7-2021

Variational learning from implicit bandit feedback

Quoc Tuan TRUONG
Singapore Management University, qttruong.2017@phdcs.smu.edu.sg

Hady W. LAUW
Singapore Management University, hadywlauw@smu.edu.sg

Noname manuscript No.
(will be inserted by the editor)

Variational Learning from Implicit Bandit Feedback

Quoc-Tuan Truong · Hady W. Lauw

Received: date / Accepted: date

Abstract Recommendations are prevalent in Web applications (e.g., search ranking, item recommendation, advertisement placement). Learning from bandit feedback is challenging due to the sparsity of feedback limited to system-provided actions. In this work, we focus on batch learning from logs of recommender systems involving both bandit and organic feedbacks. We develop a probabilistic framework with a likelihood function for estimating not only explicit positive observations but also implicit negative observations inferred from the data. Moreover, we introduce a latent variable model for organic-bandit feedbacks to robustly capture user preference distributions. Next, we analyze the behavior of the new likelihood under two scenarios, i.e., with and without counterfactual re-weighting. For speedier item ranking, we further investigate the possibility of using Maximum-a-Posteriori (MAP) estimate instead of Monte Carlo (MC)-based approximation for prediction. Experiments on both real datasets as well as data from a simulation environment show substantial performance improvements over comparable baselines.

Keywords Variational learning · Bandit feedback · Recommender systems · Computational advertising

1 Introduction

Recommender systems rely primarily on user-item interactions as feedback in model learning. We are interested in learning from *bandit feedback* (Jeunen et al., 2019), where users register feedback only for items recommended by the system. For instance, in computational advertising (ad) (Rohde et al., 2018), a user could respond only to the ad being shown, but not to other ads not shown. Contrast this to *organic feedback*, assumed to have arisen naturally from user-driven interactions

Quoc-Tuan Truong
Singapore Management University, School of Computing and Information Systems
E-mail: qttruong.2017@smu.edu.sg

Hady W. Lauw (corresponding author)
Singapore Management University, School of Computing and Information Systems
E-mail: hadywlauw@smu.edu.sg

with the system. In rating prediction (Koren et al., 2009), users presumably select items to rate. In Web browsing (Hidasi et al., 2016), we learn from which browsed products end up in a purchase. These forms of users' *organic feedback* and *bandit feedback* are effectively collected by the websites for behavioral advertising [1].

The traditional approach to dealing with bandit feedback, which is not the focus of this work, is a class of reinforcement learning techniques (Sutton et al., 1998) known as multi-armed bandit. As the system gathers data sparsely, only from the system's own actions, the key concern is to manage the trade-off between exploration (to gather more data for a better estimation of the reward function) and exploitation (to realize utmost rewards based on the data gathered so far) (Krause and Ong, 2011). The bandit algorithms may also benefit from contextual information of the actions or the target users (Langford and Zhang, 2008; Li et al., 2010; Joachims et al., 2018). In many cases, this trade-off is managed in an online fashion, necessitating experimental control over the system (Kawale et al., 2015).

This work focuses on a conceptually distinct problem, i.e., learning from *logged* bandit feedback (Swaminathan and Joachims, 2015b). In this case, batch learning is based on the existing logs of bandit feedback, instead of managing the explore-exploit trade-off online. For one advantage, it does not require experimental control over the system, making such studies more accessible. For another, it enables reuse of existing data, benefitting from cross-validation and offline model selection.

Problem. In particular, we are interested in learning from *logged bandit feedback* where there is also relevant *organic feedback* (Rohde et al., 2018). Take a scenario where a user interacts with products on an e-commerce site and in so doing generates organic feedback. Occasionally, the user may visit another "target" site (e.g., news), where she may be shown an ad featuring a product from the aforementioned e-commerce site. Her responses to the ads on the target site make up the bandit feedback. Our objective is to predict how the user would respond to an ad, in order to predict which ad to show to her the next time she visits the target site, based on the logs of both organic and bandit feedbacks.

Existing approaches (Swaminathan and Joachims, 2015a,b) tend to rely on explicit bandit feedback alone. In practice, relative to the numerous possible items to recommend, the observed data is sparse as we see user responses to recommended ads only. Moreover, it uses organic feedback merely as generic features, without recognizing its potential to learn an informative representation of user preferences.

Contributions. In this paper, we make several contributions. The *first* is our proposed model VLIB, which encodes two principles. For one, we observe that beyond a user's explicit response to an ad (click/no-click), we could potentially infer further *implicit* preference signals relating a clicked ad and previously unclicked ads. Therefore, we propose a probabilistic framework for learning user preferences from bandit feedback, which includes an adequate likelihood function for such implicit bandit feedback. For another, we introduce a latent variable model to robustly capture user preference distributions from both organic-cum-bandit feedbacks.

Secondly, we conduct rich analyses to investigate issues that affect learning and prediction, such as the effect of re-weighting likelihood using inverse propensity score as well as MAP estimate vs. Monte Carlo based approximation to speed up

[1] https://www.lotame.com/what-is-behavioral-targeting/

the predictions. *Thirdly*, we conduct experiments covering both simulated as well as real datasets to address research questions concerning the above contributions.

2 Related Work

In contrast to online learning with contextual bandit feedback (Langford and Zhang, 2008; Li et al., 2010; Joachims et al., 2018), our work is along the line of batch learning from bandit feedback (Swaminathan and Joachims, 2015a,b), based on log data recorded from search engines, recommender systems, etc. However, such data tend to be proprietary, posing some barriers to open research. Fortunately, recently there emerge simulation systems for recommender systems (Rohde et al., 2018; Ie et al., 2019) which presents a platform for organic-bandit recommendation problem with A/B testing evaluation. We experiment with one such platform RecoGym (Rohde et al., 2018), as well as real datasets from Taobao.com.

In estimating the click-through probability of a recommendation, our problem is related to click-through-rate prediction (Richardson et al., 2007; Guo et al., 2017; Zhou et al., 2018; Lian et al., 2018). Such works are typically formulated as supervised learning, predominately relying on user, item, or context features, rather than organic-bandit feedback as in our case. They also focus on offline evaluation on popular benchmark datasets[2] [3]. It has been documented that offline evaluation and online performance are not always congruent (Beel et al., 2013; Garcin et al., 2014; Rossetti et al., 2016).

Our problem is different from session-based recommendation for next-item prediction (Hidasi and Tikk, 2016; Hidasi et al., 2016; Zhou et al., 2018), which is closer to the notion of organic feedback. Along the same line, latent variable models (Blei and Lafferty, 2006; Kingma and Welling, 2014; Rezende et al., 2014) have been successfully applied for collaborative filtering (Liang et al., 2018) in the context of organic feedback. In contrast, our target is prediction in the bandit setting, with the benefit of organic feedback. Also, organic-bandit recommendation might seem to be related to the notion of human-recommender system feedback loop (Bottou et al., 2013; Sun et al., 2019), which is generally relevant to recommendation. However, our work focuses on the problem of computational advertising where users leave publisher sites upon clicking on recommended advertisement.

The notion of implicit feedback has been explored under the context of recommendation (Hu et al., 2008; Rendle et al., 2009). The main idea is making an assumption that observed user events indicate stronger preferences than unobserved ones. Such assumption has shown to be effective in mitigating sparsity issue in learning from preference data. Our work is relevant in that the modeling assumption shares similar characteristic to deal with insufficient observations.

3 Problem Formulation

Figure 1 shows two end-points of the Web that we care about. One is an e-commerce site (e.g., Amazon, Taobao), whereby users are operating in the organic

[2] https://www.kaggle.com/c/avazu-ctr-prediction

[3] https://labs.criteo.com/2013/12/download-terabyte-click-logs/

Table 1: Summary of main notations.

Notation	Explanation
$\mathbf{x}$	User events in the organic state
$\mathbf{a}$	Action or recommendation in the bandit state
$\mathbf{c}$	Binary response variable (e.g., click or no-click)
$\mathcal{T}$	Collection of triplets in logged bandit feedback
$\mathcal{A}$	Universal set of possible bandit actions
$\mathcal{P}$	Universal set of products on e-commerce sites
$\mathbf{z}$	Latent representation of user preferences
f_θ, g_ψ	Functions of generative model, parameterized by θ and ψ
μ_ϕ, σ_ϕ	Functions of inference model, parameterized by ϕ, that output the mean and covariance of the variational distribution of $\mathbf{z}$
ξ	Logging policy deployed when bandit feedback being collected

state (**O**). The other is a publisher site (e.g., The New York Times, Facebook), whereby users are browsing in the bandit state (**B**). Figure 1 illustrates the transition between states of user sessions. A session begins at the state (**S**). At first, she is in the organic state. She can then transition between organic and bandit states, eventually terminating at the end state (**E**).

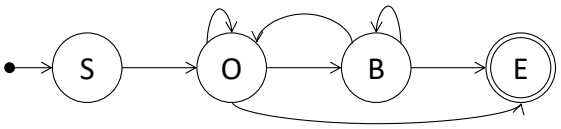


Fig. 1: State transitions of user sessions.

We are interested in a recommender system for the publisher site, which provides recommendation (e.g., displays advertisement) to users. Suppose the publisher site has a logging policy ξ to collect user feedback. While in bandit state, at time index i, suppose that based on organic feedback $\mathbf{x}_i$ (from the publisher site), the system recommends an action $\mathbf{a}_i$, to which a user provides click feedback $\mathbf{c}_i$ (on the target site). This forms a triplet $(\mathbf{x}_i, \mathbf{a}_i, \mathbf{c}_i)$ for each time point. The collection of such triplets $\mathcal{T} = \{(\mathbf{x}_i, \mathbf{a}_i, \mathbf{c}_i)\}$ constitute the logged bandit feedback. In the scope of this work, we focus on the setting of $\mathbf{x}_i$ being product browsing history of users while they are in the organic state (browsing e-commerce websites).

Problem 1 (Organic-Bandit Recommendation). *Given logged bandit feedback $\mathcal{T}$, we seek to estimate, for some time i, the probability $p(\mathbf{c}_i|\mathbf{x}_i, \mathbf{a}_i)$ that a user associated with organic feedback $\mathbf{x}_i$ will respond positively to an action $\mathbf{a}_i$.*

4 Proposed Framework: VLIB

We now describe our proposed model that is called *Variational Learning from Implicit Bandit feedback* or VLIB. We first outline the likelihood function resulting from the proposed implicit bandit feedback. Thereafter, we discuss the variational learning model that would optimize for that likelihood function. For ease of reference, we summarize the notations in Table 1.

4.1 Likelihood Function for Implicit Bandit Feedback

Given an observed triplet from the logged bandit feedback $(\mathbf{x}_i, \mathbf{a}_i, \mathbf{c}_i)$, we would like to learn a statistical model based on an assumption on how the bandit feedback would have been generated. Assuming a Bernoulli process, we have:

$$\mathbf{c}_i \sim \text{Bernoulli}(\sigma(g_\psi(\mathbf{x}_i, \mathbf{a}_i)))$$

where $\sigma(.)$ is the sigmoid function and $g_\psi(.,.)$ is some function, parameterized by ψ, capturing interactions between $\mathbf{x}_i$ and $\mathbf{a}_i$. Learning ψ can be done via maximum likelihood principle. To avoid clutter in the notation, in the following we would derive the log-likelihood for one data point. In turn, the log-likelihood of the dataset can be obtained by averaging over all the observations:

$$\log p_\psi(\mathbf{c}_i|\mathbf{x}_i, \mathbf{a}_i) = \mathbf{c}_i \log \sigma(g_\psi(\mathbf{x}_i, \mathbf{a}_i)) + (1 - \mathbf{c}_i) \log(1 - \sigma(g_\psi(\mathbf{x}_i, \mathbf{a}_i))) \quad (1)$$

From the log-likelihood function above, we need to model the probability $p(\mathbf{c}_i|\mathbf{x}_i, \mathbf{a}_i)$ of an action $\mathbf{a}_i$ being clicked, given user's organic events $\mathbf{x}_i$. In real scenarios, we may only observe logged $(\mathbf{x}_i, \mathbf{a}_i, \mathbf{c}_i)$ for one particular action $\mathbf{a}_i$ determined by the logging policy ξ, and not for the other possible actions $\mathcal{A} \setminus \{\mathbf{a}_i\}$. Due to this nature of logged bandit feedback, it is especially challenging to estimate the clicked probability distribution because of insufficient observations. Important as they are, positive observations $(\mathbf{x}_i, \mathbf{a}_i, \mathbf{c}_i = 1)$ (i.e., recommended actions being clicked) are relatively rare. Furthermore, they intensity the latter phenomenon that a positive observation usually means that we may not observe other occurrences of negative observations involving $\mathbf{x}_i$ and other actions $\mathcal{A} \setminus \{\mathbf{a}_i\}$.

To rectify the latter in particular, we seek to leverage the concept of implicit feedback, which has found great success in mitigating the sparsity issue in collaborative filtering (Rendle et al., 2009; Hu et al., 2008). Upon observing a positive feedback instance involving an action $\mathbf{a}_i$, we presume that all other actions in $\mathcal{A}$ are negative. This effectively induces a set of pseudo-observations involving the same organic events $\mathbf{x}_i$, which we refer to as implicit negative feedback. Reasonably, such implicitly negative pseudo-observations would be treated with a lower 'confidence' than the explicitly positive observations. We thus derive a new log-likelihood taking into account the implicit negative feedback:

$$\begin{aligned} \log p_\psi(\mathbf{c}_i|\mathbf{x}_i, \mathbf{a}_i) = \; & \mathbf{c}_i \log \sigma(g_\psi(\mathbf{x}_i, \mathbf{a}_i)) \\ & + (1 - \mathbf{c}_i) \log(1 - \sigma(g_\psi(\mathbf{x}_i, \mathbf{a}_i))) \\ & + \sum_{\mathbf{a}_j \in \mathcal{A} \setminus \{\mathbf{a}_i\}} \lambda \mathbf{c}_i \log(1 - \sigma(g_\psi(\mathbf{x}_i, \mathbf{a}_j))) \end{aligned} \quad (2)$$

where $\mathcal{A}$ is the set of actions, and hyper-parameter λ controls how confident we are about the implicit negative feedback. The value of λ lies in range of $[0, 1]$, in which $\lambda = 1$ implies certainty, and with $\lambda = 0$ we recover Eq. 1 that models only explicit observations. In other words, $\lambda < 1$ recognizes that a pseudo-observation instance would not be more important than an explicit observation instance.

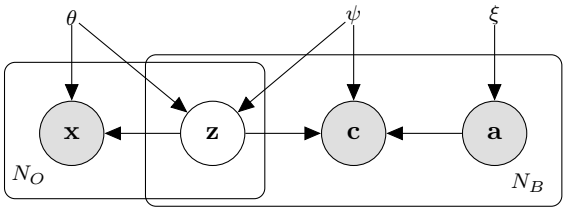


Fig. 2: Graphical representation of VLIB generative model. N_O and N_B are the number of organic sessions and bandit events, respectively.

4.2 Variational Learning for User Preferences

Latent Gaussian model has shown success in learning meaningful representations from data (Kingma and Welling, 2014; Rezende et al., 2014; Miao et al., 2016), especially for collaborative filtering (Liang et al., 2018). In our proposed model, we seek to learn a good D-dimensional variational latent representation $\mathbf{z}_i$ encoding user preference, that would result in better approximation of click probability $p(\mathbf{c}_i|\mathbf{z}_i, \mathbf{a}_i)$ for recommendations. We consider the following generative process:

$$\begin{aligned}
&\mathbf{z}_i \sim \mathcal{N}(0, \mathbf{I}_D) \\
&\mathbf{x}_i \sim \text{Multinomial}(N_i, \pi(f_\theta(\mathbf{z}_i))) \\
&\mathbf{a}_i \sim \text{Categorical}(|\mathcal{A}|, \xi) \\
&\mathbf{c}_i \sim \text{Bernoulli}(\sigma(g_\psi(\mathbf{z}_i, \mathbf{a}_i)))
\end{aligned}$$

where the latent representation $\mathbf{z}_i$ is sampled from a standard Gaussian prior. It is transformed via function $f_\theta(\mathbf{z}_i)$ and $g_\psi(\mathbf{z}_i, \mathbf{a}_i)$ to produce probability distributions from which the organic events $\mathbf{x}_i$ and the bandit event $\mathbf{c}_i$ are drawn, respectively. The organic events $\mathbf{x}_i$, represented as a bag-of-words vector, are presumably sampled from a multinomial distribution, $\pi(.)$ is the softmax function, and the total number of the organic events is $N_i = \sum_k \mathbf{x}_{ik}$. The bandit action $\mathbf{a}_i$, given by the logging policy ξ, is assumed to be sampled from a categorical distribution and represented as an one-hot vector. The bandit event $\mathbf{c}_i$ is sampled from a Bernoulli distribution, where the function $g_\psi(.,.)$ now receives $(\mathbf{z}_i, \mathbf{a}_i)$ as input.

Given $\mathbf{x}_i \perp\!\!\!\perp \mathbf{c}_i \,|\, \mathbf{z}_i$, the joint log-likelihood of $\mathbf{x}_i$ and $\mathbf{c}_i$ can be decomposed as:

$$\log p_{\psi,\theta}(\mathbf{x}_i, \mathbf{c}_i|\mathbf{z}_i, \mathbf{a}_i) = \log p_\theta(\mathbf{x}_i|\mathbf{z}_i) + \log p_\psi(\mathbf{c}_i|\mathbf{z}_i, \mathbf{a}_i) \tag{3}$$

with the log-likelihood for organic events is:

$$\log p_\theta(\mathbf{x}_i|\mathbf{z}_i) = \sum_k \mathbf{x}_{ik} \log \pi_k(f_\theta(\mathbf{z}_i)) \tag{4}$$

and the log-likelihood for the bandit event follows Eq. 2:

$$\begin{aligned}
\log p_\psi(\mathbf{c}_i|\mathbf{z}_i, \mathbf{a}_i) = \; & \mathbf{c}_i \log \sigma(g_\psi(\mathbf{z}_i, \mathbf{a}_i)) \\
& + (1 - \mathbf{c}_i) \log(1 - \sigma(g_\psi(\mathbf{z}_i, \mathbf{a}_i))) \\
& + \sum_{\mathbf{a}_j \in \mathcal{A} \setminus \{\mathbf{a}_i\}} \lambda \mathbf{c}_i \log(1 - \sigma(g_\psi(\mathbf{z}_i, \mathbf{a}_j)))
\end{aligned} \tag{5}$$

4.3 Optimization

To learn the parameters $\{\psi, \theta\}$ of the generative model, for each observation, we need to approximate the posterior distribution $p(\mathbf{z}_i|\mathbf{x}_i, \mathbf{a}_i, \mathbf{c}_i)$, which is intractable. Variational inference technique (Jordan et al., 1999) allows us to approximate the true intractable distribution with a simpler distribution $q(\mathbf{z}_i)$. Here we use Gaussian distribution with diagonal covariance matrices:

$$q(\mathbf{z}_i) = \mathcal{N}(\boldsymbol{\mu}_i, \text{diag}\{\boldsymbol{\sigma}_i\})$$

For system scalability, it is nigh impossible to learn free variational parameters $\{\boldsymbol{\mu}_i, \boldsymbol{\sigma}_i\}$ for each observation, especially in recommendation scenario which we are dealing with. Amortized inference (Kingma and Welling, 2014) offers a solution by learning an inference model to produce data-dependent variational distributions:

$$q_\phi(\mathbf{z}_i|\mathbf{x}_i, \mathbf{a}_i, \mathbf{c}_i) = \mathcal{N}(\mu_\phi(\mathbf{x}_i, \mathbf{a}_i \odot \mathbf{c}_i), \text{diag}\{\sigma_\phi(\mathbf{x}_i, \mathbf{a}_i \odot \mathbf{c}_i)\})$$

where $\mu_\phi(.,.)$ and $\sigma_\phi(.,.)$ are functions, parameterized by ϕ, that output the variational parameters. $\odot$ denotes the element-wise multiplication.

Under variational inference framework, learning latent variable models boils down to maximizing the lower-bound of the marginal log-likelihood over observations (Blei et al., 2017). The parameters of the variational distributions are learned so that Kullback-Leibler divergence $\text{KL}(q(\mathbf{z}_i)||p(\mathbf{z}_i|\mathbf{x}_i, \mathbf{a}_i, \mathbf{c}_i))$ is minimized. For each bandit event, we optimize:

$$\begin{aligned}\log p_{\psi,\theta}(\mathbf{c}_i, \mathbf{x}_i|\mathbf{a}_i) &\geq \mathbb{E}_{q_\phi(\mathbf{z}_i|\mathbf{x}_i,\mathbf{a}_i,\mathbf{c}_i)}[\log p_{\psi,\theta}(\mathbf{c}_i, \mathbf{x}_i|\mathbf{z}_i, \mathbf{a}_i)] \\ &\quad - \text{KL}(q_\phi(\mathbf{z}_i|\mathbf{x}_i, \mathbf{a}_i, \mathbf{c}_i)||p(\mathbf{z}_i)) \\ &= \mathcal{L}(\psi, \theta, \phi; \mathbf{x}_i, \mathbf{a}_i, \mathbf{c}_i) \end{aligned} \tag{6}$$

This objective function, or evidence lower bound (ELBO), is estimated by sampling $\mathbf{z}_i \sim q_\phi$ and maximized using stochastic gradient ascent. One challenge during optimization is to take the gradients with respect to ϕ. Using *re-parameterization trick* (Kingma and Welling, 2014; Rezende et al., 2014), we derive an unbiased Monte Carlo estimator of the ELBO, which yields:

$$\tilde{\mathcal{L}}(\psi, \theta, \phi; \mathbf{x}_i, \mathbf{a}_i, \mathbf{c}_i) = \sum_{\mathbf{x}_i, \mathbf{a}_i, \mathbf{c}_i} [\log p_{\psi,\theta}(\mathbf{c}_i, \mathbf{x}_i|\tilde{\mathbf{z}}_i, \mathbf{a}_i) - \text{KL}(q_\phi(\mathbf{z}_i|\mathbf{x}_i, \mathbf{a}_i, \mathbf{c}_i)||p(\mathbf{z}_i))] \tag{7}$$

where we re-parameterize $\mathbf{z}_i = \mu_\phi(\mathbf{x}_i, \mathbf{a}_i, \mathbf{c}_i) + \boldsymbol{\epsilon} \odot \sigma_\phi(\mathbf{x}_i, \mathbf{a}_i, \mathbf{c}_i)$ with $\boldsymbol{\epsilon}$ is sampled from $\mathcal{N}(0, \mathbf{I}_K)$.

Algorithm 1 sketches the parameter learning procedure. Input data is a collection $\mathcal{T} = \{(\mathbf{x}_i, \mathbf{a}_i, \mathbf{c}_i)\}_{i=1}^N$, where each instance $(\mathbf{x}_i, \mathbf{a}_i, \mathbf{c}_i)$ consists of user's organic events $\mathbf{x}_i$, bandit recommendation $\mathbf{a}_i$, and bandit feedback $\mathbf{c}_i$. Model parameters $\{\theta, \psi, \phi\}$ are updated to maximize the ELBO (Eq. 6) using gradient ascent. In practice, instead of online stochastic update as described, we employ mini-batch gradient ascent to speed up the learning with parallel computation. Each mini-batch $\mathcal{B} = \{(\mathbf{x}_i, \mathbf{a}_i, \mathbf{y}_i)\}_i^{batch\_size}$, uniformly sampled from the collection $\mathcal{T}$, is used to estimate the gradients instead of a single observation. Consequently, the optimization is more stable, and the time for the model to converge reduces drastically.

Algorithm 1: Parameter learning with stochastic gradient ascent

Data: $\mathcal{T} = \{(\mathbf{x}_i, \mathbf{a}_i, \mathbf{c}_i)\}_{i=1}^{N}$
Result: Learned parameters $\{\theta, \psi, \phi\}$
$\eta \leftarrow$ learning rate;
$\theta, \psi, \phi \leftarrow$ randomly initialized;
while *not converged* **do**
  forall $(\mathbf{x}_i, \mathbf{a}_i, \mathbf{c}_i) \in \mathcal{T}$ **do**
    $\boldsymbol{\mu}_i = \mu_\phi(\mathbf{x}_i, \mathbf{a}_i \odot \mathbf{c}_i)$;
    $\boldsymbol{\sigma}_i = \sigma_\phi(\mathbf{x}_i, \mathbf{a}_i \odot \mathbf{c}_i)$;
    Sample $\tilde{\mathbf{z}}_i \sim \mathcal{N}(\boldsymbol{\mu}_i, \text{diag}\{\boldsymbol{\sigma}_i\})$;
    $\theta = \theta + \eta \cdot \frac{\partial}{\partial \theta}[\log p_{\psi,\theta}(\mathbf{c}_i, \mathbf{x}_i | \tilde{\mathbf{z}}_i, \mathbf{a}_i) - \text{KL}(q_\phi(\mathbf{z}_i | \boldsymbol{\mu}_i, \boldsymbol{\sigma}_i) || p(\mathbf{z}_i))]$;
    $\psi = \psi + \eta \cdot \frac{\partial}{\partial \psi}[\log p_{\psi,\theta}(\mathbf{c}_i, \mathbf{x}_i | \tilde{\mathbf{z}}_i, \mathbf{a}_i) - \text{KL}(q_\phi(\mathbf{z}_i | \boldsymbol{\mu}_i, \boldsymbol{\sigma}_i) || p(\mathbf{z}_i))]$;
    $\phi = \phi + \eta \cdot \frac{\partial}{\partial \phi}[\log p_{\psi,\theta}(\mathbf{c}_i, \mathbf{x}_i | \tilde{\mathbf{z}}_i, \mathbf{a}_i) - \text{KL}(q_\phi(\mathbf{z}_i | \boldsymbol{\mu}_i, \boldsymbol{\sigma}_i) || p(\mathbf{z}_i))]$;
  end
end
return $\{\theta, \psi, \phi\}$;

Perspectives on optimizing the ELBO. (Higgins et al., 2016) propose a modification of the optimization objective, reminiscent of Eq. 6, by introducing a hyper-parameter β controlling the effect of Kullback–Leibler divergence as follows:

$$\begin{aligned}\mathcal{L}(\psi, \theta, \phi; \mathbf{x}_i, \mathbf{a}_i, \mathbf{c}_i) = \mathbb{E}_{q_\phi(\mathbf{z}_i | \mathbf{x}_i, \mathbf{a}_i, \mathbf{c}_i)}[\log p_{\psi,\theta}(\mathbf{c}_i, \mathbf{x}_i | \mathbf{z}_i, \mathbf{a}_i)] \\ - \beta \times \text{KL}(q_\phi(\mathbf{z}_i | \mathbf{x}_i, \mathbf{a}_i, \mathbf{c}_i) || p(\mathbf{z}_i))\end{aligned} \tag{8}$$

When setting $\beta \neq 1.0$, we are no longer maximizing a lower bound of the log marginal likelihood. One perspective to look at the optimization, when $\beta > 1.0$, is learning disentangled representations of the data. Increasing β will force the posterior to be close to the prior (isotropic Gaussian). In turn, the learnt representations will be more independent in each of their latent dimensions, which improves the degree of disentanglement. This is important when the objective is having more control and interpretation over newly generated samples. However, our goal is not sampling more user organic and bandit histories, but rather a good predictive accuracy on future observations. Another perspective is to view the KL term in Eq. 8 as regularization factor. With that, we are more interested in the scenario, when $\beta < 1.0$, in which the model is putting more of its capacity on maximizing the likelihood. In other words, the focus is on maximizing *negative* reconstruction error while having weaker constraint on the form of the posterior distribution. Under this perspective, determining a proper amount of regularization, by selecting a good setting of hyper-parameter β, would potentially lead to better predictive performance of our model given a specific dataset.

Despite potential benefit from choosing a good value for β, our main focus is on analysing the effectiveness of the proposed likelihood function with implicit bandit feedback. Therefore, if not explicitly mentioned, the value of β is set to 1.0 by default, which is equivalent to maximizing the original ELBO (Eq. 6). For completeness, we still conduct experiment with varying values of $\beta \in [0, 1]$ in conjunction with our new likelihood, and provide in-depth analysis, later in RQ#3. One important aspect that we would like to emphasize is choosing value for λ (Eq. 2), which determines our likelihood function, is orthogonal to searching for a good value of β, which controls the model regularization. Alemi et al.

(2018) provide a perspective on the effect of β in maximizing the ELBO under the information-theoretic framework. Obeying that interpretation, choosing λ is defining a *Rate-Distortion (RD-plane)* for the model to operate on, while choosing β is searching for an optimal point in that *RD-plane*. Such optimal point is a good balance between the *distortion (D)* measuring the reconstruction error over the samples in the training set, and the *rate (R)* measuring the relative KL divergence between the encoding distribution and $p(\mathbf{z})$. Thus, ones should look for a suitable value of λ given the problem at hand before optimizing the value of β.

4.4 Prediction

Our goal is to provide recommendation to users while they are in the bandit state. In order to do so, we are interested in estimating the following:

$$p(\mathbf{c}|\mathbf{x}, \mathbf{a}) = \int p(\mathbf{c}|\mathbf{z}, \mathbf{a}) p(\mathbf{z}|\mathbf{x}) d\mathbf{z} \approx \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})} p_\psi(\mathbf{c}|\mathbf{z}, \mathbf{a})$$

We compute the expectation using Monte Carlo based approximation, first by drawing S samples:

$$\mathbf{z}^{(s)} \sim \mathcal{N}(\mu_\phi(\mathbf{x}, \mathbf{0}), \text{diag}\{\sigma_\phi(\mathbf{x}, \mathbf{0})\})$$

and then compute:

$$p(\mathbf{c}|\mathbf{x}, \mathbf{a}) = \frac{1}{S} \sum_{s}^{S} p(\mathbf{c}|\mathbf{z}^s, \mathbf{a})$$

Another solution is to apply MAP estimate for $\mathbf{z}_i$ as follows:

$$p(\mathbf{c}|\mathbf{x}, \mathbf{a}) = p(\mathbf{c}|\mu_\phi(\mathbf{x}, \mathbf{0}), \mathbf{a})$$

The latter will get rid of the sampling process and produce fast approximations (in exchange for the loss of information captured by the covariances).

To produce a recommendation, we rank all possible actions based on the probabilities $p(\mathbf{c}|\mathbf{x}, \mathbf{a})$. The best action can be chosen in greedy fashion.

$$\mathbf{a}^\star = \underset{\mathbf{a}}{\text{argmax}}\, p(\mathbf{c}|\mathbf{x}, \mathbf{a})$$

4.5 Complexity Analysis

Assuming that f_θ, g_ψ, μ_ϕ, and σ_ϕ are linear transformations, given a collection of logged bandit feedback $\mathcal{T}$, the computational complexity (i.e., the number of floating point operations) for one optimization epoch is $\mathcal{O}(|\mathcal{T}| \times (|\mathcal{A}| + |\mathcal{P}|) \times 4D)$. A computational burden in our approach is to approximate the multinomial distribution $\pi(f_\theta(\mathbf{z}))$ when the universal set of products $\mathcal{P}$ in the organic state is big. This is a common challenge in statistical modeling (e.g., learning a language model when the size of vocabulary is huge). If this computation becomes a bottleneck, it can be mitigated by well-developed efficient sampling method (Botev et al., 2017) or other approximation techniques (Chen et al., 2016; Morin and Bengio, 2005).

For prediction, the reconstruction of organic events $\mathbf{x}$ is not required. Each recommendation takes $\mathcal{O}((2|\mathcal{A}|+|\mathcal{P}|)\times D)$ if using MAP estimation, and $\mathcal{O}((2|\mathcal{A}|+|\mathcal{P}|)\times 2D + S \times D)$ if using MC approximation (S is the number of drawing samples).

5 Experiments

Our objective is to evaluate the performance of the proposed model VLIB as compared to other learning approaches on the logged bandit feedback. In particular, we would organize the experimental analysis along several research questions on learning effective model as well as generating predictions efficiently for scalable online recommender systems.

5.1 Datasets

Bandit data tend to be proprietary, posing some barriers to open research. Fortunately, recently there emerge simulation systems for recommender systems (Rohde et al., 2018; Ie et al., 2019) that provide a platform for organic-bandit recommendation problem with A/B testing evaluation.

- We first conduct our experiments on simulated data from *RecoGym* (Rohde et al., 2018) simulation enviroment of product recommendation in online advertising. Using RecoGym, we evaluate all the agents with two settings of 100 and 1000 products, denoted as **RG-100** and **RG-1000**, respectively.
- We also experiment on real data [4] of online advertising display on *Taobao.com* e-commerce website. The dataset comes with users' organic behaviors (product browsing, adding to the shopping cart, favoring, buying) and ad bandit click events. We filter duplicate records and retain the logs of 2000 most frex quent brands, which yields 294,191,912 organic events and 26,557,962 bandit events by 1,129,944 users in total. In the end, we create two datasets with the number of brands are 500 and 2000, **TB-500** and **TB-2000**, respectively. To simulate user sessions, organic and bandit events of each user are lined up based on their timestamps. This is a standard experimental procedure for session-based recommendation in which we seek to model users' future adoption.

The evaluation scheme consists of offline training and online testing. In the training phase, the models receive logs of 1000 users as training data (approximately 80,000 bandit events for RG-100 and RG-1000, and 23,000 bandit events for TB-500 and TB-2000). In the testing phase, the models are deployed and evaluated over another 1000 users with roughly the same numbers bandit events.

5.2 Evaluation Metrics

As the main objective is for online advertising, we seek high *Click-Through Rate (CTR)* measured as:

$$\text{CTR} = \frac{\text{number of clicks}}{\text{number of bandit events}} \times 100(\%)$$

In addition, we are interested in the ranking quality of the models in the context of top-K recommendations. Thus, we employ two widely used ranking metrics for recommendation evaluation, *Hit Ratio (HR)* and *Normalized Discounted Cumulative Gain (NDCG)*. Let $\mathbf{r}_K^{(t)} \subset \mathcal{A}$ be the K actions with the highest predicted

[4] https://tianchi.aliyun.com/dataset/dataDetail?dataId=56

probabilities $\{p(\mathbf{c}_t|\mathbf{x}_t, \mathbf{a}_t) \mid \mathbf{a}_t \in \mathbf{r}_K^{(t)}\}$ at time t (right before user transits to the organic state), and $v^{(t+1)}$ the very first viewed product in the organic state. We compute *HR@K* and *DCG@K* as follows:

$$\text{HR@}K = \begin{cases} 1 & \text{if } v^{(t+1)} \in \mathbf{r}_K^{(t)} \\ 0 & \text{otherwise.} \end{cases} \qquad \text{DCG@}K = \sum_{i=1}^{K} \frac{2^{\mathbb{1}[v^{(t+1)} \in \mathbf{r}_K^{(t)}]} - 1}{\log(i+1)}$$

NDCG@K is the DCG@K normalized into $[0, 1]$ after normalizing it with the best possible DCG@K, in which $v^{(t+1)}$ is ranked at the top. For all metrics, we compute the average results over all testing user sessions. The reported numbers are averaged results across 10 independent runs for each experiment.

5.3 Comparative Methods

We compare the proposed VLIB with simple heuristic baselines in recommendation, statistical and deep learning methods for click-through rate prediction, as well as state-of-the-art method for learning from bandit feedback:

- **Random** is the simplest baseline without learning from data. The actions are randomly selected with uniform probability $p(\mathbf{a}) = 1/|\mathcal{A}|$.
- **MostPop** is a simple yet effective baseline in the context of recommendation, selecting action of the most popular item in the organic events.
- **Cooccur** selects the actions that have the highest co-occurrences with the latest organically-viewed item (i.e., it assumes first-order Markov dependency and takes into account the temporal information).
- **MLR** (*Multinomial Logistic Regression*) directly models the probability of an action given the organic events $p(\mathbf{a}|\mathbf{x})$, its goal is to learn a policy that would maximize number of clicks if was being deployed instead of the logging policy ξ, thus, observations are re-weighted by the inverse propensity score of the logging policy $w_i = \mathbf{c}_i/\xi(\mathbf{a}_i)$. More details on this method can be found in (Jeunen et al., 2019).
- **xDeepFM** (*eXtreme Deep Factorization Machines*) (Lian et al., 2018) is a strong method for *click-through rate* prediction. It combines the power of factorization machines (FM) for recommendation, deep neural network (DNN) for capturing feature interactions with the proposed Compressed Interaction Network (CIN). xDeepFM is included as a representative baseline of the family of models, which try to estimate the probability of user-click $p(\mathbf{c}|\mathbf{x}, \mathbf{a})$, treating organic events as input features without further assumption. The model uses second-order FM with the searched-grid of hyper-parameters as follows: $E \in [8, 16, 32]$ is the size of embeddings, $T \in [1, 2, 3]$ is the number of hidden layers in DNN and CIN, $H \in [64, 128, 256]$ is the number of neuron units per layer. Each neuron uses *hyperbolic tangent* as non-linear activation function.
- **POEM** (*Policy Optimizer for Exponential Models*) (Swaminathan and Joachims, 2015b) tackles the counterfactual effect with Counterfactual Risk Minimization (CRM) learning principle. It is considered state-of-the-art for the problem of learning from logged bandit feedback (Jeunen et al., 2019). We follow the authors' recommendation of clipping constant M based on propensity score, and search for the best hyper-parameter $c \in [10^{-6}, ..., 1]$ in multiples of 10.

Table 2: Comparison between VLIB and comparative methods on various datasets.

		CTR(%)	HR@10	NDCG@10
RG-100	Random	1.069 ± 0.050	0.071 ± 0.058	0.038 ± 0.026
	MostPop	1.128 ± 0.332	0.796 ± 0.047	0.496 ± 0.053
	Cooccur	1.400 ± 0.214	0.848 ± 0.030	0.594 ± 0.042
	MLR	1.493 ± 0.258	0.796 ± 0.034	0.604 ± 0.044
	xDeepFM	1.513 ± 0.253	0.763 ± 0.029	0.590 ± 0.046
	POEM	1.542 ± 0.223	0.824 ± 0.036	0.624 ± 0.043
	VLIB	$\mathbf{1.672} \pm 0.263^\star$	$\mathbf{0.879} \pm 0.024^\star$	$\mathbf{0.659} \pm 0.039^\star$
RG-1000	Random	1.069 ± 0.043	0.010 ± 0.019	0.003 ± 0.006
	MostPop	2.043 ± 0.752	0.591 ± 0.093	0.372 ± 0.077
	Cooccur	2.316 ± 0.166	0.656 ± 0.079	0.451 ± 0.076
	MLR	2.460 ± 0.167	0.626 ± 0.079	0.477 ± 0.077
	xDeepFM	2.479 ± 0.174	0.612 ± 0.086	0.422 ± 0.073
	POEM	2.501 ± 0.165	0.634 ± 0.079	0.481 ± 0.078
	VLIB	$\mathbf{2.613} \pm 0.134^\star$	$\mathbf{0.719} \pm 0.065^\star$	$\mathbf{0.522} \pm 0.073^\star$
TB-500	Random	1.073 ± 0.042	0.044 ± 0.081	0.035 ± 0.078
	MostPop	1.823 ± 0.684	0.620 ± 0.117	0.394 ± 0.108
	Cooccur	2.149 ± 0.204	0.685 ± 0.093	0.473 ± 0.100
	MLR	2.297 ± 0.224	0.666 ± 0.083	0.503 ± 0.094
	xDeepFM	2.302 ± 0.236	0.634 ± 0.081	0.465 ± 0.092
	POEM	2.315 ± 0.225	0.674 ± 0.085	0.508 ± 0.095
	VLIB	$\mathbf{2.453} \pm 0.157^\star$	$\mathbf{0.743} \pm 0.077^\star$	$\mathbf{0.544} \pm 0.091^\star$
TB-2000	Random	1.071 ± 0.044	0.003 ± 0.004	0.001 ± 0.001
	MostPop	2.220 ± 0.646	0.512 ± 0.081	0.318 ± 0.064
	Cooccur	2.503 ± 0.088	0.573 ± 0.071	0.386 ± 0.068
	MLR	2.610 ± 0.086	0.556 ± 0.061	0.417 ± 0.064
	xDeepFM	2.613 ± 0.092	0.522 ± 0.075	0.352 ± 0.069
	POEM	2.627 ± 0.101	0.550 ± 0.080	0.418 ± 0.076
	VLIB	$\mathbf{2.678} \pm 0.099^\star$	$\mathbf{0.642} \pm 0.064^\star$	$\mathbf{0.462} \pm 0.067^\star$

* improvements over the second-best baseline are statistically significant with paired sample t-test (p-value < 0.01).

VLIB could learn highly expressive functions f_θ and g_ψ with deep neural networks if such modeling capacity is required to discover complex interactions between $\mathbf{x}$ and $\mathbf{a}$ (see the discussion on Proposed Framework: VLIB). Here, we prioritize efficiency and experiment with simpler linear functions for both f_θ and g_ψ, as these already achieve competitive performances. The number of dimensions for the latent variable $\mathbf{z}$ is $D = 50$ across all datasets, while hyper-parameter λ is searched within $[10^{-4}, \ldots, 1]$ in multiples of 10. The best obtained values for λ are 0.01 on RG-100, RG-1000, TB-500 datasets, and 0.1 on TB-2000 dataset.

5.4 Empirical Results and Discussion

We analyze the empirical results along five research questions (RQ#1 to RQ#5).

RQ#1: How Does VLIB Perform as Compared to the Baselines?
The experimental results in Table 2 show that in many cases, the simple methods MostPop and Cooccur achieve competitive performance to the model-based approaches. Cooccur even surpasses learning methods MLR, xDeepFM, and POEM in terms of HR measurement, though the ranking quality is not as good, as reflected by lower NDCG scores. xDeepFM achieves competitive performance in

Table 3: Effectiveness of including implicit bandit feedback.

		CTR(%)	HR@10	NDCG@10
RG-100	*Exp*	1.177 ± 0.070	0.204 ± 0.043	0.095 ± 0.022
	Imp	1.555 ± 0.216	0.802 ± 0.049	0.563 ± 0.056
RG-1000	*Exp*	1.764 ± 0.124	0.179 ± 0.026	0.094 ± 0.014
	Imp	2.414 ± 0.127	0.571 ± 0.087	0.394 ± 0.072
TB-500	*Exp*	1.607 ± 0.103	0.195 ± 0.031	0.101 ± 0.017
	Imp	2.248 ± 0.154	0.619 ± 0.105	0.427 ± 0.099
TB-2000	*Exp*	2.000 ± 0.110	0.182 ± 0.024	0.104 ± 0.017
	Imp	2.568 ± 0.081	0.480 ± 0.088	0.337 ± 0.071

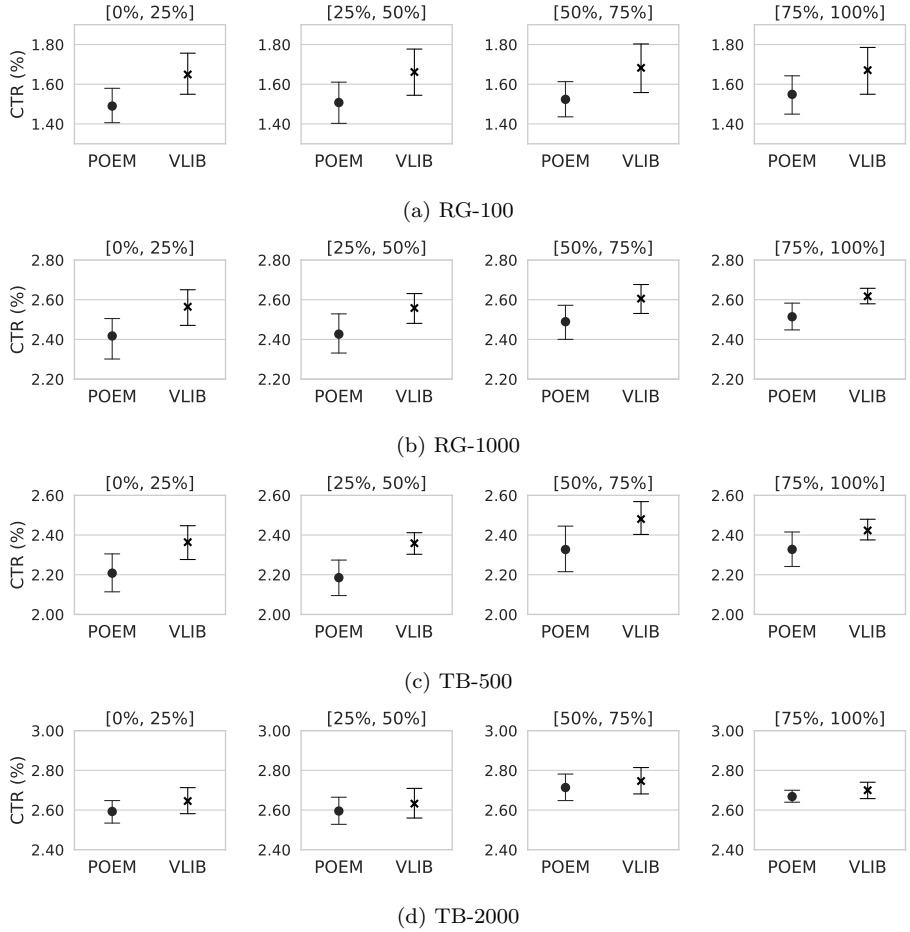
terms of CTR as it is what the model is designed for. However, the model performs poorly for top-K ranking metrics (i.e., HR and NDCG). One explanation for that could be on the perspective of learning of the model family which tries to estimate $p(\mathbf{c}|\mathbf{x}, \mathbf{a})$. The model is lacking negative samples to contrast with the positive ones (i.e., clicked recommendations). This affirms our motivation to come up with the notion of implicit bandit feedback and the corresponding likelihood function (Eq. 2). POEM consistently shows better performance than both xDeepFM and MLR, especially in CTR metric. The improvement can be credited to better learning algorithm derived from CRM.

Evidently, VLIB achieves the highest performance. The gaps are notable especially in terms of the top-K recommendation metrics. We attribute that to the contribution of learning better representation of preferences via generative modeling of observational events together with implicit bandit feedback. Thus, VLIB can better rank the actions as compared to MLR and POEM, that directly optimize for determining only the best action. We statistically test the performance of VLIB against the second best method POEM using paired samples t-test, and find VLIB to be significantly better than POEM across all metrics. This suggests that VLIB is an effective approach for dealing with organic-bandit recommendation.

In Figure 3, we report the performance of VLIB in terms of click-through rate with different level of user's organic activity (how many organic events by a user before she enters a bandit state). POEM which is the second-best baseline (Table 2) is also included as a reference compared to VLIB. Overall, a clear trend is that both models perform better when observing more organic events. In other words, users' preferences are being captured from their organic feedback, which turns into more accurate recommendations during bandit state. Among the two methods, VLIB is consistently better than POEM, especially in the lower percentiles (lack of organic events). The gap is closer with higher percentiles (sufficient organic events for modelling user preferences). This result emphasizes that VLIB is suitable for dealing with less organically active users (viewing less items on the organic states), and it also explains the improvements of VLIB over the compared baselines. Furthermore, recommendations by VLIB are more accurate and reliable when observing more organic feedback (error bars shrinking). This is particularly prominent for RG-1000 (Figure 3b) and TB-2000 (Figure 3d).

RQ#2: Does Learning From Implicit Bandit Feedback Improve the Accuracy?

Table 3 reports an experiment comparing the two choices of likelihood functions for learning from logged bandit feedback. *Exp* (see Eq. 1) relies only on explicit



(a) RG-100

(b) RG-1000

(c) TB-500

(d) TB-2000

Fig. 3: Click-through rate (CTR) breakdown with increasing levels of user's organic activity (how many organic events by a user before she enters a bandit state). POEM (second-best baseline) is included as a reference for comparison with VLIB.

feedback. *Imp* (see Eq. 2) is based on both explicit and implicit bandit feedback. We see that the improvement of *Imp* over *Exp* is substantial and consistent. *Imp* generalizes *Exp* and has the advantage of controlling the confidence using λ. The effectiveness of the additional term in the *Imp* likelihood will be further analyzed later when answering a research question on re-weighting samples.

We analyze the effect of implicit bandit feedback on the performance of VLIB. Figure 4 shows the results in terms of CTR while varying the values of hyper-parameter λ. Generally, we observe that the best click-through rate is achieved when $\lambda > 0$, demonstrating the positive impact of implicit bandit feedback on our model. Intuitively, we might think that the optimal value of λ is data-dependent and needs to be carefully selected. However, this experiment suggests that CTR is less sensitive to λ when λ reaches a certain threshold, $\lambda = 0.01$ in this case.

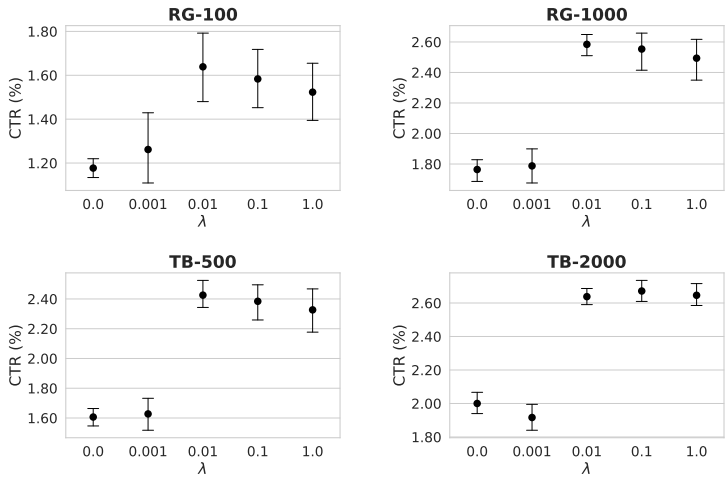


Fig. 4: Effect of the implicit feedback on VLIB in terms of click-through rate (CTR). The y-axis displays CTR, and the x-axis shows varied values of the hyper-parameter λ controlling the certainty of implicit negative feedback (Eq. 2).

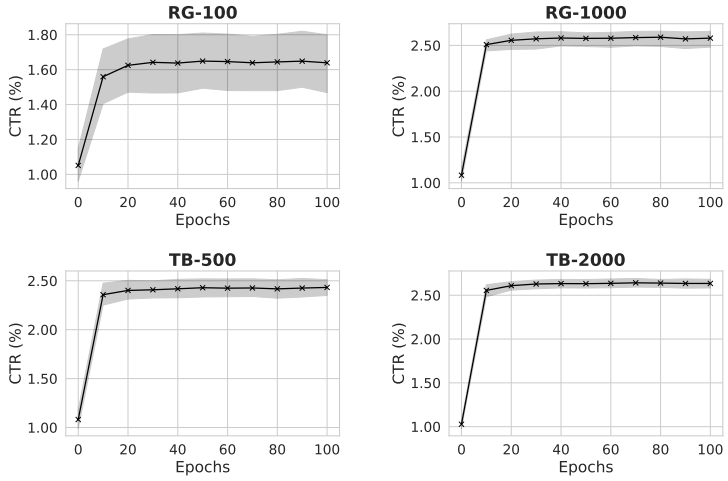


Fig. 5: Convergence of VLIB in terms of click-through rate (CTR). The y-axis displays CTR, and the x-axis shows the number of epochs through training data.

To avoid doing grid search, one heuristic approach to select good value for λ is based on annealing. λ can be set to 1.0 at first and decreasingly annealed during training. While annealing, we perform validation and stop decreasing λ when we notice the validation metric dropping.

Figure 5 illustrates the performance of VLIB in terms of CTR with different number of training epochs. The model achieves good performance after a few epochs, and its results keep improving and stabilizing when we increase the time

Table 4: Effectiveness of learning variational representation of user preferences.

		CTR(%)	HR@10	NDCG@10
RG-100	*Det*	1.555 ± 0.216	0.802 ± 0.049	0.563 ± 0.056
	Var	1.672 ± 0.263	0.879 ± 0.024	0.659 ± 0.039
RG-1000	*Det*	2.414 ± 0.127	0.571 ± 0.087	0.394 ± 0.072
	Var	2.613 ± 0.134	0.719 ± 0.065	0.522 ± 0.073
TB-500	*Det*	2.248 ± 0.154	0.619 ± 0.105	0.427 ± 0.099
	Var	2.453 ± 0.157	0.743 ± 0.077	0.544 ± 0.091
TB-2000	*Det*	2.568 ± 0.081	0.480 ± 0.088	0.337 ± 0.071
	Var	2.678 ± 0.099	0.642 ± 0.064	0.462 ± 0.067

for training. This shows that optimizing the proposed Eq. 2 would lead to improvements in CTR, the main objective when displaying online advertisements.

RQ#3: How Effective Is Learning Variational Representation of User Preferences?

One contribution in the proposed framework is to learn a latent variable model that can explain both organic and bandit feedbacks. We examine the effect of representing user preferences in a low D-dimensional space as opposed to directly modeling the relationship between $\mathbf{x}$ and $\mathbf{a}$. We denote the former as *Var* (for variational), and the latter as *Det* (for deterministic). For parity, we only learn linear mapping functions f_θ, g_ψ, as well as μ_ϕ and σ_ϕ for *Var*. In terms of model capacity, *Var* has fewer parameters than *Det*, thus has no advantage in memorization.

In Table 4, we can see that *Var*'s outperformance over *Det* is especially remarkable in terms of ranking metrics (*HR* and *NDCG*). From one perspective, *Var* imposes stronger modeling assumptions than *Det*, the prior of latent space, and therefore could be more robust when the feedback is scarce. From another perspective, there is a regularization effect in forcing $\mathbf{z}$ to also explain $\mathbf{x}$, which drives *Var* model away from putting all capacity in being discriminative of $p(\mathbf{c}|\mathbf{x}, \mathbf{a})$, as *Det* model does. That could be an explanation for improvements in the ranking measurements as *Var* can rank actions better than rather just determine the best action. This result proves the effectiveness of learning variational representation of user preferences for better recommendation to the problem at hand.

Figure 6 illustrates the performance of our model in terms of CTR while varying the value of hyper-parameter β (in Eq. 8). As discussed earlier, ones should look for a suitable value of λ (in Eq. 2) before optimizing the value of β. In this experiment, $\lambda = 0.001$ for TB-2000 and $\lambda = 0.01$ for the rest of the datasets as they show the best performance in the previous experiment (see Fig. 4), while the value of β is varied in the range of $[0, 1]$. We observe a clear improvement with $\beta = 0.9$ on TB-2000 dataset while it is negligible on the others. Nevertheless, with $\beta = 1.0$, our model still achieves competitive performance suggesting that optimizing the ELBO lies near the optimal point of the *RD-plane* if properly defined via a good selection of value for λ. Our proposed framework is less sensitive to the value of β, although with a cost of searching it potentially yields an improvement. While β is decreasing towards 0.0, weakening the effect of KL term and approaching the deterministic learning, the CTR performance drastically declines. It vividly showcases the effectiveness of learning variational representation (*Var*) as compared to deterministic representation (*Det*), which reinforces the same observation demonstrated in Table 3.

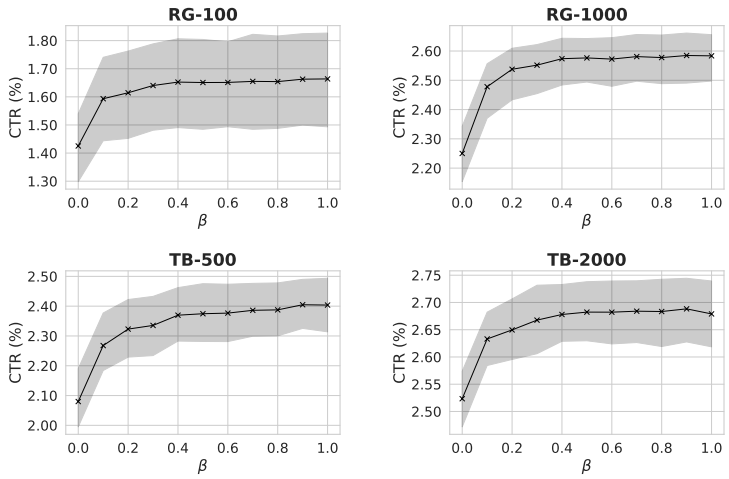


Fig. 6: Effect of the KL regularizer on VLIB in terms of click-through rate (CTR). The y-axis displays CTR, and the x-axis shows varied values of the hyper-parameter β controlling the regularization effect on our model (Eq. 8).

Table 5: Effect of re-weighting likelihood function.

		CTR(%)	HR@10	NDCG@10
RG-100	*NoIPS*	1.672 ± 0.239	0.879 ± 0.024	0.659 ± 0.039
	IPS	1.672 ± 0.263	0.867 ± 0.030	0.627 ± 0.048
RG-1000	*NoIPS*	2.613 ± 0.134	0.719 ± 0.065	0.522 ± 0.073
	IPS	2.563 ± 0.159	0.679 ± 0.080	0.468 ± 0.084

RQ#4: Does Re-weighting Likelihood Function Using IPS Help?
When using a simple model with the standard maximum likelihood approach, the model may underfit and only focus on minimizing error around the common observations $(\mathbf{x}_i, \mathbf{a}_i)$ by the logging policy. The problem has been characterized and commonly known as *covariance shift* (Shimodaira, 2000). One solution is to re-weight the likelihood to compensate for underrepresented samples. In the context of bandit feedback, we can practically achieve that by using inverse propensity score (IPS) of the logging policy $w_i = 1/\xi(\mathbf{a}_i)$. Table 5 reports the comparison of VLIB using the proposed likelihood (Eq. 2) denoted as *NoIPS* and re-weighted version *IPS*. We conduct the experiment on two datasets *RG-100* and *RG-1000*, *TB* datasets are omitted because we do not have access to the logging policy.

Interestingly, the results favor *NoIPS*, i.e., the proposed likelihood without re-weighting tends to perform better. One explanation could be the contribution of the implicit feedback assumption. The additional term to the likelihood function also has re-weighting effect by emphasizing the importance of the positive feedback, which is usually scarce. Furthermore, it augments the data with more observations of organic feedback and negative action pairs $(\mathbf{x}_i, \mathbf{a}_j)$, $\mathbf{a}_j \in \mathcal{A} \setminus \{\mathbf{a}_i\}$.

RQ#5: Can We Use MAP Estimate Instead of MC Sampling for Prediction?

Table 6: MC sampling vs. MAP for prediction.

		CTR(%)	HR@10	NDCG@10
RG-100	*MC*	1.661 ± 0.245	0.881 ± 0.025	0.659 ± 0.039
	MAP	1.672 ± 0.239	0.879 ± 0.024	0.659 ± 0.039
RG-1000	*MC*	2.606 ± 0.135	0.720 ± 0.063	0.523 ± 0.071
	MAP	2.613 ± 0.134	0.719 ± 0.065	0.522 ± 0.073
TB-500	*MC*	2.460 ± 0.161	0.742 ± 0.076	0.544 ± 0.091
	MAP	2.453 ± 0.157	0.743 ± 0.077	0.544 ± 0.091
TB-2000	*MC*	2.706 ± 0.094	0.646 ± 0.073	0.463 ± 0.075
	MAP	2.678 ± 0.099	0.642 ± 0.064	0.462 ± 0.067

There are two ways to obtain predictions from VLIB due to the variational representation of user preferences. Using Monte Carlo sampling, we use the uncertainty captured by the covariance of the variational distributions. This comes at a cost as we need a reasonable number of samples for a stable prediction. It makes deployment of real-time recommender systems challenging. For faster approximation, we apply MAP estimate to only use the mean $\boldsymbol{\mu}$ of $\mathbf{z}$ and ignore the covariance $\boldsymbol{\sigma}$.

Table 6 provides the comparison between the two approaches across the four datasets. For the former approach, denoted as *MC*, we draw 200 samples for each approximation. The latter point estimate approach is denoted as *MAP*. As shown by the results, it is perhaps surprising that just using the posterior mean can perform similarly well to the Monte Carlo approach. The gain by *MC* is marginal in terms of *HR* and *NDCG*, where, there are noticeable differences in terms of *CTR* on *TB* datasets. This result suggests that MAP estimate can be effectively used for deployment of real-time recommender systems with a low cost of accuracy in return for a remarkable gain in efficiency.

6 Conclusion

We address the problem of learning from logged bandit feedback, which is a different scenario from online reinforcement learning in that the former is batch learning from existing logs. The proposed method VLIB optimizes a more adequate likelihood function incorporating implicit negative feedback involving organic events associated with positive bandit feedback. Comprehensive experiments on simulated bandit scenario using RecoGym and real-life datasets from *Taobao.com* yield insightful results. VLIB outperforms comparable baselines comprehensively. We further validate the contributions of modeling components ablatively, such as the proposed implicit feedback (vs. modeling just the explicit user response) and variational learning of user preferences (vs. deterministic learning). In terms of likelihood estimation, we discover that re-weighting using inverse propensity score does not make much difference, while using MAP in place of Monte Carlo sampling provides efficiency gains at minimal accuracy loss.

As future work, we would further investigate the impact of implicit bandit feedback. The objective is to gain more theoretical insights on how not only it improves click-through rate prediction but also eases the need of using inverse propensity score re-weighting, which is not trivial when the access to logging policy is limited.

Acknowledgements This research is supported by the National Research Foundation, Prime Minister's Office, Singapore under its NRF Fellowship Programme (Award No. NRF-NRFF2016-07).

References

A. Alemi, B. Poole, I. Fischer, J. Dillon, R. A. Saurous, and K. Murphy. Fixing a broken elbo. In *ICML*, pages 159–168. PMLR, 2018.

J. Beel, M. Genzmehr, S. Langer, A. Nürnberger, and B. Gipp. A comparative analysis of offline and online evaluations and discussion of research paper recommender system evaluation. In *Proceedings of the international workshop on reproducibility and replication in recommender systems evaluation*, 2013.

D. Blei and J. Lafferty. Correlated topic models. *Advances in neural information processing systems*, 2006.

D. M. Blei, A. Kucukelbir, and J. D. McAuliffe. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 2017.

A. Botev, B. Zheng, and D. Barber. Complementary sum sampling for likelihood approximation in large scale classification. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS 2017*, volume 54. PMLR (Proceedings of Machine Learning Research), 2017.

L. Bottou, J. Peters, J. Quiñonero-Candela, D. X. Charles, D. M. Chickering, E. Portugaly, D. Ray, P. Simard, and E. Snelson. Counterfactual reasoning and learning systems: The example of computational advertising. *The Journal of Machine Learning Research*, 14(1): 3207–3260, 2013.

W. Chen, D. Grangier, and M. Auli. Strategies for training large vocabulary neural language models. In *ACL*, 2016.

F. Garcin, B. Faltings, O. Donatsch, A. Alazzawi, C. Bruttin, and A. Huber. Offline and online evaluation of news recommender systems at swissinfo. ch. In *ACM conference on recommender systems*, 2014.

H. Guo, R. Tang, Y. Ye, Z. Li, and X. He. Deepfm: a factorization-machine based neural network for ctr prediction. In *IJCAI*, 2017.

B. Hidasi and D. Tikk. General factorization framework for context-aware recommendations. *Data Mining and Knowledge Discovery*, 2016.

B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk. Session-based recommendations with recurrent neural networks. In *ICLR*, 2016.

I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. 2016.

Y. Hu, Y. Koren, and C. Volinsky. Collaborative filtering for implicit feedback datasets. In *ICDM*, pages 263–272. IEEE, 2008.

E. Ie, C.-w. Hsu, M. Mladenov, V. Jain, S. Narvekar, J. Wang, R. Wu, and C. Boutilier. Recsim: A configurable simulation platform for recommender systems. *arXiv preprint arXiv:1909.04847*, 2019.

O. Jeunen, D. Mykhaylov, D. Rohde, F. Vasile, A. Gilotte, and M. Bompaire. Learning from bandit feedback: An overview of the state-of-the-art. *arXiv*, 2019.

T. Joachims, A. Swaminathan, and M. de Rijke. Deep learning with logged bandit feedback. In *International Conference on Representation Learning*, 2018.

M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul. An introduction to variational methods for graphical models. *Machine learning*, 37(2):183–233, 1999.

J. Kawale, H. H. Bui, B. Kveton, L. Tran-Thanh, and S. Chawla. Efficient thompson sampling for online matrix-factorization recommendation. In *Advances in neural information processing systems*, 2015.

D. P. Kingma and M. Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations*, 2014.

Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.

A. Krause and C. S. Ong. Contextual gaussian process bandit optimization. In *Advances in neural information processing systems*, 2011.

J. Langford and T. Zhang. The epoch-greedy algorithm for multi-armed bandits with side information. In *Advances in neural information processing systems*, 2008.
L. Li, W. Chu, J. Langford, and R. E. Schapire. A contextual-bandit approach to personalized news article recommendation. In *International World Wide Web Conference*, 2010.
J. Lian, X. Zhou, F. Zhang, Z. Chen, X. Xie, and G. Sun. xdeepfm: Combining explicit and implicit feature interactions for recommender systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1754–1763, 2018.
D. Liang, R. G. Krishnan, M. D. Hoffman, and T. Jebara. Variational autoencoders for collaborative filtering. In *WWW*, 2018.
Y. Miao, L. Yu, and P. Blunsom. Neural variational inference for text processing. In *ICML*, 2016.
F. Morin and Y. Bengio. Hierarchical probabilistic neural network language model. In *AISTATS*, 2005.
S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. In *UAI*, 2009.
D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *ICML*, 2014.
M. Richardson, E. Dominowska, and R. Ragno. Predicting clicks: estimating the click-through rate for new ads. In *WWW*, 2007.
D. Rohde, S. Bonner, T. Dunlop, F. Vasile, and A. Karatzoglou. Recogym: A reinforcement learning environment for the problem of product recommendation in online advertising. *arXiv preprint arXiv:1808.00720*, 2018.
M. Rossetti, F. Stella, and M. Zanker. Contrasting offline and online results when evaluating recommendation algorithms. In *ACM Conference on Recommender Systems*, 2016.
H. Shimodaira. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of statistical planning and inference*, 90(2):227–244, 2000.
W. Sun, S. Khenissi, O. Nasraoui, and P. Shafto. Debiasing the human-recommender system feedback loop in collaborative filtering. In *Companion Proceedings of The 2019 World Wide Web Conference*, pages 645–651, 2019.
R. S. Sutton, A. G. Barto, et al. *Introduction to reinforcement learning*. 1998.
A. Swaminathan and T. Joachims. Batch learning from logged bandit feedback through counterfactual risk minimization. *The Journal of Machine Learning Research*, 2015a.
A. Swaminathan and T. Joachims. Counterfactual risk minimization: Learning from logged bandit feedback. In *ICML*, 2015b.
G. Zhou, X. Zhu, C. Song, Y. Fan, H. Zhu, X. Ma, Y. Yan, J. Jin, H. Li, and K. Gai. Deep interest network for click-through rate prediction. In *ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018.