Hierarchical Neural Variational Model for Personalized Sequential Recommendation

Teng Xiao
School of Data and Computer Science, Sun Yat-sen University, China
Guangdong Key Laboratory of Big Data Analysis and Processing, Guangzhou, China
cstengxiao@gmail.com

Shangsong Liang*
School of Data and Computer Science, Sun Yat-sen University, China
Guangdong Key Laboratory of Big Data Analysis and Processing, Guangzhou, China
liangshangsong@gmail.com

Zaiqiao Meng
School of Data and Computer Science, Sun Yat-sen University, China
Guangdong Key Laboratory of Big Data Analysis and Processing, Guangzhou, China
zqmeng@aliyun.com

ABSTRACT

In this paper, we study the problem of recommending personalized items to users given their sequential behaviors. Most sequential recommendation models only capture a user's short-term preference in a short session, and neglect his general (unchanged over time) and long-term preferences. Besides, they are all based on deterministic neural networks, and consider users' latent preferences as point vectors in a low-dimensional continuous space. However, in real world, the evolutions of users' preferences are full of uncertainties. We address this problem by proposing a hierarchical neural variational model (HNVM). HNVM models users' three preferences: general, long-term and short-term preferences through an unified hierarchical deep generative process. HNVM is a hierarchical recurrent neural network that enables it to capture both user's long-term and short-term preferences. Experiments on two public datasets demonstrate that HNVM outperforms state-of-the-art sequential recommendation methods.

CCS CONCEPTS

• **Information systems → Recommender systems**.

KEYWORDS

Sequential Recommendation; Recurrent Networks; Generative Model

ACM Reference Format:
Teng Xiao, Shangsong Liang, and Zaiqiao Meng. 2019. Hierarchical Neural Variational Model for Personalized Sequential Recommendation. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313603

1 INTRODUCTION

Recommender Systems aim at helping users to find interesting items from rich candidates. Most traditional methods [7, 9, 18, 24, 37] are based on matrix completion where the goal is to predict the missing value. In most real scenarios, a user's actions are sequential, which can reflect the user's short-term preference and item's sequential information [25]. However, pervious methods [7, 9, 18, 24] based on matrix completion neglect sequential patterns for recommendation.

To capture users' sequential patterns, a number of methods called session-based recommendation methods [3, 10, 11, 16, 19, 34] built based on Recurrent Neural Networks (RNN) have been proposed. Session-based recommendation aims at predicting the next item that the user will interact with based on on pervious interactions with items inside the current session. Despite the appeal of these methods, they all assume that users are anonymous. However, users' profiles in many application platforms such as YouTube and Linkedin are available. In these platforms, a user's pervious session and profile can be utilized for capturing user's long-term and general preferences [15, 32] so as to personalize recommendation results. As unveiled by Jannach et al. [15, 22], both user's short-term and long-term preferences are crucial to improve recommendation performance. However, recent session-based methods [10, 11, 19] only consider user's most recent actions (short-term preference) in the current session, and neglect user's past session behavior (long-term preference) and users' profiles (general preference).

To consider both user's short-term and long-term preferences, some recent sequential models have been proposed [5, 13, 22, 26, 39]. These methods demonstrate a number of major drawbacks: (a) They do not model user's general (unchanged over time) preference corresponding to a user's profile, e.g., his id and demographics which can be leveraged to improve performance especially in job recommendation websites, e.g., Linkedin, (b) they don't model the relationship between user' past session and current session (with the exception of Quadrana et al [26]) , which is important for capturing users' long-term preferences, and (c) Recent work [26] is based on deterministic hierarchical recurrent neural networks which limit it to capture users' long-term preferences due to the restricted shallow generation process [30].

To address the aforementioned problems, we study the problem of sequential recommendation, which is to enhance personalization recommendation performances. We aim at jointly modeling users' general, long- and short-term preferences though a unified hierarchical deep generative model. Deep generative models [28, 40] have emerged as a powerful model to learn latent disentangled representations from data and improve the diversity of generative results [2, 14, 30, 41], while they have led to promising improvements in computer vision (CV) [14, 40] and neural language processing (NLP) tasks [4, 30, 41]. Specifically, we proposed a **H**ierarchical **N**eural **V**ariational **M**odel (HNVM), which models a hierarchical generative

*Shangsong Liang is the corresponding author.

WWW '19, May 13–17, 2019, San Francisco, CA, USA

ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313603

process of three different preferences: general, long- and short-term preferences. Specifically, HNVM models user's general, long- and short-term preferences to be conditioned on user's profile, the past session and the current session, respectively. Our HNVM extends existing session-based recommendation in three directions: (a) The hierarchical generative process of HNVM makes itself to be able to effectively capture three kinds of preferences for a user, (b) It uses a two-level RNN network to model user's sequential pattern inside session and cross session, and (c) It is a hierarchical latent variables model, it is able to learn users' general and long-term preferences as latent Gaussian representations and can effectively handle the problem of restricted shallow generation process [26]. Our main contributions can be summarized as follows:

(1) We propose a novel hierarchical deep generative model which is the first model to jointly model different categories of preferences: general, long- and short- term preferences in the context of streams of sessions, such that user's profile and his past session information can be effectively captured for recommendation. (2) We propose a hierarchical stochastic gradient variational Bayes (SGVB) inference algorithm to jointly infer latent variables and estimate parameters. We derived a *conditional session − level variational lower bound* that leads to efficient Bayesian learning with back-propagation. (3) The experiment results demonstrate that our HNVM outperforms the state-of-the-art methods for sequential recommendation task.

2 PRELIMINARIES

We first introduce the task of personalized sequential recommendation, and then review the session-based recommendation with recurrent neural network (RNN) that related most to our HNVM.

2.1 Notations and Problem Formulation

Let $\mathcal{U} = \left\{u_1, u_2, ..., u_{|\mathrm{U}|}\right\}$ and $\mathcal{I} = \left\{i_1, i_2, ..., i_{|\mathrm{I}|}\right\}$ be sets of users and items, with $|\mathcal{U}|$ and $|\mathcal{I}|$ being the sizes, respectively. For each user u, let his sequential actions (sessions) sorted by time denote as a sequence of sessions $\mathbf{S}^u = \left\{\mathbf{s}_1^u, \mathbf{s}_2^u, ..., \mathbf{s}_{N_u}^u\right\}$, where N_u is the number of sessions for user u, $\mathbf{s}_n^u = \left\{i_{n,1}^u, i_{n,2}^u, ..., i_{n,|\mathbf{s}_n^u|}^u\right\}$ denotes the item sequence corresponding to the n-th session $\mathbf{s}_n^u$ of user u. Let $\mathbf{a}^u$ denote a user u's profile such as his attributes. The sequential recommendation can be formalized as a sequence prediction task: given a session sequence $\mathbf{S}^u = \left\{\mathbf{s}_1^u, \mathbf{s}_2^u, ..., \mathbf{s}_{N_u}^u\right\}$ and attribute $\mathbf{a}^u$ of user u, generate a sequence of items $\mathbf{s}_{N_u+1}^u$ that user u will interact with at session $N_u + 1$.

2.2 Session-based Recommendation with RNN

RNN has been widely used in session-based recommendation [10, 11, 19, 22, 23, 26, 32] due to it's power for modeling sequence data. However, traditional RNN suffers from vanishing gradient problem. To track this problem, two variant RNNs, namely Gated Recurrent Unit (GRU) [6] and Long Short Term Memory (LSTM) [12] networks have been proposed. Due to the simpler and less parameters, most session-based recommendation models [10, 11, 19, 22, 23, 26, 32] are built on GRU. The GRU takes as input the current item ID in the session and outputs a predicted score for next item in the session.

Given a sequence of actions (a session) $\mathbf{s}_n^u = \left\{i_{n,1}^u, i_{n,2}^u, \ldots, i_{n,|\mathbf{s}_n^u|}^u\right\}$ of user u. The session-based GRU network computes the following current hidden state vector $h_{n,t}^u$ conditional on the last hidden state vector and the current input item ID.

$$\mathbf{h}_{n,t}^u = GRU^{user}(\mathbf{h}_{n,t-1}^u, \mathbf{i}_{n,t}^u), \tag{1}$$

where the GRU^{user} represents the GRU unit [6] called user-level GRU. The $\mathbf{i}_{n,t}^u$ is a one-hot vector of current item ID $i_{n,t}^u$. The output of GRU at time t is the distributions of predicted scores for all items:

$$\hat{\mathbf{y}}_{n,t}^u = g(\mathbf{h}_{n,t}^u), \tag{2}$$

where $g(\cdot)$ is a ranking function such as softmax function. The network can be trained with several ranking loss function like cross-entropy or BPR [27] by stochastic gradient descent.

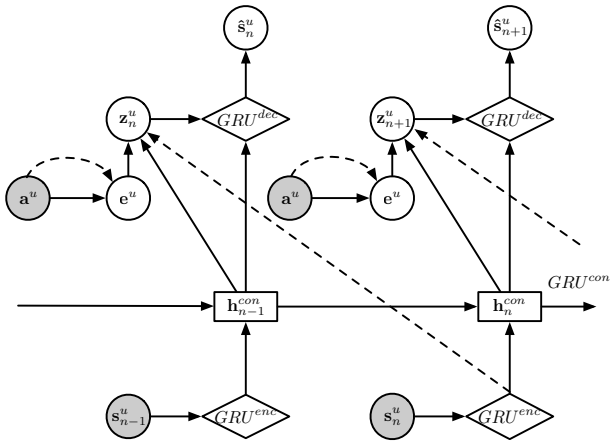

Figure 1: Graphical model of HNVM. Gray circles and white rounded boxes represent observed and latent variables, respectively. Bold solid and dashed lines represent generative and inference processes, respectively. For simplicity, we omit the latent states in encoder GRU and decoder GRU.

3 HIERARCHICAL NEURAL VARIATIONAL MODEL

We proposed a hierarchical neural variational model (HNVM) to model the session generative process for personalized recommendation. We will first introduce the hierarchical generative process, and then detail the hierarchical network in HNVM and the inference algorithm. The graphical model of HNVM is shown in Figure 1.

3.1 Generative Process

Our model HNVM can bee seen as a hierarchical generative model with a series stochastic latent variables. The goal of our model is to maximize the variational evidence lower bound [36] of the conditional log-likelihood of observed variables. For user u, the likelihood of his observed sessions is modeled as follows:

$$p(\mathbf{s}_1^u, \mathbf{s}_2^u, ..., \mathbf{s}_{N_u}^u) = \prod_{n=1}^{N_u} p(\mathbf{s}_n^u | \mathbf{s}_{<n}^u). \tag{3}$$

Due to each session $\mathbf{s}_n^u$ is a sequence of items $(i_{n,1}^u, i_{n,2}^u, ..., i_{n,3}^u)$, Eq. 3 can be further formulated as follows:

$$p(\mathbf{s}_1^u, \mathbf{s}_2^u, ..., \mathbf{s}_{N_u}^u) = \prod_{n=1}^{N_u} \prod_{t=1}^{|\mathbf{s}_n^u|} p(i_{n,t}^u | \mathbf{s}_{<n}^u, i_{n,<t}^u). \tag{4}$$

Our HNVM is a latent variable model which contains a series of latent variables $\mathbf{z}_n^u$ for each session $t = 1, ..., N_u$ in terms of user u. In order to incorporate the previous session information of user u into the next session, we first consider the prior of latent variable $\mathbf{z}_n^u$ (long-term preference) conditioned on all the past observed sessions (the session context) and the general preference $\mathbf{e}^u$:

$$\begin{aligned} p_\theta(\mathbf{z}_n^u | \mathbf{s}_{<n}^u, \mathbf{e}^u) &= \mathcal{N}(\mathbf{z}_n^u | f_{\mu_z}(\mathbf{s}_{<n}^u, \mathbf{e}^u), f_{\sigma_z^2}(\mathbf{s}_{<n}^u, \mathbf{e}^u)) \\ &= \mathcal{N}(\mathbf{z}_n^u | \mu_{\mathbf{z}_n^u}, diag(\sigma_{\mathbf{z}_n^u}^2)), \end{aligned} \tag{5}$$

where $\mathcal{N}(\mu, \Sigma)$ is the multivariate normal distribution with mean $\mu \in \mathbb{R}^{D_z}$ and diagonal covariance matrix $\Sigma \in \mathbb{R}^{D_z \times D_z}$. The conditional prior of the long-term preference $\mathbf{z}_n^u$ is the multivariate Gaussian distribution whose mean and diagonal covariance matrix are parameterized by neural networks $f_{\mu_z}(\cdot)$ and $f_{\delta_z^2}(\cdot)$ with input $\mathbf{s}_{<n}^u$ and $\mathbf{e}^u$. Here, $\mathbf{e}^u$ denotes the latent embedding of user u which represents his general preference and the conditional prior of $\mathbf{e}^u$ is:

$$\begin{aligned} p_\theta(\mathbf{e}^u | \mathbf{a}^u) &= \mathcal{N}(\mathbf{e}^u | f_{\mu_e}(\mathbf{a}^u), diag(f_{\sigma_e^2}(\mathbf{a}^u))) \\ &= \mathcal{N}(\mathbf{e}^u | \mu_{\mathbf{e}^u}, diag(\sigma_{\mathbf{e}^u}^2)), \end{aligned} \tag{6}$$

where the conditional prior of $\mathbf{e}^u$ is also the multivariate Gaussian distribution whose mean and diagonal covariance matrix are parameterized by neural networks $f_{\mu_e}(\cdot)$ and $f_{\delta_e^2}(\cdot)$ with input $\mathbf{a}^u$. Note that we utilize two latent variables $\mathbf{e}^u$ and $\mathbf{z}_n^u$ to represents a user's general and long-term preferences, respectively.

We model the current session $\mathbf{s}_n^u$ to be generative from the long-term preference $\mathbf{z}_n^u$ and the user's context $\mathbf{s}_{<n}^u$:

$$p_\theta(\mathbf{s}_n^u | \mathbf{z}_n^u, \mathbf{s}_{<n}^u) = \prod_{t=1}^{|\mathbf{s}_n^u|} p_\theta(i_{n,t}^u | \mathbf{z}_n^u, \mathbf{s}_{<n}^u, i_{n,<t}^u). \tag{7}$$

According the above generative process of our model, the conditional distribution can be factorized as:

$$\begin{aligned} p_\theta(\mathbf{S}, \mathbf{E}, \mathbf{Z} | \mathbf{A}) = \prod_u \prod_{n=1}^{N_u} \prod_{t=1}^{|\mathbf{s}_n^u|} p_\theta(i_{n,t}^u | \mathbf{s}_{<n}^u, i_{n,<t}^u, \mathbf{z}_n^u) \cdot \\ p_\theta(\mathbf{z}_n^u | \mathbf{s}_{<n}^u, \mathbf{e}^u) p_\theta(\mathbf{e}^u | \mathbf{a}^u), \end{aligned} \tag{8}$$

where $\mathbf{S}$ and $\mathbf{A}$ denote the observed sessions and the attributes of all users, respectively. $\mathbf{E}$ and $\mathbf{Z}$ denote the sets of latent variables of $\mathbf{e}^u$ and $\mathbf{z}_n^u$ of all users, respectively. We use θ to denote the all parameters in the generative model. In what follows, we will detail the hierarchical generative network architecture of HNVM.

3.2 The Hierarchical Network Architecture

As shown in Figure 1, our model is hierarchical, which contains three RNN networks: the *encoder* RNN, the *context* RNN [29] and the *decoder* RNN. The *encoder* RNN encodes session $\mathbf{s}_n^u$ into a fixed-size vector:

$$\mathbf{h}_{n,t}^u = GRU^{enc}(\mathbf{h}_{n,t-1}^u, \mathbf{i}_{n,t}^u), \ t = 1, 2, .., |\mathbf{s}_n^u|, \tag{9}$$

where $\mathbf{h}_{n,t}^u$ is the hidden state vector at time t, $\mathbf{i}_{n,t}^u$ is the one-hot vector of item $i_{n,t}^u$, and GRU^{enc} represents the non-linear activation GRU function [6]. The last hidden vector $\mathbf{h}_{n,|\mathbf{s}_n^u|}^u$ can be viewed as the summary of current session $\mathbf{s}_n^u$.

For the *context* RNN, it encodes the session sequence as:

$$\mathbf{h}_n^{con} = GRU^{con}(\mathbf{h}_{n-1}^{con}, \mathbf{h}_{n,|\mathbf{s}_n^u|}^u), \ n = 1, 2, .., |\mathbf{S}^u|, \tag{10}$$

where the GRU^{con} denotes the non-linear function parameterized by *context* RNN and $\mathbf{h}_n^{con}$ is the hidden state of *context* RNN. $\mathbf{h}_n^{con}$ can be viewed as the context-level representation which is the summary of all the observed previous sessions ($\mathbf{s}_{\leq n}^u$).

For the *decoder* RNN, it takes $\mathbf{h}_{n-1}^{con}$ and $\mathbf{z}_n^u$ as input to generate the next session $\mathbf{s}_n^u$:

$$\begin{aligned} \mathbf{h}_{n,0}^{dec} = \mathbf{0}, \ \mathbf{h}_{n,t}^{dec} = GRU^{dec}(\mathbf{h}_{n,t-1}^{dec}, \mathbf{i}_{n,t}^u, \mathbf{h}_{n-1}^{con}, \mathbf{z}_n^u), \\ t = 1, 2, .., |\mathbf{s}_n^u|, \end{aligned} \tag{11}$$

and the generative items are sampled according to Eq. 2. In our HNVM, the *decoder* RNN captures sequential pattern inside the current session (i.e., short-term preference).

As shown in Figure 1, the generative process (§ 3.1) of a session sequence for user u through the hierarchical network are:

(1) The *encoder* RNN encodes the $(n-1)$-th session ($\mathbf{s}_{n-1}^u$) of user u into a fix-size vector $\mathbf{h}_{n-1,|\mathbf{s}_n^u|}^{enc}$, which is taken by the *context* RNN as input to compute the context hidden state $\mathbf{h}_{n-1}^{con}$ (Eq. 10).

(2) Sample the user latent embedding $\mathbf{e}^u$ from conditional prior distribution (Eq. 6).

(3) A two-layer Multi Layer Perceptron (MLP) takes as input the concatenation of both $\mathbf{h}_{n-1}^{con}$ and the general preference $\mathbf{e}^u$. The network output is transformed to give the mean and diagonal covariance matrix of $\mathbf{z}_n^u$ in (Eq. 5) as:

$$\begin{bmatrix} \mu \\ \log(\sigma_{\mathbf{z}_n^u}) \end{bmatrix} = \text{MLP}(\mathbf{h}_{n-1}^{con}, \mathbf{e}^u). \tag{12}$$

(4) Draw latent variable $\mathbf{z}_n^u$ from the prior $p_\theta(\mathbf{z}_n^u | \mathbf{s}_{<n}^u, \mathbf{e}^u)$ whose mean and covariance matrix provided by step (3).

(5) The *decoder* RNN takes context vector $\mathbf{h}_{n-1}^{con}$ and $\mathbf{z}_n^u$ as input to generative next session $\mathbf{s}_n^u$ of user u (Eq. 7).

3.3 Inference Process

As shown in graph model Figure 1, the goal of HNVM is to infer the latent variables $\mathbf{e}^u$ and $\mathbf{z}_n^u$ for every users and all their sessions, and estimate the network parameter θ. Due to the latent variables are hierarchical and the generative process is parameterized by the neural network, it is intractable to infer these latent variables and estimate network parameters by using traditional mean-filed approximation [36]. Inspired by Deep Latent Gaussian models (DLGm) [28, 31], we propose a hierarchical conditional stochastic gradient variational Bayes (SGVB) algorithm to infer our HNVM. For our model, we consider the following type of variational distribution factorization structure:

$$q_\phi(\mathbf{E}, \mathbf{Z} | \mathbf{S}, \mathbf{A}) = \prod_u q_\phi(\mathbf{e}^u | \mathbf{a}^u) \prod_{n=1}^{N_u} q_\phi(\mathbf{z}_n^u | \mathbf{s}_{<n}^u, \mathbf{s}_n^u), \tag{13}$$

$$q_\phi(\mathbf{e}^u | \mathbf{a}^u) = \mathcal{N}(\mathbf{u} | g_{\mu_e}(\mathbf{a}^u), diag(g_{\sigma_e^2}(\mathbf{a}^u))) \tag{14}$$

$$= \mathcal{N}(\mathbf{e}^u | \tilde{\mu}_{\mathbf{e}^u}, diag(\tilde{\sigma}_{\mathbf{e}^u}^2)), \tag{15}$$

$$q_\phi(\mathbf{z}_n^u | \mathbf{s}_{<n}^u, \mathbf{s}_n^u) = \mathcal{N}(\mathbf{z}_n^u | g_{\mu_z}(\mathbf{s}_{<n}^u, \mathbf{s}_n^u), g_{\sigma_z^2}(\mathbf{s}_{<n}^u, \mathbf{s}_n^u)) \tag{16}$$

$$= \mathcal{N}(\mathbf{e}^u | \tilde{\mu}_{\mathbf{z}_n^u}, diag(\tilde{\sigma}_{\mathbf{z}_n^u}^2)). \tag{17}$$

The posteriors over $\mathbf{e}^u$ and $\mathbf{z}_n^u$ are all multivariate diagonal Gaussian distributions. Similar to the notations in generative model, we let $g_{\mu_e}(\cdot)$, $g_{\sigma_e^2}(\cdot)$, $g_{\mu_z}(\cdot)$ and $g_{\sigma_z^2}(\cdot)$ be neural networks whose parameters are denoted by ϕ. This factorization assumes the general preference $\mathbf{e}^u$ is only conditioned on u's attribute $\mathbf{a}^u$, whereas the long-term preference $\mathbf{z}_n^u$ is conditioned on $\mathbf{s}_{<n}^u$ which is the *context* RNN hidden state $\mathbf{h}_{n-1}^{con}$, and current session summary $\mathbf{h}_{n,|\mathbf{s}_n^u|}^u$. To infer latent variables and estimate network parameters, we maximize the hierarchical variational **E**vidence **L**ower **BO**und (ELBO) of the conditional log likelihood (i.e., Eq. 8):

$$\log p_\theta(\mathbf{S}|\mathbf{A}) \geq \sum_u \mathcal{L}(\theta,\phi;\mathbf{S}^u,\mathbf{a}^u) = \sum_u \sum_{n=1}^{N_u} \mathbb{E}_{q_\phi(\mathbf{z}_n^u|\mathbf{s}_{<n}^u,\mathbf{s}_n^u)}\Big[\log p_\theta(\mathbf{s}_n^u|\mathbf{z}_n^u,\mathbf{s}_{<n}^u)\Big] - \mathbb{E}_{q_\phi(\mathbf{e}^u|\mathbf{a}^u)}\Big[\mathrm{KL}(q_\phi(\mathbf{z}_n^u|\mathbf{s}_{<n}^u,\mathbf{s}_n^u)||p_\theta(\mathbf{z}_n^u|\mathbf{s}_{<n}^u,\mathbf{e}^u)\Big] - \mathrm{KL}(q_\phi(\mathbf{e}^u|\mathbf{a}^u)||p_\theta(\mathbf{e}^u|\mathbf{a}^u)), \tag{18}$$

where $\mathrm{KL}(q||p)$ is the Kullback-Leibler (KL) divergence between distributions p and q. The derivation of the ELBO in Eq. 18 is included in Appendix A. Maximizing the ELBO (Eq. 18) is equivalent to maximize the log-likelihood of observed variables and to make the variational distributions $q_\phi(\mathbf{e}^u|\mathbf{a}^u)$ and $q_\phi(\mathbf{z}_n^u|\mathbf{s}_{<n}^u,\mathbf{s}_n^u)$ of each user to be as close as possible to their intractable true posteriors $p_\theta(\mathbf{e}^u|\mathbf{S}^u,\mathbf{a}^u)$ and $p_\theta(\mathbf{z}_n^u|\mathbf{S}^u,\mathbf{a}^u)$, respectively. The last KL term in Eq. 18 has analytical form. However, for the two expectation terms, we can not compute them analytically. To solve this problem, we approximate the two terms using Monte Carlo sampling, and use suitable reparameterization trick [17, 28] with low variance.

Since the variational distribution of $\mathbf{e}^u$ does not depend on the entire session sequence $\mathbf{S}^u$, the *conditional sequence − level variational lower bound* $\mathcal{L}(\theta,\phi;\mathbf{S}^u,\mathbf{a}^u)$ can be decomposed into the sum of $\mathcal{L}(\theta,\phi;\mathbf{s}_n^u|\mathbf{a}^u)$, the *conditional session − level variational lower bound*, and the function $\mathcal{F}_{\theta,\phi}(\mathbf{a}^u)$ w.r.t θ and ϕ (see Eq. 22 in Appendix A):

$$\mathcal{L}(\theta,\phi;\mathbf{S}^u,\mathbf{a}^u) = \sum_{n=1}^{N_u} \mathcal{L}(\theta,\phi;\mathbf{s}_n^u|\mathbf{a}^u) + \mathcal{F}_{\theta,\phi}(\mathbf{a}^u). \tag{19}$$

Instead of sampling a batch at the sequence-level to maximize the ELBO, we can sample a batch at the session-level to maximize it. This method makes our model scalable when the session sequence $\mathbf{S}^u$ is extremely long, such that computing an entire sequence for a batched update is very computationally expensive.

4 EXPERIMENTAL SETUP

4.1 Research Questions

The main research questions guiding the paper are: (**RQ1**) Does HNVM outperform state-of-the-art sequential recommendation methods? (**RQ2**) How does the captured the long-term and general preferences impact recommendation accuracy? (**RQ3**) What is the impact of the number of users' past sessions for recommendation performance? (**RQ4**) Is HNVM effective on the other task like next-session recommendation?

4.2 Datasets and Data Preprocessing

In order to answer our research questions, we work with two publicly available datasets: RecSys Challenge 2015[1] (RSC) [10] and IJCAI-15 Competition datasets[2] (IJC) [26]. RSC dataset contains interactions on job posting for 770K users over a 80-days' period. IJC contains users' shopping log over six months' period in online shopping site Tmall. For RSC dataset, we first partitioned the interaction data into sessions by considering 30-minutes' idle threshold and removed interactions having type 'delete'. We hold out the last interacted item in the last session of each user as the test data, and the item before the last item as the validation set. The rest items are treated as training dataset. For the IJC dataset, we follow the the preprocessing procedure as same as that in [39]. For simplicity, we just use user's id number's one-hot vector as static profile representation $\mathbf{a}^u$ (Note that we don't use other attributes such age or gender, because there is no such information in IJC dataset). Table 1 shows the statistics of the two datasets after preprocessing.

Table 1: Statistics of datasets.

Datasets	RSC	IJC
Users	11,479	20,648
Items	59,297	25,129
Sessions	89,591	75,426
Training sessions	78,276	71,892
Test sessions	11,315	3,635
Interactions per session	6.1	2.73

4.3 Baselines and Settings

We make comparisons between the proposed HNVM model and the following state-of-the-art algorithms for sequential recommendation: (1) **POP**: This method always recommends the most-popular (the largest number of interactions in the training set) items. (2) **Item-KNN**: This method recommends items based on the cosine similarity between items. The similarity is defined by the co-occurrence number between items in sessions. (3) **BPR**: This method [27] optimizes a pair-wise ranking loss for general recommendation. (4) **NCF**: This method [9] is the state-of-the-art general recommendation method. (5) **GRU4Rec**: This model [10] utilizes the basic GRU [6] unit to the task of session-based recommendation. (6) **HRNN**: This model [26] utilizes past session information through a hierarchical RNN. We use it's HRNN-*All* version in our experiments. (7) **Caser**: This is the state-of-the-art sequential recommendation model [35] which uses convolutional filters to learn sequential patterns. It also captures user's general preference. (8) **STAMP**: This model [22] utilizes a short-term attention/memory mechanism to capture user's long- and short-term preferences.

For fair comparisons, all GRU based baselines (GRU4Rec, HRNN) including our HNVM use the BPR loss function. For all baselines, we use a grid search to find the optimal parameters using the validation set. For HNVM, we randomly initialize parameters using uniform distribution with variance 0.01 and bases equal 0. We use one-layer GRU for encoder, context and decoder RNN. The hyper-parameters are found via extensive grid search. We use AdaGrad [8] with momentum to optimize our model. The batch size is set to 256. The

[1] http://2016.recsyschallenge.com/

[2] https://ijcai-15.org/index.php/repeat-buyers-prediction-competition

hidden states of GRU are all set to 200. Since the high variance leads the gradients not stable, we multiple the diagonal covariance matrices of the prior and posterior distributions by 0.05. We use a dropout regularization [33] on the hidden states of GRU with a fixed dropout rate of 0.1. Similar to [1], we multiply the KL terms in Eq. 18 by scalar, which starts from 0 and linearly increases to 1.

4.4 Evaluation Metrics

Similar to that in [13, 26], we perform our model and the baselines on the next-item sequential recommendation task for comparisons. We use the widely used evaluation metrics: Recall@k (Recall at k), MRR@k (Mean Reciprocal Rank at k) [21] and Pre@k (Precision at k) [20, 38]. We compute the scores at depth 5, i.e., let k =5 for evaluation.

5 RESULTS AND ANALYSIS

5.1 Overall Performance (RQ1)

To start, for question **RQ1**, to evaluate the effectiveness of HNVM in personalized sequential recommendation, we examine the performance of HNVM on the next-item recommendation task. Table 2 shows the overall performance of all methods in terms of Recall@5, MRR@5 and Pre@5. From Table 2, we have the following conclusions: (1) Methods capturing users' short-term preferences like GRU4Rec, HRNN, Caser, STAMP and HNVM work better than general methods, i.e., BRP and NCF, which illustrates users' short-term preferences need to be captured for next-item recommendation task. (2) On the RSC dataset, our model HNVM and Caser outperform other sequential recommendation models, i.e., GRU4Rec and STAMP. The reason is that the user's static profile is more important in job search platform (RSC) than shopping or video platforms (IJC) and both our model HNVM and Caser can capture users' general preferences corresponding to users' profile. (3) Although our HNVM and Caser both can capture users' general and short-term preferences, our HNVM works better than Caser. The reason is that user has different purchase purpose in different sessions and Caser doesn't model this cross-session relationship (i.e., long-term preferences). (4) Our HNVM significantly outperforms HRNN, which demonstrates that the latent variable characteristic makes our model to be able to more effectively capture the users' long-term preferences. All of these findings demonstrate that HNVM can effectively capture users' different preferences and it is able to maintain significant improvements of sequential recommendation performance over the state-of-the-arts baseline methods.

5.2 Impact of the Long-term and General Preferences (RQ2)

Next, we turn to **RQ2**. For the purpose of analyzing the impact of user's long-term and general preferences on the next-item recommendation task, we spilt every sessions by the order of interaction within the sessions. We only consider sessions which have ≥ 5 interactions on RSC datasets (6736 sessions). We group interactions in these sessions as three groups: *Begin*, *Middle* and *End*. The *Begin* group contains the first 2 interactions in these sessions, the *Middle* contains 3rd -4th and the *End* contains the rest of interactions. We evaluate recommendation performance of our HNVM on these three groups, respectively. Since sessions' lengths in the IJC dataset are general very short, we only conduct this analysis on the RSC dataset. Figure 2 shows the Recall@5 and MRR@5 results with different positions of interactions on RSC datasets (the performance pattern in terms of Pre@5 is similar to that in terms of Recall@5, and thus we omit it here due to space limitations). From the results, some interesting findings can be observed: (1) For all RNN-based methods, i.e., HNVM, HRNN and GRU4Rec, the performance increases with the number of pervious items in the session, which demonstrates that these methods can both effectively capture users' short-term preference within the current session. (2) Personalized recommendation models, HNVM, HRNN and Caser significantly outperform vanilla RNN-based methods like GRU4Rec and STAMP by a large margin, and the margin is larger on the *Begin* group than that on the *Middle* group. The reason is that GRU4Rec and STAMP do not incorporate any user's past session information (i.e., long-term preferences) when user starts a new session and these models can hardly capture the user's short-term preference when she/he just clicks a few items. (3) Our HNVM significantly and consistently outperforms the best two baselines, HRNN and Caser, especially in the *Begin* and *End* groups, which illustrates that capturing both users' long-term (past session information) and general (profile information) preferences can significantly contribute to predict the first several actions and the last several actions in long sessions.

Table 2: Recommendation performance comparison on the RSC and IJC datasets in terms of Recall@5, MRR@5 and Pre@5. "Improv." denotes the improvement of HNVM relative to the best baseline.

	RSC			IJC		
	Recall	MRR	Pre	Recall	MRR	Pre
POP	.0253	.0174	.0084	.0049	.0025	.0009
Item-KNN	.0697	.0406	.0139	.0126	.0141	.0018
BPR	.0714	.0497	.0159	.0154	.0158	.0067
NCF	.0847	.0586	.0184	.0174	.0166	.0098
GRU4Rec	.1292	.0799	.0258	.0180	.0159	.0141
HRNN	.1356	.0832	.0284	.0325	.0265	.0157
Caser	.1422	.0921	.0301	.0344	.0254	.0182
STAMP	.1337	.0802	.0267	.0321	.0234	.0171
HNVM	**.1521**	**.0991**	**.0357**	**.0371**	**.0278**	**.0197**
Improv.	+6.9%	+7.6%	+18.6%	+7.8%	+4.9%	+8.2%

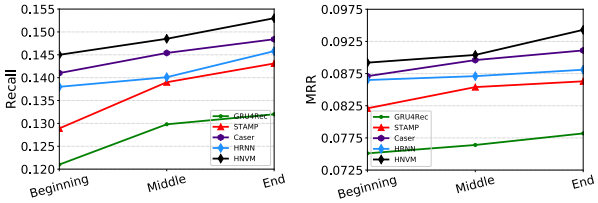

Figure 2: Performance comparison in terms of Recall@5 and MRR@5 on RSC dataset with different positions in sessions.

5.3 The Number of Past Sessions (RQ3)

We now examine the impact of the number of user's past sessions for performance. We partitioned users on RSC dataset into two groups based on the number of their past sessions: *Short* and *Long* groups. The *Short* group refers the number of user's sessions ≤ 6

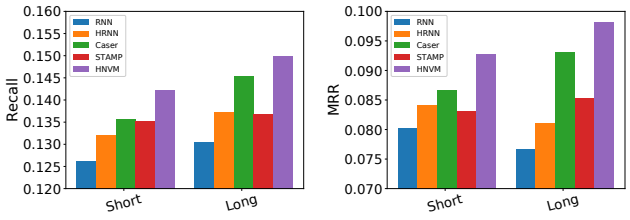

Figure 3: Performance comparison in terms of Recall@5 and MRR@5 on RSC datasets with different past session length.

Table 3: Next-session recommendation performance comparison in terms of Recall@5, MRR@5 and Pre@5.

	RSC			IJC		
	Recall	MRR	Pre	Recall	MRR	Pre
GRU4Rec	.0732	.0387	.0098	.0122	.0101	.0095
HRNN	.1054	.0607	.0149	.0213	.0198	.0157
Caser	.0987	.0531	.0128	.0187	.0177	.0139
STAMP	.0872	.0412	.0114	.0162	.0148	.0118
HNVM	**.1134**	**.0696**	**.0154**	**.0221**	**.0209**	**.0164**
Improv.	+7.6%	+14.7%	+3.4%	+3.8%	+5.6%	+4.5%

(67%), while the *Long* refers to > 6 (33%). We evaluate recommendation performance based on the two groups, respectively. As showed in Figure 3, we can observe that the performance of HRNN, Caser and HNVM on the *Long* group is better than that on the *Short* group. The reason is that HRNN, Caser and HNVM not only capture users' short-term preferences in the current session but also consider users' general (profile) or long-term (past session information) preferences , and the more the user past sessions are observed, the better the model learns the users' long-term preferences. Figure 3 shows that our model significantly outperforms the best baselines (i.e., HRNN and Caser) on next-item recommendation task.

5.4 Performance Comparison on the Next-session Recommendation Task (RQ4)

To further evaluate our model, similar to [13], we consider another sequential recommendation task: next-session recommendation. We use user's last session as test set, the session before the last as validation set and the rest of sessions as training set. Table 3 shows the next-session performance between HNVM with the best baselines. From Table 3, we can observe our HNVM and HRNN significantly outperform basic session-based methods, i.e., STAMP and GRU4Rec. The reason is that both our HNVM and HRNN effectively utilize users' past sessions and capture users' long-term preferences and these long-term preferences help models mining users' different purpose at different sessions. The finding that our HNVM outperforms HRNN indicts that the users' latent long-term preferences learnt by HNVM work better than those of HRNN.

6 CONCLUSION

In this paper, we have studied the problem of personalized sequential recommendation. To track this problem, we have proposed a hierarchical neural variational model, abbreviated as HNVM, that is the first attempt to capture users' different level preferences, i.e., general, long- and short-term preferences through a unified hierarchical generative process. HNVM can effectively model user's sequential patterns inside sessions and across sessions through it's hierarchical network structure. Our HNVM learns users' different preferences via a hierarchical Gaussian latent representations, which make it be able to capture better preferences than other deterministic model. To effectively infer our model, we have proposed hierarchical stochastic gradient variational Bayes inference algorithm. We also have derived a *conditional session − level variational lower bound*, which makes our HNVM be able to be effectively learned via sampling mini-batches at session-level. Experimental results on two publicly available datasets demonstrate the effectiveness of the proposed algorithms.

A DERIVATION OF CONDITIONAL SESSION-LEVEL VARIATIONAL LOWER BOUND

All notations below are defined in the body of the paper. The conditional variational ELBO for user u can be derived as follows:

$$\log p_\theta(\mathbf{S}^u|\mathbf{a}^u) \geq \mathcal{L}(\theta,\phi;\mathbf{S}^u,\mathbf{a}^u) =$$

$$\mathbb{E}_{q_\phi(\mathbf{e}^u,\mathbf{z}_n^u|\mathbf{S}^u)} \log \frac{p_\theta(\mathbf{e}^u|\mathbf{a}^u)\prod_{n=1}^{N_u} p_\theta(\mathbf{z}_n^u|\mathbf{s}_{<n}^u,\mathbf{e}^u)p_\theta(\mathbf{s}_n^u|\mathbf{z}_n^u,\mathbf{s}_{<n}^u)}{q_\phi(\mathbf{e}^u|\mathbf{a}^u)\prod_{n=1}^{N_u} q_\phi(\mathbf{z}_n^u|\mathbf{s}_{<n}^u,\mathbf{s}_n^u)}$$

$$= \sum_{n=1}^{N_u} \mathbb{E}_{q_\phi(\mathbf{z}_n^u|\mathbf{s}_{<n}^u,\mathbf{s}_n^u)}\Big[\log p_\theta(\mathbf{s}_n^u|\mathbf{z}_n^u,\mathbf{s}_{<n}^u)\Big]$$

$$- \sum_{n=1}^{N_u} \mathbb{E}_{q_\phi(\mathbf{e}^u|\mathbf{a}^u)}\Big[\mathrm{KL}(q_\phi(\mathbf{z}_n^u|\mathbf{s}_{<n}^u,\mathbf{s}_n^u)||p_\theta(\mathbf{z}_n^u|\mathbf{s}_{<n}^u,\mathbf{e}^u)\Big] \quad (20)$$

$$- \mathrm{KL}(q_\phi(\mathbf{e}^u|\mathbf{a}^u)||p_\theta(\mathbf{e}^u|\mathbf{a}^u)), \quad (21)$$

where the expected KL terms in Eq. 20 are two Gaussian distributions, and the KL divergence $\mathrm{KL}(q_\phi(\mathbf{e}^u|\mathbf{a}^u)||p_\theta(\mathbf{e}^u|\mathbf{a}^u))$ in Eq. 21 can be computed analytically. Thus the hierarchical ELBO for a user u's sessions' sequence $\mathbf{S}^u$ as follows:

$$\mathcal{L}(\theta,\phi;\mathbf{S}^u,\mathbf{a}^u) = \sum_{n=1}^{N_u} \mathbb{E}_{q_\phi(\mathbf{z}_n^u|\mathbf{s}_{<n}^u,\mathbf{s}_n^u)}\Big[\log p_\theta(\mathbf{s}_n^u|\mathbf{z}_n^u,\mathbf{s}_{<n}^u)\Big]$$

$$+ \mathbb{E}_{q_\phi(\mathbf{e}^u|\mathbf{a}^u)}\Big[\frac{1}{2}\sum_{d=1}^{D_z} 1 + \log\frac{\tilde{\sigma}^2_{\mathbf{z}_n^u,d}}{\sigma^2_{\mathbf{z}_n^u,d}} - \frac{(\tilde{\mu}_{\mathbf{z}_n^u,d}-\mu_{\mathbf{z}_n^u,d})^2+\tilde{\sigma}^2_{\mathbf{z}_n^u,d}}{\sigma^2_{\mathbf{z}_n^u,d}}\Big]$$

$$+ \frac{1}{2}\sum_{d=1}^{D_e}(1 + \log\frac{\tilde{\sigma}^2_{\mathbf{e}^u,d}}{\sigma^2_{\mathbf{e}^u,d}} - \frac{(\tilde{\mu}_{\mathbf{e}^u,d}-\mu_{\mathbf{e}^u,d})^2+\tilde{\sigma}^2_{\mathbf{e}^u,d}}{\sigma^2_{\mathbf{e}^u,d}})$$

$$= \sum_{n=1}^{N_u} \mathcal{L}(\theta,\phi;\mathbf{s}_n^u|\mathbf{a}^u) + \mathcal{F}_{\theta,\phi}(\mathbf{a}^u), \quad (22)$$

where the two expectation terms can not be solved analytically. Therefore, we apply Monte Carlo sampling and the reparametrization trick [17] to approximate the two terms. The samples are:

$$\mathbf{e}^u = \tilde{\mu}_{\mathbf{e}^u} + \tilde{\sigma}_{\mathbf{e}^u} \odot \epsilon_{\mathbf{e}^u}, \quad \epsilon_{\mathbf{e}^u} \sim \mathcal{N}(\mathbf{0},\mathbf{I}_{D_e}), \quad (23)$$

$$\mathbf{z}_n^u = \tilde{\mu}_{\mathbf{z}_n^u} + \tilde{\sigma}_{\mathbf{z}_n^u} \odot \epsilon_{\mathbf{z}_n^u}, \quad \epsilon_{\mathbf{z}_n^u} \sim \mathcal{N}(\mathbf{0},\mathbf{I}_{D_z}), \quad (24)$$

where $\odot$ denotes an element-wise product and $\mathcal{N}(\mathbf{0},\mathbf{I})$ is the normal Gaussian distribution with mean and covariance being $\mathbf{0}$ and $\mathbf{I}$, respectively.

REFERENCES

[1] Samuel R Bowman, Luke Vilnis, Oriol Vinyals, Andrew Dai, Rafal Jozefowicz, and Samy Bengio. 2016. Generating Sentences from a Continuous Space. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*. 10–21.

[2] Kris Cao and Stephen Clark. 2017. Latent Variable Dialogue Models and their Diversity. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, Vol. 2. 182–187.

[3] Sotirios P Chatzis, Panayiotis Christodoulou, and Andreas S Andreou. 2017. Recurrent Latent Variable Networks for Session-Based Recommendation. In *Proceedings of the 2nd Workshop on Deep Learning for Recommender Systems*. ACM, 38–45.

[4] Hongshen Chen, Zhaochun Ren, Jiliang Tang, Yihong Eric Zhao, and Dawei Yin. 2018. Hierarchical Variational Memory Network for Dialogue Generation. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 1653–1662.

[5] Xu Chen, Hongteng Xu, Yongfeng Zhang, Jiaxi Tang, Yixin Cao, Zheng Qin, and Hongyuan Zha. 2018. Sequential recommendation with user memory networks. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. ACM, 108–116.

[6] Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259* (2014).

[7] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems*. ACM, 191–198.

[8] John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research* 12, Jul (2011), 2121–2159.

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

[10] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2016. Session-based recommendations with recurrent neural networks. *ICLR* (2016).

[11] Balázs Hidasi, Massimo Quadrana, Alexandros Karatzoglou, and Domonkos Tikk. 2016. Parallel recurrent neural network architectures for feature-rich session-based recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems*. ACM, 241–248.

[12] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[13] Jin Huang, Wayne Xin Zhao, Hongjian Dou, Ji-Rong Wen, and Edward Y Chang. 2018. Improving Sequential Recommendation with Knowledge-Enhanced Memory Networks. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. ACM, 505–514.

[14] Unnat Jain, Ziyu Zhang, and Alexander G Schwing. 2017. Creativity: Generating Diverse Questions using Variational Autoencoders.. In *CVPR*. 5415–5424.

[15] Dietmar Jannach, Lukas Lerche, and Michael Jugovac. 2015. Adaptation and evaluation of recommendations for short-term shopping goals. In *Proceedings of the 9th ACM Conference on Recommender Systems*. ACM, 211–218.

[16] Dietmar Jannach and Malte Ludewig. 2017. When recurrent neural networks meet the neighborhood for session-based recommendation. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*. ACM, 306–310.

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

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

[19] Jing Li, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tao Lian, and Jun Ma. 2017. Neural Attentive Session-based Recommendation. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, CIKM 2017, Singapore, November 06 - 10, 2017*. 1419–1428.

[20] Shangsong Liang, Ilya Markov, Zhaochun Ren, and Maarten de Rijke. 2018. Manifold learning for rank aggregation. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*. 1735–1744.

[21] Shangsong Liang, Xiangliang Zhang, Zhaochun Ren, and Evangelos Kanoulas. 2018. Dynamic embeddings for user profiling in twitter. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 1764–1773.

[22] Qiao Liu, Yifu Zeng, Refuoe Mokhosi, and Haibin Zhang. 2018. STAMP: short-term attention/memory priority model for session-based recommendation. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 1831–1839.

[23] Pablo Loyola, Chen Liu, and Yu Hirate. 2017. Modeling User Session and Intent with an Attention-based Encoder-Decoder Architecture. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*. ACM, 147–151.

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

[25] Massimo Quadrana, Paolo Cremonesi, and Dietmar Jannach. 2018. Sequence-Aware Recommender Systems. *ACM Comput. Surv.* 51, 4 (July 2018).

[26] Massimo Quadrana, Alexandros Karatzoglou, Balázs Hidasi, and Paolo Cremonesi. 2017. Personalizing session-based recommendations with hierarchical recurrent neural networks. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*. ACM, 130–137.

[27] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*. AUAI Press, 452–461.

[28] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. 2014. Stochastic Backpropagation and Approximate Inference in Deep Generative Models. In *International Conference on Machine Learning*. 1278–1286.

[29] Iulian Vlad Serban, Alessandro Sordoni, Yoshua Bengio, Aaron C Courville, and Joelle Pineau. 2016. Building End-To-End Dialogue Systems Using Generative Hierarchical Neural Network Models.. In *AAAI*, Vol. 16. 3776–3784.

[30] Iulian Vlad Serban, Alessandro Sordoni, Ryan Lowe, Laurent Charlin, Joelle Pineau, Aaron Courville, and Yoshua Bengio. 2017. A Hierarchical Latent Variable Encoder-Decoder Model for Generating Dialogues. In *Thirty-First AAAI Conference on Artificial Intelligence*.

[31] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. 2015. Learning structured output representation using deep conditional generative models. In *Advances in Neural Information Processing Systems*. 3483–3491.

[32] Gabriele Sottocornola, Panagiotis Symeonidis, and Markus Zanker. 2018. Session-based News Recommendations. In *Companion of the The Web Conference 2018 on The Web Conference 2018*. International World Wide Web Conferences Steering Committee, 1395–1399.

[33] 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.

[34] Yong Kiam Tan, Xinxing Xu, and Yong Liu. 2016. Improved recurrent neural networks for session-based recommendations. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*. ACM, 17–22.

[35] Jiaxi Tang and Ke Wang. 2018. Personalized top-n sequential recommendation via convolutional sequence embedding. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. ACM, 565–573.

[36] Martin J Wainwright, Michael I Jordan, and others. 2008. Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning* 1, 1–2 (2008), 1–305.

[37] Teng Xiao, Shangsong Liang, Weizhou Shen, and Zaiqiao Meng. 2019. Bayesian Deep Collaborative Matrix Factorization. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence (AAAI 2019)*. AAAI.

[38] Christopher C Yang. 2010. Search engines information retrieval in practice. *Journal of the American Society for Information Science and Technology* 61, 2 (2010), 430–430.

[39] Haochao Ying, Fuzhen Zhuang, Fuzheng Zhang, Yanchi Liu, Guandong Xu, Xing Xie, Hui Xiong, and Jian Wu. 2018. Sequential Recommender System based on Hierarchical Attention Networks. In *the 27th International Joint Conference on Artificial Intelligence*.

[40] Li Yingzhen and Stephan Mandt. 2018. Disentangled sequential autoencoder. In *International Conference on Machine Learning*. 5656–5665.

[41] Tiancheng Zhao, Ran Zhao, and Maxine Eskenazi. 2017. Learning Discourse-level Diversity for Neural Dialog Models using Conditional Variational Autoencoders. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vol. 1. 654–664.