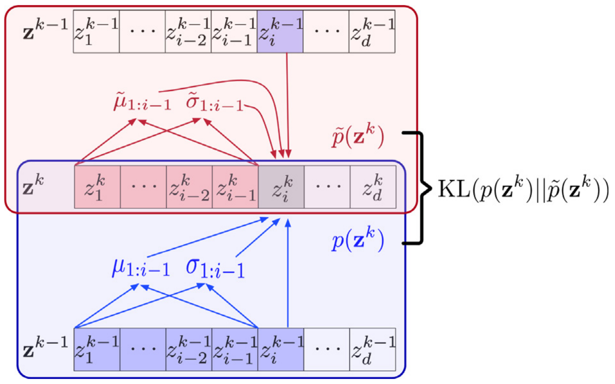

$\mathbf{z}^{k-1}$
z_1^{k-1} ⋯ z_{i-2}^{k-1} z_{i-1}^{k-1} z_i^{k-1} ⋯ z_d^{k-1}
$\tilde{\mu}_{1:i-1}$ $\tilde{\sigma}_{1:i-1}$
$\tilde{p}(\mathbf{z}^k)$
$\mathbf{z}^k$
z_1^k ⋯ z_{i-2}^k z_{i-1}^k z_i^k ⋯ z_d^k
$\mathrm{KL}(p(\mathbf{z}^k)||\tilde{p}(\mathbf{z}^k))$
$p(\mathbf{z}^k)$
$\mu_{1:i-1}$ $\sigma_{1:i-1}$

Fig. 2. Inference with the autoregressive flows. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

is achieved by the proposed flow-based generative models, as illustrated in the autoregressive decoders in Fig. 1.

An immediate solution is to utilize the normalizing flow (cf. Section 3) for improving the posterior approximation. However, this kind of flow actually modifies the initial density by applying a series of contractions and expansions over only a small volume in the original space. Since the RHS of Eq. (4) can be interpreted as a *single-neuron* MLP, it results in the information going through the single bottleneck. As the volume of the space grows exponentially with the number of dimension d, it requires many coupling layers to transform a simple base distribution into a complex one.

Autoregressive transformation: We extend the autoregressive transformations (Kingma et al., 2016; Papamakarios et al., 2017) to variational recommendations, which have been successfully used in image generation (van den Oord, Kalchbrenner et al., 2016) and audio synthesis (van den Oord, Dieleman et al., 2016). Autoregressive density estimation decomposes the joint density into a product of conditional densities, where each dimension depends on only the previous values. More specifically, let $\mathbf{z} \in \mathbb{R}^d$ (e.g., $\mathbf{z}_v$ or $\mathbf{z}_u$) be the latent variables we are interested in. With a slight abuse of symbols, we denote the latent variable in kth transformation as $\mathbf{z}^k$. The probability of observing $\mathbf{z}^k$ is given by:

$$p(\mathbf{z}^k) = \prod_{i=2}^{d} p(z_i^k|z_{1:i-1}^k) \tag{13}$$

where z_i^k is a one-dimensional density and is conditioned on the previous $i-1$ probabilities.

Recall that our aim is to transform a base distribution of $p(\mathbf{z}^0)$ with K flows to obtain a complex target distribution $p(\mathbf{z}^K)$, which could asymptotically approximate the real posterior $p_\theta(\mathbf{z}_*|*)$ via approximating the variational distribution $q_\phi(\mathbf{z}_*|*)$. Towards this goal, we need more powerful and tractable transformations enabling conditional factorization among latent variables. Formally, let $\mu_{1:i-1}$ and $\sigma_{1:i-1}$ be the mean and standard deviations – the output of neural networks $\mathtt{M}()$ and $\mathtt{S}()$ – of the previous dimensions $z_{1:i-1}^{k-1}$ in the base distribution. We introduce the autoregressive dependencies between the d-textitdimension variable $\mathbf{z}^k = \{z_1^k, \ldots, z_d^k\}$ in target distribution:

$$\mu_{1:i-1} = \mathtt{M}(\mathbf{z}_{1:i-1}^{k-1}), \qquad \sigma_{1:i-1} = \mathtt{S}(\mathbf{z}_{1:i-1}^{k-1}) \tag{14}$$

$$z_i^k = \mu_{1:i-1} + z_i^{k-1}\cdot\sigma_{1:i-1} \tag{15}$$

which means each dimension of the variable in *target* distribution depends only on the previous $i-1$ dimensions $\mathbf{z}_{1:i-1}^{k-1}$ in *base* distribution, as illustrated in the *bottom* (blue rectangle) of Fig. 2.

During each transformation, Eq. (15) describes a *generation* process whereby we generate a target distribution $p(\mathbf{z}^k)$ iteratively with *known* probabilistic density function of the base distribution $p(\mathbf{z}^{k-1})$. Therefore, this process is fast with the parallelization of GPU, since computations of each dimension z_i^k in the target distribution do not depend on each other. This is the main idea of *inverse autoregressive flows (IAF)* (Kingma et al., 2016), which uses *masked autoencoder* neural networks (Germain, Gregor, Murray, & Larochelle, 2015) – i.e., a special kind of neural networks which can output all values of mean and standard deviations simultaneously, while masking the weights so that the autoregressive property is preserved – to facilitate the sampling of $\mu_{1:i-1}$ and $\sigma_{1:i-1}$. To output the correct distribution at time-instant *i*, it infers what it would have output at previous timesteps $1 : i-1$ based on the noise inputs $\mathbf{z}_{1:i-1}^{k-1}$. This allows to output all $\mathbf{z}_{1:i-1}^{k}$ in parallel, given $\mathbf{z}_i^{k-1}$. The main procedure of the proposed autoregressive flows is described in Algorithm 1.

Regularization: We observe that the inverse process – the density estimation of $p(\mathbf{z}^{k-1})$ – is slow, since each dimension z_i^k is conditioned on the previous $i-1$ dimensions $z_{1:i-1}^k$. Thus the recovery computations $z_i^{k-1} = (z_i^k - \mu_{1:i-1})/\sigma_{1:i-1}$, i.e., the invertible transformations of Eq. (15), need to be sequentially evaluated and cannot be parallelized. To overcome the efficiency problem of density estimation mentioned above, we use a speed-up method inspired by *masked autoregressive flows (MAF)* (Papamakarios et al., 2017). Specifically, we evaluate the density of target distribution $p(\mathbf{z}^k)$ with

$$\tilde{\mu}_{1:i-1} = \mathtt{M}(\mathbf{z}_{1:i-1}^k), \qquad \tilde{\sigma}_{1:i-1} = \mathtt{S}(\mathbf{z}_{1:i-1}^k) \tag{16}$$

$$z_i^k = \tilde{\mu}_{1:i-1} + z_i^{k-1} \cdot \tilde{\sigma}_{1:i-1} \tag{17}$$

where $\tilde{\mu}_{1:i-1}$ and $\tilde{\sigma}_{1:i-1}$ are the output of the mean and standard deviations of the previous $i-1$ dimensions $\mathbf{z}_{1:i-1}^k$ in the *target* distribution, as depicted in the *top* (red rectangle) of Fig. 2. This means that we can evaluate $\tilde{\mu}_{1:i-1}$ and $\tilde{\sigma}_{1:i-1}$ from $z_{1:i-1}^k$ with a single pass by a masked autoencoder network. Thus, the density estimation of target distribution $p(\mathbf{z}^k)$ becomes more efficient.

By now we have introduced two models for modeling the target distributions $p(\mathbf{z}^k)$ and $\widetilde{p}(\mathbf{z}^k)$, both with the autoregressive transformations but different on the conditionals, i.e., $p(z_i^k|z_{1:i-1}^{k-1})$ and $\widetilde{p}(z_i^k|z_{1:i-1}^k)$, respectively. For $p(\mathbf{z}^k)$, it is transformed from the base distribution $p(\mathbf{z}^{k-1})$ and directly computed from random variables $\mathbf{z}_{1:i-1}^{k-1}$, similar to the inverse autoregressive flow (Kingma et al., 2016). Due to the autoregressive structure, the Jacobian of this transformation is a lower-triangular matrix with $\sigma_{1:i-1}$ on the diagonal, whose determinant is the product of the terms on the diagonal. As for $\widetilde{p}(\mathbf{z}^k)$, it is a forward density estimation by masked autoregressive flow (Papamakarios et al., 2017) conditioned on previous data variables $\mathbf{z}_{1:i-1}^k$. The Jacobian is also a lower-triangular matrix yet with $1/\sigma_{1:i-1}$ on the diagonal. Consequently, the flexibility of the distribution $p(\mathbf{z}^K)$ of the *K*th flow (final iterate), as well as its ability to closely fit to the true posterior $p_\theta(\mathbf{z}_*|*)$, increases with the depth of the chain. Even though each transformation (in both flows) is merely a *scale-and-shift*, the scale and shift can have *any* complex dependencies on previous variables decided by the expressiveness of the autoregressive models.

Density distillation: If $p(\mathbf{z}^k)$ and $\widetilde{p}(\mathbf{z}^k)$ share the same output distribution, they should be able to model the same multivariate distributions with the autoregressive flows in theory. Unfortunately, there is no guarantee that the output of the two models are exactly the same. In practice, they have different inductive biases and may vary greatly in their capacity to model certain autoregressive processes. Meanwhile, it is desirable to leverage both outputs for stabilizing the training process and for speeding up the convergence. Therefore, we minimize the KL divergence between the two output distribution:

Algorithm 1: Autoregressive Flows.

```
Input: The output of user encoder $\mathbf{z}_u^0$ and item encoder $\mathbf{z}_v^0$;
       The number of transformations $K$; The embedding
       size $d$.
Output: Latent variables $\mathbf{z}_u^K$ and $\mathbf{z}_v^K$.
1  foreach $k = 1, \cdots, K$ do
2      Sample $\mathbf{z}_u^{k-1}$ from base distribution $p(\mathbf{z}_u^{k-1})$;
3      Sample $\mathbf{z}_v^{k-1}$ from base distribution $p(\mathbf{z}_v^{k-1})$;
4      Initialize $\mu_1$, $\sigma_1$, $\tilde{\mu}_1$ and $\tilde{\sigma}_1$ randomly;
5      foreach $i = 2, \cdots, d$ do
6          Estimate $\mu_{1:i-1}$ and $\sigma_{1:i-1}$ via Eq. (14);
7          Compute $z_i^k$ via Eq. (15);
8      end
9      Compute $p(\mathbf{z}^k)$ via Eq. (13);
10     foreach $i = 2, \cdots, d$ do
11         Estimate $\tilde{\mu}_{1:i-1}$ and $\tilde{\sigma}_{1:i-1}$ via Eq. (16);
12         Compute $\tilde{z}_i^k$ via Eq. (17);
13     end
14     Compute $\widetilde{p}(\mathbf{z}^k)$ via Eq. (13);
15     Compute cross-entropy $\mathcal{H}(p(\mathbf{z}^k), \widetilde{p}(\mathbf{z}^k))$ via Eq. (19);
16     Compute entropy $\mathcal{H}(p(\mathbf{z}^k))$ via Eq. (20);
17     Minimize the $\mathrm{KL}(p(\mathbf{z}^k)||\widetilde{p}(\mathbf{z}^k))$ via Eq. (18).
18 end
```

$$\begin{aligned}
&\mathrm{KL}(p(\mathbf{z}^k)\|\widetilde{p}(\mathbf{z}^k)) \\
&= \sum_{z_i^k \in \mathbf{z}^k} p(z_i^k) \log \frac{1}{\widetilde{p}(z_i^k)} - \sum_{z_i^k \in \mathbf{z}^k} p(z_i^k) \log \frac{1}{p(z_i^k)} \\
&= \mathcal{H}(p(\mathbf{z}^k), \widetilde{p}(\mathbf{z}^k)) - \mathcal{H}(p(\mathbf{z}^k))
\end{aligned} \tag{18}$$

where $\mathcal{H}(p(\mathbf{z}^k), \widetilde{p}(\mathbf{z}^k))$ is the cross-entropy between the two distributions, and $\mathcal{H}(p(\mathbf{z}^k))$ is the entropy of the distribution $p(\mathbf{z}^k)$. The main purpose is for the distribution $p(\mathbf{z}^k)$ learned from base distribution to match the probability of the samples under the distribution $\widetilde{p}(\mathbf{z}^k)$ estimated from target distribution. This choice is inspired by the *probability density distillation* used in audio synthesis (van den Oord et al., 2018), which results in a great speed up (1000x faster) compared to the original WaveNet (van den Oord, Dieleman et al., 2016), a deep autoregressive generative model, without sacrificing data generation quality.

Specifically, the cross-entropy term can be computed via decomposing $p(\mathbf{z}^k)$ into conditional distributions by applying the chain rule:

$$\begin{aligned}
\mathcal{H}(p(\mathbf{z}^k), \widetilde{p}(\mathbf{z}^k)) &= -\sum_{z_i^k \in \mathbf{z}^k} p(z_i^k) \log \sum_{i=1}^{d} \widetilde{p}(z_i^k|z_{1:i-1}^k) \\
&= -\sum_{i=1}^{d} \sum_{z_i^k \in \mathbf{z}^k} p(z_{1:i-1}^k) p(z_{i:d}^k|z_{1:i-1}^k) \log \widetilde{p}(z_i^k|z_{1:i-1}^k) \\
&= \sum_{i=1}^{d} \mathbb{E} - \sum_{z_{1:i}^k} p(z_i^k|z_{1:i-1}^k) \log \widetilde{p}(z_i^k|z_{1:i-1}^k) \sum_{z_{i+1:d}^k} p \\
&\quad \times (z_{i+1:d}^k|z_{1:i}^k) \\
&= \sum_{i=1}^{d} \mathop{\mathbb{E}}_{p(z_{1:i-1}^k)} \mathcal{H}(p(z_i^k|z_{1:i-1}^k), \widetilde{p}(z_i^k|z_{1:i-1}^k)) \\
&= \sum_{i=1}^{d} \mathop{\mathbb{E}}_{z_i^{k-1} \sim \mathcal{N}(0,1)} \mathcal{H}(p(z_i^k|z_{1:i-1}^{k-1}), \widetilde{p}(z_i^k|z_{1:i-1}^k)),
\end{aligned} \tag{19}$$

Algorithm 2: Training with CAF.

```
Input: User's feature representation $\mathbf{u}$; User's collaborative
       information $\mathbf{u}_c$; Item's feature representation $\mathbf{v}$;
       Item's collaborative information $\mathbf{v}_c$.
Output: The model parameters $\theta$ and $\phi$.
1  Initialize $\theta$ and $\phi$ randomly;
2  while not converged do
3      $\epsilon \sim N(0, I)$;
4      Sample a batch of users, compute $\mathbf{z}_u^0 = \mu_u + \sigma_u \times \epsilon$;
5      Sample a batch of items, compute $\mathbf{z}_v^0 = \mu_v + \sigma_v \times \epsilon$;
6      Obtain latent variables $\mathbf{z}_u^K$ and $\mathbf{z}_v^K$ using autoregressive
       flows (Algorithm 1);
7      Estimate log-likelihoods $\log q(\mathbf{z}_v^K|\mathbf{v})$ and $\log q(\mathbf{z}_u^K|\mathbf{u})$ via
       Eq. (21);
8      $\mathbf{u} = \mathbf{z}_u^K + \mathbf{u}_c$; $\mathbf{v} = \mathbf{z}_v^K + \mathbf{v}_c$;
9      Update model parameters $\theta$ and $\phi$ by maximizing ELBO
       of Eq. (22).
10 end
```

where z_i^{k-1} are independent samples drawn from the Gaussian. Eq. (19) explicitly depends on samples from the base distribution $p(\mathbf{z}^{k-1})$ to estimate. Note that both $p(z_i^k|z_{1:i-1}^{k-1})$ and $\widetilde{p}(z_i^k|z_{1:i-1}^k)$ can be evaluated in parallel with masked autoencoders. More importantly, this estimator has a much lower variance than evaluating the samples solely from the distribution $p(\mathbf{z}^k)$ or $\widetilde{p}(\mathbf{z}^k)$ (van den Oord et al., 2018).

Alternatively, the entropy term in Eq. (18) can be estimated as:

$$\begin{aligned}\mathcal{H}(p(\mathbf{z}^k)) &= \mathop{\mathbb{E}}_{z_i^{k-1}\sim\mathcal{N}(0,1)}\left[\sum_{i=1}^{d} -\log p(z_i^k|z_{1:i-1}^{k-1})\right]\\ &= \mathop{\mathbb{E}}_{z_i^{k-1}\sim\mathcal{N}(0,1)}\left[\sum_{i=1}^{d}\log\sqrt{2\pi}\sigma_{1:i-1}\right]+\frac{1}{2}d, \end{aligned}\tag{20}$$

where $\sigma_{1:i-1} = \mathbb{S}(\mathbf{z}_{1:i-1}^{k-1})$. Eq. (20) follows because the fact that the entropy of a Gaussian $\mathcal{N}(\mu, \sigma)$ is $\log\sqrt{2\pi}\sigma + \frac{1}{2}$, which can be computed numerically without having to generate samples z_i^k.

Posterior estimation: We leverage the sampling efficiency of inverse autoregressive flows and the efficient density estimation with masked autoregressive flows to better approximate the real data distribution. Suppose we have K flows and let $\mathbf{z}_*^K$ be the output of the final flow (for users or items). The posterior can be evaluated following Eq. (3) as:

$$\begin{aligned}\log q(\mathbf{z}_*^K|*) &= \log q(\mathbf{z}_*^0|*) - \sum_{k=1}^{K}\log\det\left|\frac{d\mathbf{z}_*^k}{d\mathbf{z}_*^{k-1}}\right|\\ &= -\sum_{i=1}^{d}\Big(\frac{1}{2}\log(2\pi)+\frac{1}{2}\epsilon_i^2+\sum_{k=0}^{K}\log\sigma_{k,i}\Big)\end{aligned}\tag{21}$$

where $\mathbf{z}_*^K = \{\mathbf{z}_v, \mathbf{z}_u\}$; $*$ is either v or u, i.e., $\mathbf{z}_*^K$ denotes the vector representation of items or users. Here, $d\mathbf{z}_*^k/d\mathbf{z}_*^{k-1}$ is a triangular matrix with determinant $\prod_{i=1}^{d}\sigma_i$ on the diagonal. In addition, we assume a random sample $\epsilon \sim \mathcal{N}(0, I)$ and initialize the density estimation $\mathbf{z}_*^0 = \mu_* + \sigma_* \times \epsilon$, as illustrated in Fig. 1.

In summary, we train the item and user representation with CAF by maximizing the following new ELBO:

$$\begin{aligned}\mathcal{L}_{\text{CAF}}(\mathbf{u},\mathbf{v};\theta,\phi) =& \mathbb{E}_{\mathbf{z}_u,\mathbf{z}_v}\left[\log p(\mathbf{u},\mathbf{v},\mathbf{z}_u,\mathbf{z}_v) - \log q(\mathbf{z}_u,\mathbf{z}_v|\mathbf{u},\mathbf{v})\right]\\ =& \mathbb{E}_{\mathbf{z}_u,\mathbf{z}_v}\left[\log p(\mathbf{u},\mathbf{v},\mathbf{z}_u,\mathbf{z}_v) - \log q(\mathbf{z}_u|\mathbf{u})\right.\\ &\left. - \log q(\mathbf{z}_v|\mathbf{v})\right]\\ =& \mathbb{E}_{q(\mathbf{z}_u^0),q(\mathbf{z}_v^0)}\left[\log p(\mathbf{u},\mathbf{v},\mathbf{z}_u^K,\mathbf{z}_v^K) - \log q(\mathbf{z}_u^K)\right.\\ &\left. - \log q(\mathbf{z}_v^K)\right]\\ =& \mathbb{E}_{q(\mathbf{z}_u^0)}\left[\log p(\mathbf{u}|\mathbf{z}_u^K)\right] + \mathbb{E}_{q(\mathbf{z}_v^0)}\left[\log p(\mathbf{v}|\mathbf{z}_v^K)\right]\\ &- \mathbb{E}_{q(\mathbf{z}_v^0)}\left[\log q(\mathbf{z}_v^0)\right] + \mathbb{E}_{q(\mathbf{z}_u^0)}\left[\log p(\mathbf{z}_u^K)\right]\\ &+ \mathbb{E}_{q(\mathbf{z}_u^0)}\left[\sum_{k=1}^{K}\log\det\left|\frac{d\mathbf{z}_u^k}{d\mathbf{z}_u^{k-1}}\right|\right]\\ &- \mathbb{E}_{q(\mathbf{z}_u^0)}\left[\log q(\mathbf{z}_u^0)\right]\\ &+ \mathbb{E}_{q(\mathbf{z}_v^0)}\left[\sum_{k=1}^{K}\log\det\left|\frac{d\mathbf{z}_v^k}{d\mathbf{z}_v^{k-1}}\right|\right]\\ &+ \mathbb{E}_{q(\mathbf{z}_v^0)}\left[\log p(\mathbf{z}_v^K)\right]\end{aligned}\tag{22}$$

which is derived based on the fact that $\mathbf{z}_u$ and $\mathbf{z}_v$ are marginally independent given $\mathbf{u}$ and $\mathbf{v}$, and with two independent flows $q_\phi(\mathbf{z}_u|\mathbf{u}) := q(\mathbf{z}_u^K)$ and $q_\phi(\mathbf{z}_v|\mathbf{v}) := q(\mathbf{z}_v^K)$ (Eq. (22)). The training process of CAF is outlined in Algorithm 2.

5. Experiments

In this section, we describe the experimental settings and report the empirical evaluation results. To validate our model and technical contributions, we aim to answer the following questions:

- **Q1** How does CAF perform compared with the state-of-the-art recommendation models, especially recent Bayesian recommender systems?
- **Q2** Is the modeling of autoregressive flows helpful for learning more desirable posterior for item recommendation? If the answer is yes, then how does it work, and what is the price?
- **Q3** How do the key hyper-parameters (e.g., number of transformations K and the embedding size d) affect CAF's performance?

5.1. Datasets, baselines, evaluation protocols and settings

(1) Datasets: We conducted our experiments on three widely used benchmark datasets: MovieLens 1M (ML), CiteULike (CU), and LastFM (LF). 数据集

- **MovieLens 1M (ML).**[1] The user-movie ratings collected from a movie recommendation service. We converted all ratings of one or higher to positive feedback and thus interpret them as implicit feedback following Lee et al. (2017) and Liang et al. (2018).
- **CiteULike (CU).**[2] CU dataset consists of users and their libraries of articles obtained from CiteUlike. Each article has a title and abstract. We use the one collected by Wang and Blei (2011) but excluding users with fewer than 10 articles. The content information of the articles that we use is the concatenation of the titles and abstracts having a vocabulary size of 8000 following Lee et al. (2017) and Li and She (2017). The sparsity of the rating matrix is 99.8%, i.e., only 0.2% of the rating matrix have values.
- **LastFM (LF).**[3] LF is a social music sharing dataset containing listening records of users. We consider an artist that has been listened by the user as positive feedback and negative feedback for that has not been listened by the user.

1 https://grouplens.org/datasets/movielens/1M/.
2 http://www.citeulike.org/.
3 http://www.lastfm.com/.

Table 2
Statistics of datasets.

Dataset	#users	#items	#interaction	Sparsity
MovieLens	6040	3544	993,482	95.4%
CiteULike	5551	16,980	204,986	99.8%
LastFM	1892	17,632	92,834	97.3%

For each dataset, we randomly select 70% of the user–item interactions as the training set and the remaining as the validation (10%) and testing (20%) data. Table 2 shows the statistics of the datasets.

(2) ***Baselines***: We compare CAF with the following state-of-the-art methods:

- **BPR** (Rendle, Freudenthaler, Gantner, & Schmidt-Thieme, 2009) is one of widely used matrix factorization (MF) methods, which optimizes the latent factors with implicit feedback using a pairwise ranking objective function via stochastic gradient descent.
- **CDL** (Wang et al., 2015) is a joint Bayesian model learning auxiliary information and extracting latent features with stacked denoising autoencoder and collaborative filtering.
- **CVAE** (Li & She, 2017) is the first collaborative VAE-based item recommendation method, which uses vanilla VAE for incorporating content information from items into MF. We improve CVAE by incorporating the BPR model to refine the ranking results of CVAE, which is referred to as **CVAE-B**.
- **IRGAN** (Wang et al., 2017) is the first attempt applying GAN in recommender systems, which plays a minimax game to iteratively optimize the recommendation results.
- **VAE-AR** (Lee et al., 2017) models the implicit user feedback and the auxiliary information with VAE, and uses GAN (Goodfellow et al., 2014) to extract the low-dimensional representations influenced by the auxiliary information.
- **CLVAE** (Lee et al., 2017) is a conditional ladder VAE (Sønderby et al., 2016) based recommendation method which extends the CVAE with hierarchical VAE structure.
- **CFGAN** (Chae et al., 2018) is a GAN-based model whose generator tries to generate real-valued vectors – instead of a single discrete item in IRGAN – to prevent discriminator's confusion. It consistently improves the generator's performance compared to IRGAN.
- **MVAE** (Liang et al., 2018) is very similar to CVAE except that it uses multinomial conditional likelihood as the prior and that it does not incorporate the auxiliary information for recommendation.

For fair comparison, all baselines follow the settings in Lee et al. (2017) that explores item rating and movie genre as the item-side information for the ML dataset, and the word frequency for the CU dataset. For the LF dataset, we take social relationships as users' content information and use the tags as item-side information. Since the main contribution in this paper is not on combining various features, we exclude the comparisons with other VAE based methods (Chen & de Rijke, 2018; Karamanolakis et al., 2018; Xiao et al., 2018) that mainly leverage various content information (e.g., text reviews, user demographics, etc.) for improving recommendation performance.

(3) ***Evaluation protocols***: To evaluate our model, we adopt a variety of standard metrics which are widely used in previous works (Lee et al., 2017; Li & She, 2017; Liang et al., 2018), including P@N (Precision), R@N (Recall), nDCG@N (normalized discounted cumulative gain), and MAP@N (mean average precision). They demonstrate different aspects of the recommendation performance: precision and recall measure the number of correct recommendations, while nDCG and MAP take the rank of the recommended items into account.

(4) ***Parameter settings***: All models were implemented in tensorflow on Ubuntu 16.04 operating system. The machine is a server with two Intel(R) Xeon(R) CPU E5-2630, 128 GB memory, and a single GTX 2080Ti GPU. For all baselines, we represent the latent factors $\mathbf{z}_u$ and $\mathbf{z}_v$, users $\mathbf{u}$ and items $\mathbf{v}$ in a latent low-dimensional space of the same dimension of 100. For CAF, the embedding size is also 100 unless otherwise specified. All models are trained with the optimizer of Adam, where the dropout ratio is tuned with 9 different values: $\{0.0, 0.1, 0.2, \dots, 0.8\}$. The mini-batch size and learning rate were tuned according to performance in validation sets. For our CAF, the number of flows are tuned to 7, 5 and 5 for MovieLens, CiteUlike and LastFM, respectively — the choices will be discussed in detail later. Moreover, early stopping strategy is adopted for training all models, i.e., premature stopping if recall@10 on the validation set does not increase for 10 successive epochs.

5.2. Overall performance (Q1)

Table 3 shows the results of comparison to the existing state-of-the-art recommendation models, from which we can clearly observe that the proposed CAF model consistently yields the best performance on all metrics in all the datasets. Overall, our proposed model CAF, by leveraging flows for posterior approximation, significantly outperforms the baselines by a large margin (e.g., achieving 9.36%, 10.13%, 8.40% and 9.14% improvement, respectively, over the second best model on P@5, R@5, nDCG@10 and MAP@10 for the CU dataset), which proves the superiority of our model. By incorporating autoregressive transformation of latent space, CAF is capable of exploring the flexible and tractable variational posterior in an explicit way, so as to capture discernible latent representations more effectively. This justifies the effectiveness of autoregressive flows in CAF, i.e., specifying a sequence of expressive transformations to exact likelihood estimation.

Deep generative models (e.g., CVAE, VAE-AR, CLVAE, MVAE, etc.), including ours, usually outperform the shallow matrix factorization methods (e.g., BPR and CDL) and GAN-based models (e.g., IRGAN and CFGAN), which indicates the effectiveness of modeling the non-linearity and uncertainty of user–item interaction with Bayesian framework. Among the Bayesian models, two pioneering models CVAE and MVAE are not comparable, largely due to the vanilla VAE used in their models. For example, CVAE only learns the item-side representation while MVAE does not leverage the auxiliary information. An interesting observation is that we can largely improve the results of CVAE by simply incorporating the pairwise ranking, referred to CVAE-B. VAE-AR and CLVAE are two strong baselines which usually perform the second best in most cases. CLVAE benefits from learning richer flexibility of variational distributions with hierarchical VAE, while VAE-AR improves the collaborative VAE performance with latent representation enhancement using adversarial learning. However, they, as well as CVAE and MVAE, suffer from the problem of user-agnostic posterior approximation. As the first GAN-based model, IRGAN achieves better accuracy than BPR and CDL but usually shows inferior performance when compared with other models. Since IRGAN samples a single discrete item in each iteration, it is highly probable for the generator to sample items that are exactly the same as users' choices and therefore confuses the discriminator due to contradicting labels for the same item. CFGAN, in contrast, alleviates this issue by generating continuous vectors of plausible items which indeed improves the recommendation accuracy over IRGAN. However, both IRGAN and CFGAN cannot capture the uncertainty representation of users/items as Bayesian models can, and are therefore more prone to suffer the data sparsity problem, i.e., the user interaction data is extremely sparse (cf. Table 2).

Table 3
Recommendation performance comparison among different methods on three datasets. The best method is shown in **bold**, and the second best is shown as underlined. The number in brackets is the percentages of increases over the second best. A paired t-test is performed.

Dataset	Model	P@5	P@10	P@20	R@5	R@10	R@20	nDCG@10	MAP@10
MovieLens	BPR	0.1776	0.1582	0.1335	0.0742	0.1346	0.2188	0.4416	0.4513
	CDL	0.1412	0.1276	0.1129	0.0726	0.1278	0.2150	0.4375	0.4412
	CVAE	0.1532	0.1259	0.0964	0.0874	0.1385	0.2216	0.4465	0.4687
	CVAE-B	0.1825	0.1655	0.1436	0.0915	0.1451	0.2314	0.4502	0.4896
	IRGAN	0.1459	0.1352	0.1240	0.0753	0.1281	0.2163	0.4391	0.4467
	VAE-AR	0.1964	0.1708	0.1424	0.0979	0.1593	0.2475	0.4706	0.5156
	CLVAE	0.1981	0.1732	0.1442	0.0969	0.1596	0.2496	0.4701	0.5149
	CFGAN	0.1785	0.1667	0.1398	0.0927	0.1506	0.2380	0.4562	0.4903
	MVAE	0.1897	0.1681	0.1439	0.0942	0.1531	0.2407	0.4643	0.5016
	CAF	**0.2139*** **(7.39%)**	**0.1855*** **(6.63%)**	**0.1547*** **(6.79%)**	**0.1069*** **(8.42%)**	**0.1685*** **(5.28%)**	**0.2702*** **(7.62%)**	**0.4856*** **(3.08%)**	**0.5461*** **(5.59%)**
CiteULike	BPR	0.0885	0.0725	0.0526	0.0824	0.1356	0.1931	0.3201	0.2568
	CDL	0.1008	0.0849	0.0708	0.1047	0.1643	0.2552	0.3514	0.2854
	CVAE	0.1461	0.1221	0.0946	0.1291	0.1919	0.2725	0.3750	0.3287
	CVAE-B	0.1627	0.1436	0.1204	0.1482	0.2153	0.2863	0.4201	0.3648
	IRGAN	0.1015	0.0864	0.0696	0.1140	0.1725	0.2573	0.3502	0.3006
	VAE-AR	0.1322	0.1079	0.0830	0.1172	0.1819	0.2649	0.3628	0.3109
	CLVAE	0.1447	0.1191	0.0911	0.1229	0.1881	0.2679	0.3724	0.3216
	CFGAN	0.1412	0.1189	0.0865	0.1256	0.1842	0.2653	0.3538	0.3205
	MVAE	0.1546	0.1223	0.0994	0.1376	0.2027	0.2753	0.3620	0.3434
	CAF	**0.1795*** **(9.36%)**	**0.1576*** **(8.87%)**	**0.1326*** **(9.20%)**	**0.1649*** **(10.13%)**	**0.2347*** **(8.27%)**	**0.3152*** **(9.17%)**	**0.4587*** **(8.40%)**	**0.4015*** **(9.14%)**
LastFM	BPR	0.2853	0.2596	0.2274	0.2614	0.3156	0.3917	0.5249	0.4596
	CDL	0.3102	0.2785	0.2416	0.2916	0.3436	0.4457	0.5612	0.5247
	CVAE	0.3361	0.3108	0.2820	0.3148	0.3622	0.4653	0.6079	0.5564
	CVAE-B	0.3543	0.3238	0.3056	0.3274	0.3694	0.4765	0.6312	0.5673
	IRGAN	0.3068	0.2901	0.2746	0.3002	0.3516	0.4487	0.5776	0.5319
	VAE-AR	0.3615	0.3318	0.3017	0.3312	0.3741	0.4826	0.6328	0.5718
	CLVAE	0.3671	0.3297	0.3139	0.3386	0.3826	0.4908	0.6439	0.5836
	CFGAN	0.3226	0.3065	0.2803	0.3117	0.3523	0.4580	0.6124	0.5549
	MVAE	0.3418	0.3146	0.2928	0.3204	0.3581	0.4603	0.6253	0.5627
	CAF	**0.3804*** **(3.50%)**	**0.3497*** **(4.23%)**	**0.3267*** **(3.92%)**	**0.3578*** **(5.37%)**	**0.4217*** **(4.49%)**	**0.5139*** **(2.70%)**	**0.6618*** **(2.70%)**	**0.6092*** **(4.20%)**

*Indicates statistical significance ($p < 0.001$).

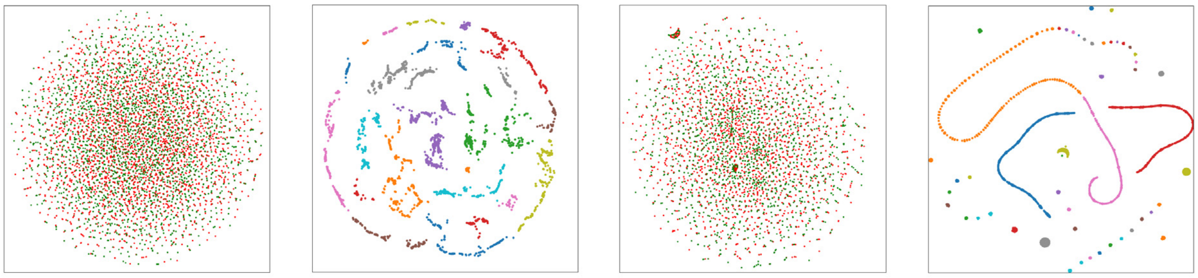

(a) User representation of CVAE. (b) User representation of CAF. (c) Item representation of CVAE. (d) Item representation of CAF.

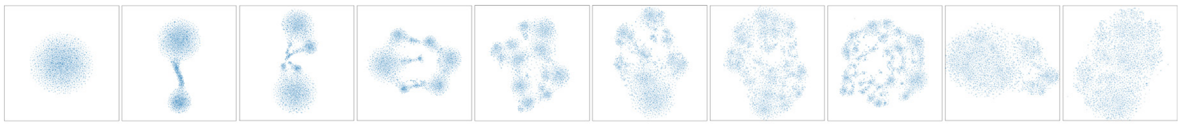
Fig. 3. Visualization of the learned user and item representation for MovieLens; $K = 7$.

Although performing Bayesian inference, previous VAE-based recommendation models may incur larger inference gaps and enlarge underfitting due to the amortized inference alone used for posterior approximation. More importantly, these methods usually approximate an improper pre-assumed distribution, e.g., the usual choice of diagonal-covariance Gaussian, and thus are subjected to a heavy bias inference problem. In contrast, our CAF model largely alleviates this problem benefiting from the introduced autoregressive flows with tractable posterior approximation. Intuitively, this justifies the use of models with more exact likelihoods and flexible approximation techniques (i.e., flows) in deep generative recommender systems.

5.3. Study of CAF (Q2)

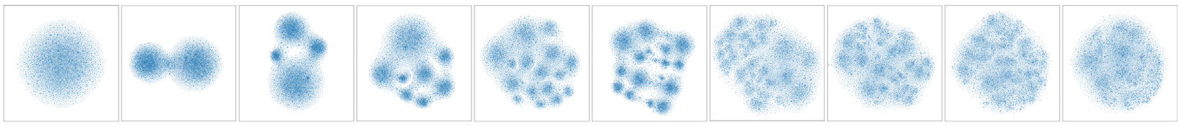
The main purpose of CAF is to learn more useful latent representations which are the crux for improving the recommendation results, especially for deep learning based recommender systems (He et al., 2017; Li et al., 2017; Liang et al., 2018). Moreover, disentangled representation is generally considered to contain interpretable semantic information and reflect separate factors of variation in the data (Higgins et al., 2017; John, Mou, Bahuleyan, & Vechtomova, 2019), which could improve the subsequent tasks such as recommendation. Here, we qualitatively analyze and interpret the performance of CAF.

User/Item representation: Fig. 3 plots the learned user and item representation by CVAE and CAF, where we extract the latent representation of the users **u** and the items **v** (both in 100 dimensions) obtained from the two models, and then apply t-SNE (Laurens van der Maaten, 2008) to reduce the representation to 2D for visualization. We can clearly observe the clustering phenomena learned by CAF, where the disentanglement of latent factors is well reflected in the embedding space, that is, similar users and items are embedded into the near part of the space.



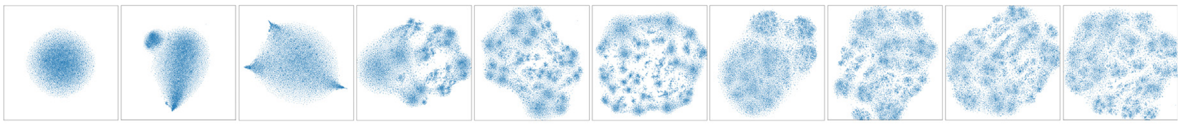
(a) $K = 0$ (b) $K = 1$ (c) $K = 2$ (d) $K = 3$ (e) $K = 4$ (f) $K = 5$ (g) $K = 6$ (h) $K = 7$ (i) $K = 8$ (j) $K = 9$

Fig. 4. Visualization of the hidden variables $\mathbf{z}_v$ (item-side) on MovieLens. $K = 0$ means there are no flows.



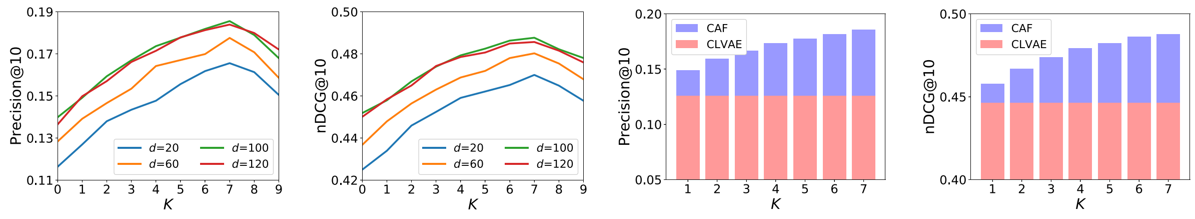
(a) $K = 0$ (b) $K = 1$ (c) $K = 2$ (d) $K = 3$ (e) $K = 4$ (f) $K = 5$ (g) $K = 6$ (h) $K = 7$ (i) $K = 8$ (j) $K = 9$

Fig. 5. Visualization of the hidden variables $\mathbf{z}_v$ (item-side) on CiteULike. $K = 0$ means there are no flows.



(a) $K = 0$ (b) $K = 1$ (c) $K = 2$ (d) $K = 3$ (e) $K = 4$ (f) $K = 5$ (g) $K = 6$ (h) $K = 7$ (i) $K = 8$ (j) $K = 9$

Fig. 6. Visualization of the hidden variables $\mathbf{z}_v$ (item-side) on LastFM. $K = 0$ means there are no flows.



(a) Precision. (b) nDCG. (c) Precision (d=100). (d) nDCG (d=100).

Fig. 7. Impact of K on precision and nDCG for MovieLens.

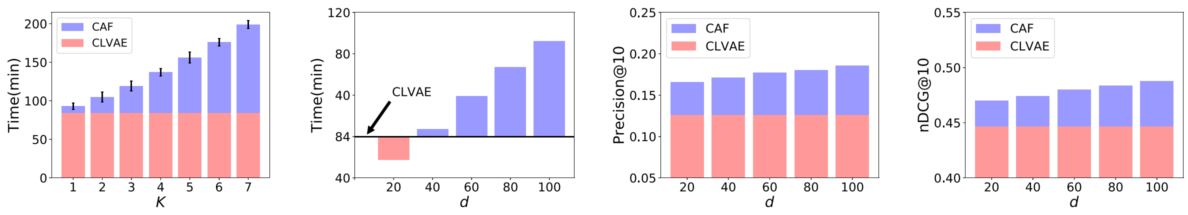
In contrast, CVAE, as well as other VAE-based models (omitted due to space limit), exhibit the Gaussian-style representation. The benefits of such clustering phenomena can be understood intuitively, i.e., the more separable the representation, the easier for the models to discriminate the spatially adjacent ones, which, consequently, are more prone to making better recommendation. This property therefore allows CAF to capture multi-modal target densities with higher accuracy, to increase expressiveness for better variational inference, and to more accurately evaluate the likelihood of samples drawn from a posterior distribution.

Latent factor visualization: The disentangled representation learned by CAF is the outcome of using autoregressive flows to approximate the true posterior. This argument can be proved by investigating the latent variables learned by the CAF models, or more precisely, by the autoregressive flows in CAF. Figs. 4–6 depict the latent variables $\mathbf{z}_v$ transformed by the flows in CAF on three datasets. Although, in theory, more transformations could approximate more complicated distribution, a smaller value is enough for our model. On ML dataset, one can clearly observe that the most distinct representation is obtained when $K = 7$, after which the latent factors become twisted again. In the case of CU and LF datasets, 5 transformations are enough to attain the best visualization. This observation is important since the disentangled representations are inherently more interpretable and the degree of disentanglement can potentially be used to guide the training procedure of the flow-based recommender systems.

Through theoretically normalizing flows are universal approximators that are flexible enough to represent any function arbitrarily well, they may suffer the problem of limited expressivity in practice (Papamakarios, Nalisnick, Rezende, Mohamed, & Lakshminarayanan, 2019). This is because a single affine autoregressive transformation of a multivariate Gaussian would still be a Gaussian. Although stacking multiple affine autoregressive layers can in theory obtain more expressive representation, a large number of flows usually make the inference network very deep and harder to train, empirically resulting in suboptimal performance as has been observed in image generation (Huang et al., 2018; Kingma et al., 2016; Papamakarios et al., 2017). This also explains why flow-based models, including ours, would suffer the problem of overfitting when stacking more layers. This result also raises an open question left for our future work, i.e., how to improve the performance of CAF or, more precisely, how to choose the optimal parameter K before training.

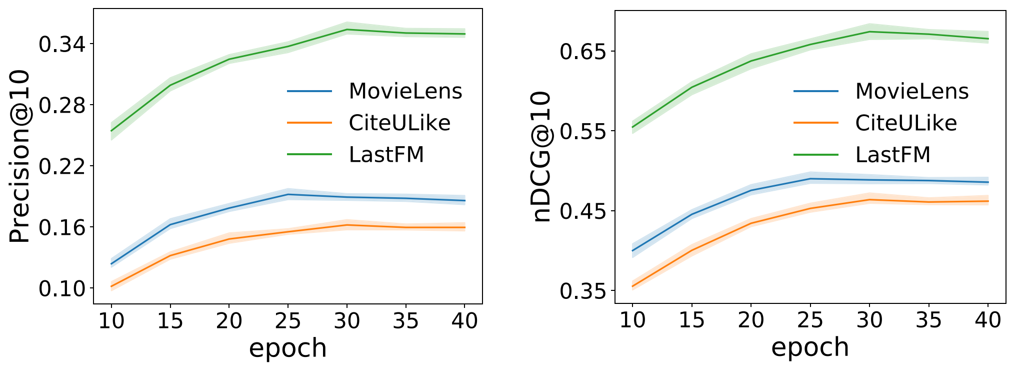
5.4. Effect of parameters (Q3)

Impact of K: From the visualization of latent space of the data, we hypothesize that the more separable the latent variable, the better the recommendation performance of CAF. To validate our assumption, we conduct experiments to investigate the impact of K (the number of transformations) on the recommendation performance. Figs. 7(a) and 7(b) illustrate the impact of K on



(a) Efficiency vs. K (d=100). (b) Efficiency vs. d (K=7). (c) Precision (K=7). (d) nDCG (K=7).

Fig. 8. CAF Efficiency on MovieLens.



(a) Precision. (b) nDCG.

Fig. 9. Convergence of CAF.

precision and nDCG for the ML dataset (we omitted the figures on other metrics and datasets due to the similar trends and lack of space), where we can see that the results are consistent with Fig. 4 — i.e., the best performance is achieved when $K = 7$. The performance of CAF would degrade to the same level of other Bayesian recommender models when removing the autoregressive flows (i.e., $K = 0$). In addition, we also observe that CAF beats CLVAE with only a few transformations as illustrated in Figs. 7(c) and 7(d). This result proves our conjecture that the number of transformations is a strong indicator of our CAF model, and, more importantly, can be used to interpret the performance gain with the visualization of the latent variables.

Efficiency: We note that there are extra computational overheads of flow-based generative models, due to the demands on the latent transformation. While it allows tractable and better posterior approximation, the computation of CAF increases linearly with the size of K, as depicted in Fig. 8(a). We also note that it has avoided the main complexity of data generation and density estimation with the density distillation as explained in Section 4.2. Though the transformation requires complex dependencies on previous dimensions of latent factors, a small number of flows can substantially improve the performance of Bayesian recommendation systems as shown in Figs. 7(c) and 7(d) — which provide flexible choices on balancing the performance and efficiency in real applications.

Effect of dimensionality d: Two possible ways can potentially alleviate the computational overhead problem. The simple one is to reduce the dimensionality of $\mathbf{z}_*$ ($* = \{u, v\}$). This can be verified by the results in Fig. 8(b), where we vary the dimensionality of $\mathbf{z}_*$. As it shows, the complexity of CAF increases linearly — due to the autoregressive dependencies of latent factors. Therefore, we need to find a proper size of embedding in order to balance the trade-off between the performance and the complexity for CAF model. Hopefully, although the performance may decrease with reducing the dimension, CAF, with a smaller size of d (e.g., 20 and 40), can outperform CLVAE, as shown in Figs. 8(c) and 8(d) — where the embedding size for CLVAE is always 100. An alternative choice is to replace the autoregressive flows with simpler non-autoregressive transformations such as planar flow and radial flow (Rezende & Mohamed, 2015). However, these are not competitive on posterior approximation since the transformations used act as a bottleneck, as observed in image generation (Kingma & Dhariwal, 2018; Kingma et al., 2016; Papamakarios et al., 2017).

Convergence: Lastly, Fig. 9 depicts the training procedure of CAF on three datasets, from which we can see that CAF converges within a few epochs, e.g., around 30 for the CU and LF datasets and 25 for the ML dataset. This is reasonable to expect, since the main computational complexity of CAF stems from the posterior transformation and density estimation which, in turn, is linear to the dimensionality of the latent factors and the number of stacking flows.

6. Conclusions and future work

In this work, we presented CAF — a novel approach for collaborative variational recommendation. Our collaborative flow algorithm addresses the biased inference problem by leveraging the Bayesian inference for probabilistic recommendation and autoregressive flows for flexible posterior approximation. CAF is a flow-based generative approach capable of alleviating the agnostic posterior estimation problem inherent in existing Bayesian recommendation methods. In addition to the new methodology that ensures higher item recommendation accuracy, we introduced the use of tractable transformations which, when it comes to training, successfully explain the recommendation behavior and provide insights of the interpretability of the recommendation performance. This, in turn, can be beneficial not only for recommender systems but also other latent factor-based applications. Our future work will focus on two main aspects: incorporating of temporal fluctuations of the rankings from historic data, and investigating the trade-offs between improving the overall efficiency of CAF and its effectiveness.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was supported by National Natural Science Foundation of China (Grant Nos. 61602097 and 61472064), National Science Foundation, United States of America grants Information and Intelligent Systems (III) 1213038 and Computer and Network Systems (CNS), United States of America 1646107 and ONR, United States of America grant N00014-14-10215.

References

Chae, D.-K., Kang, J.-S., Kim, S.-W., & Lee, J.-T. (2018). Cfgan: A generic collaborative filtering framework based on generative adversarial networks. In *ACM international conference on information and knowledge management (CIKM)*.
Chen, Y., & de Rijke, M. (2018). A collective variational autoencoder for top-n recommendation with side information. In *DLRS@RecSys*.
Cho, K.-J., Lee, Y.-C., Han, K., Choi, J., & Kim, S.-W. (2019). No, that's not my feedback: Tv show recommendation using watchable interval. In *International conference on data engineering (ICDE)*.
Chung, J., Gulcehre, C., Cho, K., & Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv:1412.3555.
Dinh, L., Krueger, D., & Bengio, Y. (2015). Nice: Non-linear independent components estimation. In *International conference on learning representations (ICLR)*.
Dinh, L., Sohl-Dickstein, J., & Bengio, S. (2017). Density estimation using real NVP. In *International conference on learning representations (ICLR)*.
Fan, W., Derr, T., Ma, Y., Wang, J., Tang, J., & Li, Q. (2019). Deep adversarial social recommendation. In *International joint conference on artificial intelligence (IJCAI)* (pp. 1351–1357).
Germain, M., Gregor, K., Murray, I., & Larochelle, H. (2015). Made: Masked autoencoder for distribution estimation. In *International conference on machine learning (ICML)*.
Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., et al. (2014). Generative adversarial nets. In *Annual conference on neural information processing systems (NIPS)*.
Harville, D. (1998). *Matrix algebra from a statistician's perspective*. Taylor & Francis Group.
He, X., Du, X., He, Z., & Chua, T.-S. (2018). Adversarial personalized ranking for recommendation. In *International conference on research and development in information retrieval (SIGIR)*.
He, X., Liao, L., Zhang, H., Nie, L., Hu, X., & Chua, T.-S. (2017). Neural collaborative filtering. In *International conference on world wide web (WWW)*.
He, J., Qi, J., & Ramamohanarao, K. (2019). A joint context-aware embedding for trip recommendations. In *International conference on data engineering (ICDE)*.
Hidasi, B., Karatzoglou, A., Baltrunas, L., & Tikk, D. (2016). Session-based recommendations with recurrent neural networks. In *International conference on learning representations (ICLR)*.
Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinick, M., Mohamed, S., & Lerchner, A. (2017). Beta-vae: Learning basic visual concepts with a constrained variational framework. In *International conference on learning representations (ICLR)*.
Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*(8), 1735–1780.
Huang, C.-W., Krueger, D., Lacoste, A., & Courville, A. C. (2018). Neural autoregressive flows. In *International conference on machine learning (ICML)*.
John, V., Mou, L., Bahuleyan, H., & Vechtomova, O. (2019). Disentangled representation learning for non-parallel text style transfer. In *Proceedings of the conference of the association for computational linguistics (ACL)* (pp. 424–434).
Karamanolakis, G., Cherian, K. R., Narayan, A. R., Yuan, J., Tang, D., & Jebara, T. (2018). Item recommendation with variational autoencoders and heterogenous priors. In *DLRS@RecSys*.
Kingma, D. P., & Dhariwal, P. (2018). Glow: Generative flow with invertible 1x1 convolutions. In *Annual conference on neural information processing systems (NIPS)*.
Kingma, D. P., Salimans, T., Józefowicz, R., Chen, X., Sutskever, I., & Welling, M. (2016). Improving variational autoencoders with inverse autoregressive flow. In *Annual conference on neural information processing systems (NIPS)*.
Kingma, D. P., & Welling, M. (2014). Auto-encoding variational Bayes. In *International conference on learning representations (ICLR)*.
Koren, Y., Bell, R., & Volinsky, C. (2009). Matrix factorization techniques for recommender systems. *Computer*.
Krishnan, R. G., Liang, D., & Hoffman, M. D. (2018). On the challenges of learning with inference networks on sparse, high-dimensional data. In *The international conference on artificial intelligence and statistics (AISTATS)*.
Lee, W., Song, K., & Moon, I.-C. (2017). Augmented variational autoencoders for collaborative filtering with auxiliary information. In *ACM international conference on information and knowledge management (CIKM)*.
Li, J., Ren, P., Chen, Z., Ren, Z., Lian, T., & Ma, J. (2017). Neural attentive session-based recommendation. In *ACM international conference on information and knowledge management (CIKM)*.
Li, X., & She, J. (2017). Collaborative variational autoencoder for recommender systems. In *Proceedings of the ACM SIGKDD international conference on knowledge discovery & data mining (KDD)*.
Liang, D., Krishnan, R. G., Hoffman, M. D., & Jebara, T. (2018). Variational autoencoders for collaborative filtering. In *international conference on world wide web (WWW)*.
Ma, C., Zhang, Y., Wang, Q., & Liu, X. (2018). Point-of-interest recommendation: Exploiting self-attentive autoencoders with neighbor-aware influence. In *ACM international conference on information and knowledge management (CIKM)*.
Laurens van der Maaten, G. H. (2008). Visualizing data using t-sne. *Journal of Machine Learning Research (JMLR)*, *9*(Nov), 2579–2605.
Manotumruksa, J., Macdonald, C., & Ounis, I. (2018). A contextual attention recurrent architecture for context-aware venue recommendation. In *International conference on research and development in information retrieval (SIGIR)*.
van den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., et al. (2016). Wavenet: A generative model for raw audio. arXiv preprint arXiv:1609.03499.
van den Oord, A., Kalchbrenner, N., & Kavukcuoglu, K. (2016). Pixel recurrent neural networks. In *International conference on machine learning (ICML)*.
van den Oord, A., Li, Y., Babuschkin, I., Simonyan, K., Vinyals, O., Kavukcuoglu, K., et al. (2018). Parallel wavenet: Fast high-fidelity speech synthesis. In *International conference on machine learning (ICML)*.
Papamakarios, G., Nalisnick, E., Rezende, D. J., Mohamed, S., & Lakshminarayanan, B. (2019). Normalizing flows for probabilistic modeling and inference. arXiv preprint arXiv:1912.02762.
Papamakarios, G., Pavlakou, T., & Murray, I. (2017). Masked autoregressive flow for density estimation. In *Annual conference on neural information processing systems (NIPS)*.
Prenger, R., Valle, R., & Catanzaro, B. (2019). Waveglow: A flow-based generative network for speech synthesis. In *IEEe international conference on acoustics, speech and signal processing (ICASSP)* (pp. 3617–3621).
Rendle, S., Freudenthaler, C., Gantner, Z., & Schmidt-Thieme, L. (2009). Bpr: Bayesian personalized ranking from implicit feedback. In *Conference on uncertainty in artificial intelligence (UAI)*.
Rezende, D. J., & Mohamed, S. (2015). Variational inference with normalizing flows. In *International conference on machine learning (ICML)*.
Salakhutdinov, R., & Mnih, A. (2008). Probabilistic matrix factorization. In *Annual conference on neural information processing systems (NIPS)*.
Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural Networks*, *61*, 85–117.
Sønderby, C. K., Raiko, T., Maaløe, L., Sønderby, S. K., & Winther, O. (2016). Ladder variational autoencoders. In *Annual conference on neural information processing systems (NIPS)*.
Symeonidis, P., & Zioupos, A. (2016). *Matrix and tensor factorization techniques for recommender systems*. Springer.
Vo, T. V., & Soh, H. (2018). Generation meets recommendation: proposing novel items for groups of users. In *ACM conference on recommender systems (RecSy)*.
Wang, C., & Blei, D. M. (2011). Collaborative topic modeling for recommending scientific articles. In *Proceedings of the ACM SIGKDD international conference on knowledge discovery & data mining (KDD)*.
Wang, P., Guo, J., Lan, Y., Xu, J., Wan, S., & Cheng, X. (2015). Learning hierarchical representation model for nextbasket recommendation. In *International conference on research and development in information retrieval (SIGIR)*.
Wang, K., Meng, W., Bian, J., Li, S., & Yang, S. (2018). Spatial context-aware user mention behavior modeling for mentionee recommendation. *Neural Networks*, *106*, 152–167.
Wang, H., Wang, N., & Yeung, D.-Y. (2015). Collaborative deep learning for recommender systems. In *Proceedings of the ACM SIGKDD international conference on knowledge discovery & data mining (KDD)*.
Wang, Q., Yin, H., Hu, Z., Lian, D., Wang, H., & Huang, Z. (2018). Neural memory streaming recommender networks with adversarial training. In *Proceedings of the ACM SIGKDD international conference on knowledge discovery & data mining (KDD)*.
Wang, J., Yu, L., Zhang, W., Gong, Y., Xu, Y., Wang, B., et al. (2017). Irgan: A minimax game for unifying generative and discriminative information retrieval models. In *International conference on research and development in information retrieval (SIGIR)*.

Xiao, T., Liang, S., Shen, H., & Meng, Z. (2018). Neural variational hybrid collaborative filtering. arXiv.org.

Xie, F., Li, S., Chen, L., Xu, Y., & Zheng, Z. (2019). Generative adversarial network based service recommendation in heterogeneous information networks. In *2019 IEEE international conference on web services (ICWS)*.

Yin, H., Wang, Q., Zheng, K., Li, Z., Yang, J., & Zhou, X. (2019). Social influence-based group representation learning for group recommendation. In *International conference on data engineering (ICDE)*.

Zhang, S., Yao, L., Sun, A., & Tay, Y. (2019). Deep learning based recommender system: A survey and new perspectives. *ACM Computing Surveys*, *52*(1), 5.

Zhou, F., Wen, Z., Zhang, K., Trajcevski, G., & Zhong, T. (2019). Variational session-based recommendation using normalizing flows. In *International conference on world wide web (WWW)*.

Zhuang, F., Zhang, Z., Qian, M., Shi, C., Xie, X., & He, Q. (2017). Representation learning via dual-autoencoder for recommendation. *Neural Networks*, *90*, 83–89.