A General Offline Reinforcement Learning Framework for Interactive Recommendation

Teng Xiao, Donglin Wang[*]
Machine Intelligence Lab (MiLAB), AI Division, School of Engineering, Westlake University
tengxiao01@gmail.com, wangdonglin@westlake.edu.cn

Abstract

This paper studies the problem of learning interactive recommender systems from logged feedbacks without any exploration in online environments. We address the problem by proposing a general offline reinforcement learning framework for recommendation, which enables maximizing cumulative user rewards without online exploration. Specifically, we first introduce a probabilistic generative model for interactive recommendation, and then propose an effective inference algorithm for discrete and stochastic policy learning based on logged feedbacks. In order to perform offline learning more effectively, we propose five approaches to minimize the distribution mismatch between the logging policy and recommendation policy: support constraints, supervised regularization, policy constraints, dual constraints and reward extrapolation. We conduct extensive experiments on two public real-world datasets, demonstrating that the proposed methods can achieve superior performance over existing supervised learning and reinforcement learning methods for recommendation.

1 Introduction

Reinforcement learning (RL) is a powerful paradigm for interactive or sequential recommender systems (RS), since it can maximize users' long-term satisfaction with the system and constantly adapt to users' shifting interest (state). However, training optimal RL algorithms requires large amounts of interactions with environments (Kakade 2003), which is impractical in the recommendation context. Performing online interaction (on-policy) learning would hurt users' experiences and the revenue of the platform. Since logged feedbacks of users are often abundant and cheap, an alternative is to make use of them and to learn a near-optimal recommendation policy offline before we deploy it online.

Although there are some offline (off-policy) RL algorithms have been proposed in the continuous robot control domain (Fujimoto, Meger, and Precup 2019; Kumar et al. 2019), the problem that how to build an effective offline RL framework for recommendation involving large numbers of discrete actions and logged feedbacks remains an open one. Learning an effective recommendation policy from logged feedbacks faces the following challenges simultaneously: **(a)** Discrete stochastic policy. In the testing, instead of recommending only one item, the RS requires generating a top-k item list according a discrete policy. Training a deterministic policy violates the simple machine learning principle: test and train conditions must match (see §4 for details). **(b)** Extrapolation error. Extrapolation error is an error in off-policy value learning which is introduced by the mismatch between the dataset and true state-action visitation of the current policy (Fujimoto, Meger, and Precup 2019; Siegel et al. 2019). This problem is even more serious for the recommendation involving large numbers of discrete actions. **(c)** Unknown logging policy. The feedbacks typically come from a unknown mixture of previous policies. The method that requires estimating logging policy are limited by their ability to accurately estimate the unknown logging policy.

Recent effort (Chen et al. 2019a) for off-policy recommendation tries to alleviate the problem (a) by utilizing the inverse propensity score (IPS) with model-free policy gradient algorithm. However the IPS suffers from high variance (Swaminathan and Joachims 2015a) and will still be biased if the logging policy does not span the support of the optimal policy (Sachdeva, Su, and Joachims 2020; Liu et al. 2019). Moreover, it cannot address other problems mentioned above, i.e., (b) and (c). Prior works (Zheng et al. 2018; Zhao et al. 2018b) try to build RL-based recommendation algorithms by directly utilizing the vanilla Q-learning (Sutton and Barto 2018), an off-policy RL algorithm. However the Q-learning is a deterministic policy method and also suffers from the extrapolation error due to no interaction with online environments (Fujimoto, Meger, and Precup 2019).

To address the aforementioned defects, in this paper, we propose a general and effective offline learning framework for interactive RS. We first formalize the interactive recommendation as a probabilisitic inference problem, and then propose a discrete stochastic actor-critic algorithm to maximize cumulative rewards based on the probabilistic formulation. In order to reduce the extrapolation error, we propose five regularization techniques: support constraints, supervised regularization, policy constraints, dual constraints and reward extrapolation for offline learning, which can constrain the mismatch between the recommendation policy and the unknown logging policy. Our approaches can be viewed as a combination of supervised learning and off-policy reinforcement learning for recommendation with discrete ac-

[*]Corresponding author.

tions. We show that such combination is critical for improving the performance of recommendation in the offline setting. We highlight that we are the first to systemically study the offline learning problem in the interactive recommendation context. Our contributions can be summarized as: (1) We propose a discrete stochastic RL algorithm to maximize cumulative rewards for interactive recommendation. (2) We propose a general offline learning framework for interactive recommendation with logged feedbacks, including support constraints, supervised regularization, policy constraints, dual constraints and reward extrapolation. (3) We conduct extensive offline experiments on two real-world public datasets, empirically demonstrating the proposed methods can achieve superior performance over existing learning methods for recommendation.

2 Related Work

2.1 Offline Learning for Recommendation

A line of work closely related to ours is batch bandit learning (Swaminathan and Joachims 2015b,a; Joachims, Swaminathan, and de Rijke 2018; Sachdeva, Su, and Joachims 2020; Saito et al. 2020), which mainly utilizes IPS to correct the selection bias from the logging policy and can be viewed as a special case of RL with one-step decision making. However, interactive recommendations typically have an effect on user behavior. Thus, we focus on the full RL setting in this paper, where the user state depends on past actions. Recently, Chen et al. (2019a) apply off-policy correction to address selection biases in logged feedbacks. The key problem of this method is that the IPS suffers huge variance which grows exponentially with the horizon (Chen et al. 2019a). An alternative to offline learning in RS is the use of model-based RL algorithms. Model-based methods (Zou et al. 2019, 2020; Chen et al. 2019b; Bai, Guan, and Wang 2019) require training a model using the log data to simulate the user environment and assist the policy learning. However, these methods heavily depend on the accuracy of the trained model and are computationally more complex than model-free methods. More recently, several methods (Wang et al. 2018; Xin et al. 2020; Gong et al. 2019) combine reinforcement learning and/or auxiliary task learning to improve recommendation performance. A simple combination of several distinct losses helps learning from logged feedbacks; however, it is more appealing to have a single principled loss and a general framework that is applicable to offline learning from logged feedbacks. In this work, we focus on providing a general offline learning framework to maximize cumulative rewards for interactive recommendation.

2.2 Interactive Recommendation

To address interactive or sequential recommendation problem, early methods (Rendle, Freudenthaler, and Schmidt-Thieme 2010; Xiao et al. 2019b) utilize Markov Chain to model sequential patterns of users. Since these methods fail to model complicated relations between interactions, a number of Recurrent Neural Network (RNN)-based methods (Hidasi et al. 2016; Xiao, Liang, and Meng 2019a) have been proposed. Apart from RNN, there are also studies that leverage other neural architectures such as convolutional neural networks (CNN) (Tang and Wang 2018; Yuan et al. 2019) and Transformer (Kang and McAuley 2018) to capture sequential patterns. Some attempts formulate the interactive recommendation as Markov Decision Process (MDP) and solve it via deep Q-learning (Zhao et al. 2018b; Zheng et al. 2018; Zhao et al. 2020; Zhou et al. 2020) and deep deterministic policy gradient (DDPG) (Zhao et al. 2018a). These methods typically focus on designing novel neural architectures to capture a particular temporal dynamic of user preferences or extract interactive information from user state. As neural architecture design is not the main focus of our work, please refer to these prior works for more details.

3 Preliminaries

Interactive Recommendation Problem. We study the interactive (sequential) recommendation problem defined as follows: assume we have a set of users $u \in \mathcal{U}$, a set of items $i \in \mathcal{I}$ and for each user we have access to a sequence of user historical events $\mathcal{E} = (i_1, i_2, \cdots)$ ordered by time. Each i_k records the item interacted at time k. Given the historical interactions, our goal is to recommend to each user a subset of items in order to maximize users' long-term satisfaction. **Markov Decision Process.** We translate this sequential recommendation into a MDP $(\mathcal{S}, \mathcal{A}, \mathbf{P}, R, \rho_0, \gamma)$ where $\mathcal{S}$: a continuous state space describing the user states, i.e., $\mathbf{s}_t = (i_1, i_2, \cdots, i_t)$; $\mathcal{A}$: a discrete action space, containing items available for recommendation; $\mathbf{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ is the state transition probability; $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, where $r(\mathbf{s}_t, a_t)$ is the immediate reward obtained by performing action a_t (the item index) at user state $\mathbf{s}_t$. ρ is the initial state distribution; γ is the discount factor for future rewards. RL-based RS learns a target policy $\pi_{\boldsymbol{\theta}}(\cdot \mid \mathbf{s}_t)$ which translates the user state $\mathbf{s} \in \mathcal{S}$ into a distribution over all actions $a \in \mathcal{A}$ to maximize expected cumulative rewards:

$$\max_{\boldsymbol{\theta}} \mathbb{E}_{\tau \sim \pi_{\boldsymbol{\theta}}}[R(\tau)], \text{ where } R(\tau) = \sum_{t=1}^{T} \gamma^t r(\mathbf{s}_t, a_t). \quad (1)$$

Here the expectation is taken over the trajectories $\tau = (\mathbf{s}_1, a_a, \cdots \mathbf{s}_T)$ obtained by acting according to the policy: $\mathbf{s}_0 \sim \rho_0(\mathbf{s}_0), a_t \sim \pi_{\boldsymbol{\theta}}(a_t \mid \mathbf{s}_t)$ and $\mathbf{s}_{t+1} \sim p(\mathbf{s}_{t+1} \mid \mathbf{s}_t, a_t)$. **Offline Learning in Recommendation.** The goal is still to optimize the objective in Eq. 1. However, the recommendation policy no longer has the ability to interact with the user in online environments. Instead, the learning algorithm is provided with a logged dataset of trajectories, and each trajectory $\tau = \{\mathbf{s}_1, a_1, \cdots, \mathbf{s}_T\}$ is drawn from a unknown logging (behavior) policy $\pi_b(a_t|\mathbf{s}_t)$. The dataset can also be represented as the buffer style $\mathcal{D} = \left\{\left(\mathbf{s}_t^i, a_t^i, \mathbf{s}_{t+1}^i\right)\right\}_{i=1}^{N}$. For brevity, we omit superscript i in what follows.

4 Analysis on Existing Baselines

Supervised Learning. The arguably most direct way to learn interactive RS from logged feedbacks is to directly apply sequential supervised learning, which typically relies on the following supervised next-item prediction loss:

$$\mathcal{L}_S(\boldsymbol{\theta}) = -\mathbb{E}_{(\mathbf{s}_t, a_t, \mathbf{s}_{t+1}) \sim \mathcal{D}}[\log \pi_{\boldsymbol{\theta}}(a_t \mid \mathbf{s}_t)], \quad (2)$$

where $\pi_{\boldsymbol{\theta}}\left(a_t \mid \boldsymbol{s}_t\right)=\frac{\exp(f_{\boldsymbol{\theta}}(\mathbf{s}_t)[a_t])}{\sum_{a'}\exp(f_{\boldsymbol{\theta}}(\mathbf{s}_t)[a'])}$ and $f_{\boldsymbol{\theta}}(\cdot)\in\mathbb{R}^{|\mathcal{A}|}$ denotes a neural network such as RNN (Song et al. 2019; Xiao, Liang, and Meng 2019b), CNN (Yuan et al. 2019) or Transformer (Kang and McAuley 2018), which is parameterized by $\boldsymbol{\theta}$ and has an output dimension of the action space $|\mathcal{A}|$. $f_{\boldsymbol{\theta}}(\mathbf{s}_t)[a_t]$ indicates the a_t^{th} index of $f_{\boldsymbol{\theta}}(\mathbf{s}_t)$, i.e., the logit corresponding the a_t^{th} item. The supervised learning methods are training stable and easy to implement. However, they can not maximize cumulative user rewards.

Q-learning. In order to maximize cumulative user rewards, prior works (Zheng et al. 2018; Zhao et al. 2018b) try to build RL-based recommendation algorithms by utilizing the deep Q-learning and minimizing the following loss:

$$\mathcal{L}_Q(\boldsymbol{\theta})=\mathbb{E}_{(\mathbf{s}_t,a_t,\mathbf{s}_{t+1})\sim\mathcal{D}}\left(Q_{\boldsymbol{\theta}}\left(\mathbf{s}_t,a_t\right)-y\right)^2, \qquad (3)$$

where the target value $y=r(\mathbf{s}_t,a_t)+\gamma Q_{\bar{\boldsymbol{\theta}}}\left(\mathbf{s}_{t+1},\pi\left(\mathbf{s}_{t+1}\right)\right)$ and $\pi\left(\mathbf{s}_{t+1}\right)=\arg\max_a Q_{\bar{\boldsymbol{\theta}}}\left(\mathbf{s}_{t+1},a\right)$. The $\bar{\boldsymbol{\theta}}$ indicates parameters of the target network (Sutton and Barto 2018). While the Q-learning have been widely used in the robot learning literature as an off-policy algorithm (Kumar et al. 2019), it is not suitable for offline learning in recommendation task. First, the Q-learning algorithm is affected by extrapolation error (Fujimoto, Meger, and Precup 2019) due to no interaction with environment. Second, in the testing, the robot learning typically chooses one action according the optimal policy: $\arg\max_a Q\left(\mathbf{s},a\right)$. Instead, in recommendation, we generate top-k item lists by sampling from the softmax function $\frac{Q(\mathbf{s},a)}{\sum_{a'}Q(s,a')}$. This violates the simple machine learning principle: testing and training must match (Vinyals et al. 2016). We also find Q-learning generally does not perform well on logged feedbacks for RS in our experiments.

5 Proposed Offline Learning Framework

5.1 Probabilistic Formulation

We start by formally defining the problem of learning interactive RS in the language of probability. The sequential supervised learning can be expressed as a **learning** problem of the probabilistic generative model over observed τ:

$$p(\tau)=\rho(\mathbf{s}_1)\prod_{t=1}^{T}\pi_{\boldsymbol{\theta}}(a_t|\mathbf{s}_t)p(\mathbf{s}_{t+1}|\mathbf{s}_t,a_t), \qquad (4)$$

where $\rho(\mathbf{s}_1)$ and $p(\mathbf{s}_{t+1}\mid\mathbf{s}_t,\mathbf{a}_t)$ are the true initial state distribution and dynamics. The supervised learning discussed in §4 can be seen as maximizing the log-likelihood of this generative model respect to $\boldsymbol{\theta}$. However, like discussed before, this method can not maximize cumulative rewards. Thus, following previous work (Haarnoja et al. 2018a; Levine 2018), we introduce a binary variable $\mathcal{O}$ which are related to rewards by $p(\mathcal{O}_t=1|\mathbf{s}_t,a_t)=\exp(\frac{r(\mathbf{s}_t,a_t)}{\alpha})$, where α is a temperature parameter, to denote whether the action taken at state $\mathbf{s}_t$ is optimal. Note that we assume rewards are nonpositive without loss of generality. Positive rewards can be scaled to be no greater than 0. In the rest of this paper, we use $\mathcal{O}_t$ to represent $\mathcal{O}_t=1$ for brevity. With this definition, we consider the following generative model with the trajectory $\tau=\{\mathbf{s}_1,a_1,\cdots,\mathbf{s}_T\}$ and variables $\mathcal{O}_{1:T}$:

$$p(\tau,\mathcal{O}_{1:T})\propto\rho(\mathbf{s}_1)\prod_{t=1}^{T}p(\mathbf{s}_{t+1}\mid\mathbf{s}_t,a_t)p(\mathcal{O}_t\mid\mathbf{s}_t,a_t). \qquad (5)$$

Note that the action prior is assumed as uniform: $p(a_t)=\frac{1}{|\mathcal{A}|}$ and is omitted. The main difference between this formulation and supervised learning (Eq.(4)) is that this formulation considers the optimal trajectory τ as latent variable and we focus on the **inference** problem: inferring the posterior distribution $P(\tau\mid\mathcal{O}_{1:T})$ given optimality of the start until end of the episode. Based on variational inference (Wainwright and Jordan 2008), we can derive the negative Evidence Lower BOund (ELBO) (Blei, Kucukelbir, and McAuliffe 2017; Xiao et al. 2019a) by introducing variational distribution $q_{\boldsymbol{\theta}}(\tau)=\rho\left(\mathbf{s}_1\right)\prod_{t=1}^{T}p\left(\mathbf{s}_{t+1}\mid\mathbf{s}_t,a_t\right)\pi_{\boldsymbol{\theta}}\left(a_t\mid\mathbf{s}_t\right)$ (derivations are omitted due to space limitation):

$$\mathcal{L}(\boldsymbol{\theta})=\mathbb{E}_{\tau\sim q_{\boldsymbol{\theta}}(\tau)}\left[\sum_{t=1}^{T}\log q_{\boldsymbol{\theta}}(a_t\mid\mathbf{s}_t)-\frac{r\left(\mathbf{s}_t,a_t\right)}{\alpha}\right]. \qquad (6)$$

Minimizing the negative ELBO respect to the $\pi_{\boldsymbol{\theta}}\left(a_t\mid\mathbf{s}_t\right)$ is equal to minimize the Kullback-Leibler (KL) divergence: $\mathrm{KL}(q(\tau)||p(\tau|\mathcal{O}_{1:T}))$. This objective function can be solved by soft Q-learning (Haarnoja et al. 2017) or soft actor-critic algorithms (Haarnoja et al. 2018b). Although the soft actor-critic is the state-of-the-art off-policy algorithm, it can not be directly applied to RS with discrete actions and it still suffers from extrapolation error as discussed in the followings. This probabilistic formulation provides a convenient starting point for our offline learning methods for recommendation.

5.2 Inference via Messages Passing

Recall that our goal is to infer the posterior $p(\tau\mid\mathcal{O}_{1:T})$. Instead of directly minimizing Eq. (6), we consider inferring the posterior through the message passing algorithm (Heskes and Zoeter 2002). We define the backward messages as:

$$\beta_t\left(\mathbf{s}_t,a_t\right):=p\left(\mathcal{O}_{t:T}\mid\mathbf{s}_t,a_t\right) \qquad (7)$$

$$\beta_t\left(\mathbf{s}_t\right):=p\left(\mathcal{O}_{t:T}\mid\mathbf{s}_t\right). \qquad (8)$$

The recursion for backward messages and optimal policy $\pi_{\boldsymbol{\theta}}(a_t\mid\mathbf{s}_t)$ can be obtained by minimizing Eq. (6) and using the Markov property of MDP (derivations are omitted due to space limitation):

$$\beta_t\left(\mathbf{s}_t,a_t\right)=p\left(\mathcal{O}_t\mid\mathbf{s}_t,a_t\right)\mathbb{E}_{p(\mathbf{s}_{t+1}|\cdot)}\left[\beta_{t+1}\left(\mathbf{s}_{t+1}\right)\right] \qquad (9)$$

$$\beta_t\left(\mathbf{s}_t\right)=\mathbb{E}_{a_t\sim p(a_t)}\left[\beta_t\left(\mathbf{s}_t,a_t\right)\right] \qquad (10)$$

$$\pi_{\boldsymbol{\theta}}(a_t\mid\mathbf{s}_t)=p(a_t\mid\mathbf{s}_t,\mathcal{O}_{t:T})=\frac{\beta_t\left(\mathbf{s}_t,a_t\right)}{\beta_t\left(\mathbf{s}_t\right)}\cdot\frac{1}{|\mathcal{A}|}, \qquad (11)$$

where $p\left(\mathbf{s}_{t+1}\mid\cdot\right)=p(\mathbf{s}_{t+1}\mid\mathbf{s}_t,a_t)$ and $p\left(\mathbf{s}_1\mid\cdot\right)=\rho(\mathbf{s}_1)$. We further use the log form of the backward messages to define the Q value function: $Q\left(\mathbf{s}_t,a_t\right)=\alpha\log\beta_t\left(\mathbf{s}_t,a_t\right)$. In order to conduct inference process, we need to approximate the backward message $Q\left(\mathbf{s}_t,a_t\right)$. A direct method is to represent it with parameterized function $Q_{\boldsymbol{\phi}}\left(\mathbf{s}_t,a_t\right)$ with parameter ϕ and optimize the parameter by minimizing the squared error (derivation is omitted due to space limitation):

$$\mathcal{L}_Q(\phi) = \mathbb{E}_{(\mathbf{s}_t,a_t,\mathbf{s}_{t+1})\sim q}\Big[\frac{1}{2}\big(Q_\phi(\mathbf{s}_t,a_t) - r(\mathbf{s}_t,a_t) \quad (12)$$

$$- \big(Q_{\bar{\phi}}(\mathbf{s}_{t+1},a_{t+1}) - \alpha\log(\pi_\theta(a_{t+1}\mid\mathbf{s}_{t+1}))\big)\big)^2\Big].$$

The $\bar{\phi}$ denotes the parameters of target networks as same as vanilla Q-learning (Hasselt, Guez, and Silver 2016). $q = \pi(\mathbf{s}_t)\pi_\theta(a_t\mid\mathbf{s}_t)p(\mathbf{s}_{t+1}\mid\mathbf{s}_t,a_t)$ denotes the variational state-action-state marginal distribution encountered when executing a policy $\pi_\theta(a_t\mid\mathbf{s}_t)$. Given the estimated value functions, we can estimate optimal policy according to Eq. (11). Since our policy is discrete, one can directly utilizes the value functions to obtain the non-parametric policy without optimizing θ: let $\pi(a_t\mid\mathbf{s}_t) = \frac{\exp(Q_\phi(\mathbf{s}_t,a_t))}{\sum_{a'}\exp\left(Q_\phi(\mathbf{s}_t,a')\frac{1}{\alpha}\right)}$. However, $Q_\phi(\mathbf{s}_t,a_t)$ is typically a very large network in recommendation, e.g. deep ranking model (Covington, Adams, and Sargin 2016), for alleviating model misspecification, thus it is not computationally efficient if we use this non-parametric policy. Instead, we consider optimizing a small parametric policy $\pi_\theta(a_t\mid\mathbf{s}_t)$ by minimizing the KL divergence $\mathrm{KL}\big(\pi_\theta(a_t\mid\mathbf{s}_t)\,||\,\frac{\exp(Q_\phi(\mathbf{s}_t,a_t))}{\sum_{a'}\exp\left(Q_\phi(\mathbf{s}_t,a')\frac{1}{\alpha}\right)}$, which can be rewritten as the following loss:

$$\mathcal{L}_\pi(\theta) = \mathbb{E}_{a_t\sim\pi_\theta}\left[\log\pi_\theta(a_t\mid\mathbf{s}_t) - \frac{Q_\phi(\mathbf{s}_t,a_t)}{\alpha}\right], \quad (13)$$

where π_θ is the short for $\pi_\theta(a_t\mid\mathbf{s}_t)$. This idea is also similar to knowledge distillation (Hinton, Vinyals, and Dean 2015): we have a large critic (value) network being somewhat "distilled" into a smaller actor (policy) network. Since $\pi_\theta(a_t\mid\mathbf{s}_t) = \frac{\exp(f_\theta(\mathbf{s}_t)[a_t])}{\sum_{a'}\exp(f_\theta(\mathbf{s}_t)[a'])}$ is the categorical distribution in our case, a typical solution for this loss is marginalizing out policy over all actions (Christodoulou 2019), however, this simple solution is expensive for a large numbers of items in recommendation. Thus, we address this by utilizing the Gumbel-Softmax (Jang, Gu, and Poole 2016; Meng et al. 2019), which provides a continuous differentiable approximation by drawing samples y from a categorical distribution with class probabilities $f_\theta(\mathbf{s}_t)$:

$$y_i = \frac{\exp\left(\left(\log\left(f_\theta(\mathbf{s}_t)[a_i]\right) + g_i\right)/\gamma_g\right)}{\sum_{i=1}^{|\mathcal{A}|}\exp\left(\left(\log\left(f_\theta(\mathbf{s}_t)[a_i]\right) + g_i\right)/\gamma_g\right)}, \quad (14)$$

where $\{g_i\}_{i=1}^{|\mathcal{A}|}$ are i.i.d. samples drawn from the Gumbel (0, 1) distribution, γ_g is the softmax temperature and y_i is the i-th value of sample $\mathbf{y}$. Compared to the Q-learning, our optimal policy $\pi_\theta(a_t\mid\mathbf{s}_t)$ has the discrete softmax form not the argmax, which can avoid the mismatch between training and testing. We can iteratively train value function (policy evaluation) Eq. (12) and estimate discrete optimal policy (policy improvement) Eq. (13) via stochastic gradient descent.

5.3 Stochastic Discrete Actor Critic

So far, we have proposed a discrete RL algorithm for interactive recommendation. However, we notice that we can not directly apply this algorithm to our offline learning case, since the policy evaluation loss (Eq. (12)) can be difficult to optimize due to the dependency between $\pi(\mathbf{s}_t)$ and $\pi_\theta(a_t\mid\mathbf{s}_t)$, as well as the need to collect samples from $\pi_\theta(a_t\mid\mathbf{s}_t)$ (on-policy learning). Thus we consider optimizing an surrogate approximation $\hat{\mathcal{L}}_Q(\phi)$ of $\mathcal{L}_Q(\phi)$ using the state and action distributions of the unknown logging policy, i.e., $\pi_b(\mathbf{s}_t)$ and $\pi_b(a_t\mid\mathbf{s}_t)$. In other words, we can use the logged feedbacks to conduct the policy evaluation step:

$$\hat{\mathcal{L}}_Q(\phi) = \mathbb{E}_{(\mathbf{s}_t,a_t,\mathbf{s}_{t+1})\sim\mathcal{D}}\Big[\frac{1}{2}\big(Q_\phi(\mathbf{s}_t,a_t) - r(\mathbf{s}_t,a_t) \quad (15)$$

$$- \big(Q_{\bar{\phi}}(\mathbf{s}_{t+1},a_{t+1}) - \alpha\log(\pi_\theta(a_{t+1}\mid\mathbf{s}_{t+1}))\big)\big)^2\Big].$$

$\hat{\mathcal{L}}_Q(\phi)$ matches $\mathcal{L}_Q(\phi)$ to first order approximation (Kakade and Langford 2002), and provides a reasonable estimate of $\mathcal{L}_Q(\phi)$ if $\pi_b(a_t\mid\mathbf{s}_t)$ and $\pi_\theta(a_t\mid\mathbf{s}_t)$ are similar. However, we find it performs dramatically worse than supervised learning methods on logged implicit feedbacks in our recommendation task due to the extrapolation error (Fujimoto, Meger, and Precup 2019). As discussed in the prior work (Fujimoto, Meger, and Precup 2019), extrapolation error can be attributed to a mismatch in the distribution of logged feedbacks induced by the policy $\pi_\theta(a_t\mid\mathbf{s}_t)$ and the distribution $\pi_b(a_t\mid\mathbf{s}_t)$ of data contained in the batch in the policy iteration process. It is therefore crucial to control divergence of the optimizing policy $\pi_\theta(a_t\mid\mathbf{s}_t)$ and the unknown logging policy $\pi_b(a_t\mid\mathbf{s}_t)$. To this effect, we explore five approaches for our discrete stochastic actor-critic algorithm in the followings.

5.4 Support Constraints

Previous work (Fujimoto, Meger, and Precup 2019) extends Q-learning and tries to disallow any action that has zero support under the logging policy, which can be formally defined as the δ-behavior constrained policy:

$$\mathbb{1}\left[a_t = \arg\max_a\left\{\hat{Q}(\mathbf{s}_t,a) : \pi_b(a\mid\mathbf{s}_t)\geq\delta\right\}\right], \quad (16)$$

where $0\leq\delta\leq 1$. This constrains the deterministic argmax greedy policy in vanilla Q-learning to actions that the logging policy chooses with probability at least δ. However, this constraint is not suitable for our proposed stochastic softmax policy. Thus, we combine the policy improvement loss $\mathcal{L}_\pi(\theta)$ (Eq. (13)) and the support constraint as a joint loss: $\mathcal{L}_\pi^{\text{sc}}(\theta) = \mathcal{L}_\pi(\theta) + \beta\mathcal{R}_{\text{sc}}(\theta)$, where β is the coefficient and the proposed support constraint term $\mathcal{R}_{sc}(\theta)$ is:

$$\mathcal{R}_{\text{sc}}(\theta) = -\sum_{a\in\mathcal{A},\pi_b(a\mid\mathbf{s}_t)\leq\delta}\pi_\theta(a\mid\mathbf{s}_t). \quad (17)$$

This method is very simple and conceptually straightforward but requires estimating the unknown logging policy when $\delta > 0$. In addition, it is just one-step constraint of policy at time step t, meaning that it can not avoid actions that may lead to higher deviation at future time steps.

5.5 Supervised Regularization

In order to avoid estimating the logging policy, we propose a auxiliary supervised regularization to reduce the extrapolation error by minimizing the forward KL divergence, i.e.,

$\mathrm{KL}\big(\pi_b(a_t \mid \mathbf{s}_t) || \pi_{\boldsymbol{\theta}}(a_t \mid \mathbf{s}_t)\big)$ between the unknown logging policy $\pi_b(a_t \mid \mathbf{s}_t)$ and recommendation policy $\pi_{\boldsymbol{\theta}}(a_t \mid \mathbf{s}_t)$:

$$\mathcal{R}_{\mathrm{sr}}(\boldsymbol{\theta}) = -\mathbb{E}_{(a_t,\mathbf{s}_t)\sim\pi_b(a_t,\mathbf{s}_t)}\left[\log \pi_{\boldsymbol{\theta}}\left(a_t \mid \boldsymbol{s}_t\right)\right]. \quad (18)$$

This method does no require a pre-estimated logging policy since we can directly use the logged data drawn from $\pi_b(a_t \mid \mathbf{s}_t)$. Similar to support constraints, we need combine the policy improvement loss and the supervised loss: $\mathcal{L}_{\pi}^{\mathrm{sr}}(\boldsymbol{\theta}) = \mathcal{L}_{\pi}(\boldsymbol{\theta}) + \beta\mathcal{R}_{\mathrm{sr}}(\boldsymbol{\theta})$. However, the two terms are on an arbitrary relative scale, which can make it difficult to choose β. To adaptively update β, we replace the soft supervised regularization with a hard constraint with parameter ϵ. The optimization loss for the policy improvement becomes:

$$\mathcal{L}_{\pi}^{\mathrm{sr}}(\boldsymbol{\theta}) = \min_{\boldsymbol{\theta}} \mathbb{E}_{a_t\sim\pi_{\boldsymbol{\theta}}}\left[\log \pi_{\boldsymbol{\theta}}\left(a_t \mid \mathbf{s}_t\right) - \frac{Q_{\boldsymbol{\phi}}\left(\mathbf{s}_t, a_t\right)}{\alpha}\right],$$
$$\text{s. t.} \quad \mathbb{E}_{\mathbf{s}_t\sim\pi_b(\mathbf{s}_t)}\left[\mathrm{KL}\left[\pi_b(a_t \mid \mathbf{s}_t) \| \pi_{\boldsymbol{\theta}}(a_t \mid \mathbf{s}_t)\right]\right] \le \epsilon. \quad (19)$$

To optimize this, we introduce a Lagrange multiplier β via dual gradient descent and Gumbel-Softmax (derivations are omitted due to space limitation):

$$\mathcal{L}_{\pi}^{\mathrm{sr}}(\boldsymbol{\theta}, \beta) = \min_{\boldsymbol{\theta}} \max_{\beta\ge 0} \mathbb{E}_{a_t\sim\pi_{\boldsymbol{\theta}}}\big[\alpha \log \pi_{\boldsymbol{\theta}}\big(a_t \mid \mathbf{s}_t\big) - \quad (20)$$
$$Q_{\boldsymbol{\phi}}\big(\mathbf{s}_t, a_t\big)\big] - \beta\left(\mathbb{E}_{\mathbf{s}_t\sim\pi_b}[\mathrm{KL}[\pi_b(a_t \mid \mathbf{s}_t) \| \pi_{\boldsymbol{\theta}}(a_t \mid \mathbf{s}_t)]] - \epsilon\right).$$

As discussed in §5.3 and demonstrated in our experiments, enforcing a specific constraint between learning policy and logging policy is critical for good performance.

5.6 Policy Constraints

Since the KL divergence is asymmetric, we can also consider constraining the policy via the reverse KL divergence $\mathrm{KL}\big(\pi_{\boldsymbol{\theta}}(a_t \mid \mathbf{s}_t) \| \pi_b(a_t \mid \mathbf{s}_t)\big)$ during our policy improvement process. To see this, we add the reverse KL constraint to Eq. (13) and remove the negative entropy term $\pi_{\boldsymbol{\theta}}(a_t \mid \mathbf{s}_t)\log \pi_{\boldsymbol{\theta}}(a_t \mid \mathbf{s}_t)$ in Eq. (13) since there is already a entropy term in the reverse KL divergence. Then, the objective for the policy improvement with policy constraint is:

$$\mathcal{L}_{\pi}^{\mathrm{pc}}(\boldsymbol{\theta}) = \min_{\boldsymbol{\theta}} \mathbb{E}_{a_t\sim\pi_{\boldsymbol{\theta}}}\left[-Q_{\boldsymbol{\phi}}\left(\mathbf{s}_t, a_t\right)\right], \quad (21)$$
$$\text{s. t.} \quad \mathbb{E}_{\mathbf{s}_t\sim\pi_b(\mathbf{s}_t)}\left[\mathrm{KL}\left[\pi_{\boldsymbol{\theta}}(a_t \mid \mathbf{s}_t) \| \pi_b(a_t \mid \mathbf{s}_t)\right]\right] \le \epsilon.$$

Similar to Supervised Regularization, optimizing this constrained problem via introducing Lagrange multiplier β results the final policy improvement objective:

$$\mathcal{L}_{\pi}^{\mathrm{pc}}(\boldsymbol{\theta}) = \min_{\boldsymbol{\theta}} \mathbb{E}_{(a_t,\mathbf{s}_t)\sim\pi_b(a_t,\mathbf{s}_t)}\left[w_t \log \pi_{\theta}(a_t \mid \mathbf{s}_t)\right], \quad (22)$$

where $w_t = -\exp\left(\frac{Q_{\boldsymbol{\phi}}(\mathbf{s}_t,a_t)}{\beta}\right)$. This objective simplifies the policy improvement problem to a weighted maximum likelihood problem, and does not require an estimated logging policy and Gumbel-Softmax approximation.

5.7 Dual Constraints

Although the methods proposed in previous subsections can effectively control the divergence of optimizing policy and data logging policy, both of them only consider one-step regularization, and thus can not avoid actions that may lead to higher deviation at future time steps. To address this problem, instead of explicitly adding constraint for policy improvement at time t, we consider the logging policy as a prior and incorporating it directly into our original probabilistic model (Eq. 5). Specifically, we first estimate the logging policy $\hat{\pi}_b(a_t \mid \mathbf{s}_t)$ via supervised learning using the logged feedbacks as the proxy of unknown logging $\pi_b(a_t \mid \mathbf{s}_t)$, and then incorporate it as the action prior into Eq. (5), which yields the following joint distribution:

$$p = \rho(\mathbf{s}_1)\prod_{t=1}^{T} p(\mathbf{s}_{t+1} \mid \mathbf{s}_t, \mathbf{a}_t)p(\mathcal{O}_t \mid \mathbf{s}_t, \mathbf{a}_t)\hat{\pi}_b(a_t|\mathbf{s}_t), \quad (23)$$

where p denotes the joint distribution $p(\tau, \mathcal{O}_{1:T})$. Given this joint distribution, similar to §5.2, we can infer the posterior $p(\tau \mid \mathcal{O}_{1:T})$ by introducing the variational distribution $q_{\boldsymbol{\theta}}(\tau) = \rho(\mathbf{s}_1)\prod_{t=1}^{T} p\left(\mathbf{s}_{t+1} \mid \mathbf{s}_t, a_t\right)\pi_{\boldsymbol{\theta}}\left(a_t \mid \mathbf{s}_t\right)$. We also consider inferring the posterior through the message passing algorithm as same as we do in §5.2. However, since the action prior is no longer uniform, we have two choices of definitions of the backward messages. The first is defined as same as Eq. (7) and Eq. (8), i.e, $\beta_t\left(\mathbf{s}_t, a_t\right) := p\left(\mathcal{O}_{t:T} \mid \mathbf{s}_t, a_t\right)$ and $\beta_t\left(\mathbf{s}_t\right) := p\left(\mathcal{O}_{t:T} \mid \mathbf{s}_t\right)$. Based on this definition and through minimizing $\mathrm{KL}\left(q_{\boldsymbol{\theta}}\left(\tau\right) \| p\left(\tau \mid \mathcal{O}_{1:T}\right)\right)$, we can obtain following objectives for policy evaluation and policy improvement:

$$\mathcal{L}_{Q}^{\mathrm{dc}}(\boldsymbol{\phi}) = \mathbb{E}_{(\mathbf{s}_t,a_t,\mathbf{s}_{t+1})\sim\mathcal{D}}\Big[\frac{1}{2}\big(Q_{\boldsymbol{\phi}}\big(\mathbf{s}_t, a_t\big) - r(\mathbf{s}_t, a_t) \quad (24)$$
$$-\big(Q_{\bar{\boldsymbol{\phi}}}\left(\mathbf{s}_{t+1}, a_{t+1}\right) + \alpha \log \hat{\pi}_b(a_{t+1} \mid \mathbf{s}_{t+1}) \quad (25)$$
$$-\alpha \log\left(\pi_{\boldsymbol{\theta}}\left(a_{t+1} \mid \mathbf{s}_{t+1}\right)\right)\big)\big)^2\Big], \quad (26)$$
$$\mathcal{L}_{\pi}^{\mathrm{dc}}(\boldsymbol{\theta}) = \mathbb{E}_{a_t\sim\pi_{\boldsymbol{\theta}}}\Big[\log \frac{\pi_{\boldsymbol{\theta}}\left(a_t \mid \mathbf{s}_t\right)}{\hat{\pi}_b(a_t \mid \mathbf{s}_t)} - \frac{Q_{\boldsymbol{\phi}}\left(\mathbf{s}_t, a_t\right)}{\alpha}\Big]. \quad (27)$$

Compared to the supervised regularization in §5.5 and policy constraint in §5.6, this method not only adds prior constraint on the policy improvement step, but also adds it to the target Q value, which can avoid actions that are far from logging policy at future time steps.

5.8 Reward Extrapolation

Following the dual constraints in §5.7, if we consider the second definition of the backward messages: $\beta_t\left(\mathbf{s}_t, a_t\right) := p\left(\mathcal{O}_{t:T} \mid \mathbf{s}_t, a_t\right)\hat{\pi}_b(a_t \mid \mathbf{s}_t)$ and $\beta_t\left(\mathbf{s}_t\right) := p\left(\mathcal{O}_{t:T} \mid \mathbf{s}_t\right)$, we can obtain following objectives for policy iteration:

$$\mathcal{L}_{Q}^{\mathrm{re}}(\boldsymbol{\phi}) = \mathbb{E}_{(\mathbf{s}_t,a_t,\mathbf{s}_{t+1})\sim\mathcal{D}}\Big[\frac{1}{2}\big(Q_{\boldsymbol{\phi}}\big(\mathbf{s}_t, a_t\big) - \hat{r}(\mathbf{s}_t, a_t) \quad (28)$$
$$-\big(Q_{\bar{\boldsymbol{\phi}}}\left(\mathbf{s}_{t+1}, a_{t+1}\right) - \alpha \log\left(\pi_{\boldsymbol{\theta}}\left(a_{t+1} \mid \mathbf{s}_{t+1}\right)\right)\big)\big)^2\Big], \quad (29)$$
$$\mathcal{L}_{\pi}^{\mathrm{re}}(\boldsymbol{\theta}) = \mathbb{E}_{a_t\sim\pi_{\boldsymbol{\theta}}}\Big[\log \pi_{\boldsymbol{\theta}}\left(a_t \mid \mathbf{s}_t\right) - \frac{Q_{\boldsymbol{\phi}}\left(\mathbf{s}_t, a_t\right)}{\alpha}\Big], \quad (30)$$

where $\hat{r}(\mathbf{s}_t, a_t) = r(\mathbf{s}_t, a_t) + \alpha \log \hat{\pi}_b(a_t \mid \mathbf{s}_t)$. We can observe that this method extrapolates the task specific rewards with the output of the estimated logging policy $\hat{\pi}_b(a_t \mid \mathbf{s}_t)$. This modified objective forces the model to learn that the most valuable actions, but still have high probability in the

	RecSys				Kaggle			
	H@5	N@5	H@10	N@10	H@5	N@5	H@10	N@10
SL	.2876	.1982	.3793	.2279	.2233	.1735	.2673	.1878
DQN	.2134	.1215	.3125	.1673	.1471	.0953	.1965	.1176
PG	.2151	.1279	.3218	.1792	.1585	.1041	.2083	.1212
SL+DQN	.2991	.2012	.3951	.2348	.2487	.1939	.2967	.2094
SL+PG	.3012	.2106	.4013	.2382	.2504	.1972	.3036	.2118
SDAC	.2341	.1332	.3316	.1872	.1669	.1162	.2173	.1358
SC	.2987	.1991	.3905	.2356	.2352	.1885	.2854	.1962
SR	.3197	.2234	.4184	.2515	.2586	.2087	.3153	.2259
PC	.3081	.1986	.3903	.2319	.2354	.1913	.2941	.1958
DC	**.3272***	**.2306***	**.4217***	**.2593***	**.2659***	**.2181***	**.3204***	**.2351***
RE	.3128	.2195	.4071	.2416	.2528	.2043	.3085	.2192

Table 1: Performance comparison of difference learning algorithms utilizing RNN as the backbone. The best and the second best performance are marked with boldfaces and underlined, respectively. $*$ indicates the method outperforms others at a significance level of $p \leq 0.01$ by paired t-tests.

	RecSys				Kaggle			
	H@5	N@5	H@10	N@10	H@5	N@5	H@10	N@10
SL	.2728	.1896	.3593	.2177	.1966	.1566	.2302	.1675
DQN	.1946	.1075	.3004	.1562	.1132	.0621	.1323	.0958
PG	.2031	.1191	.3079	.1616	.1212	.0696	.1428	.1024
SL+DQN	.2742	.1909	.3613	.2192	.2089	.1611	.2454	.1778
SL+PG	.2776	.1977	.3678	.2215	.2107	.1747	.2504	.1804
SDAC	.2138	.1207	.3109	.1643	.1342	.0927	.1566	.1212
SC	.2665	.1913	.3702	.2202	.2007	.1576	.2369	.1772
SR	.2841	.2076	.3741	.2311	.2241	.1798	.2538	.1895
PC	.2782	.1918	.3692	.2214	.2132	.1752	.2498	.1831
DC	**.2951***	**.2136***	**.3853***	**.2413***	**.2341***	**.1853***	**.2745***	**.1920***
RE	.2866	.2101	.3847	.2371	.2289	.1801	.2673	.1866

Table 2: Performance comparison of difference learning algorithms utilizing CNN as the backbone. The best and the second best performance are marked with boldfaces and underlined, respectively.

original logged feedbacks. Note that while inference objectives for dual constraints and reward extrapolation are same, the particular choice of how the backward messages are defined can make a significant difference in practice due to the difference between actor and critic in network architectures.

6 Experiments

In this section, we empirically analyze and compare the effectiveness of the proposed approaches. We conduct experiments on two public real-world datasets and investigate the following research questions: **(RQ1)** How do the proposed methods perform compared with existing methods for interactive recommendation? **(RQ2)** Are the proposed learning methods robust to different types of neural architectures and sparse logged feedbacks? **(RQ3)** Can the adaptive update step improve performance of the supervised regularization? **(RQ4)** How sensitive is the performance of the proposed learning methods with respect to the trade-off parameters?

6.1 Experimental Setup

We use following two public real-world datasets:
RecSys [1]: This dataset is a public dataset released by RecSys Challenge 2015 and contains sequences of user purchases and clicks. After preprocessing, it contains 200,000 session sequence and 1,110,965 interactions over 26,702 items.
Kaggle [2]: This dataset comes from a real-world e-commerce website. After preprocessing, it contains 195,523 sequence and 1,176,680 interactions over 70,852 items. **Evaluation Metrics**. For offline evaluation, we employ top-k Hit Ratio (HR@k) and Normalized Discounted Cumulative Gain (NDCG@k) to evaluate the performance, which are widely used in related works (Chen, Wang, and Yin 2021; Chen, Xu, and Wang 2021; Chen, Gai, and Wang 2019; Xiao and Shen 2019). We report results on HR (H)@{5, 10} and NDCG (N)@{5, 10}. We adopt cross-validation to evaluate the performance of the proposed methods. we randomly sample 80% sequences as the training set, 10% as validation and the rest as test set. We use all items as candidates and rank them for evaluation. Each experiment is repeated 5 times, and the average performance is reported. **Baselines**. Since this paper focuses on proposing learning algorithms. We consider following learning algorithms as our baselines: Supervised Learning (SL), Deep Q-Learning (DQN), Off-Policy Gradient (PG) (Chen et al. 2019a), (SL+DQN) (Xin et al. 2020), (SL+PG) (Gong et al. 2019). We compare these baselines with our proposed stochastic discrete actor critic (SDAC), support constraints (SC), supervised regularization (SR), policy constraints (PC), dual constraints (DC) and reward extrapolation (RE). All learning methods are based on the same backbone i.e., recurrent neural networks (RNN) introduced by work (Chen et al. 2019a), in order to avoid the choice of backbone to be a confounding factor. Hyperparameters are tuned on validation set.

6.2 Experimental Results

Overall Performance (RQ1). Table 1 shows the performance of our proposed methods and the baselines. From this table, we have the following observations: (a) The off-policy RL algorithms (DQN and PG) perform dramatically worse than SL, demonstrating that off-policy RL algorithms can not effectively learn a optimal policy without online interactions due to the extrapolation error. (b) Our SDAC outperform DQN, though both of them are off-policy RL algorithms. One possible reason is that our SDAC learns a stochastic discrete policy, which make it more suitable for recommendation task with discrete items compared to DQN. (c) The proposed methods with policy constraints or regularization, e.g., SR, RE and DC significantly outperform the proposed SDAC, which demonstrates that minimizing the mismatch between recommendation policy and logging policy is important when training an off-policy RL algorithm in the offline setting. (d) Our offline learning methods RE and DC outperform the SL, indicting that exploiting both supervision and task reward, and maximizing cumulative rewards does help improve the performance of recommendation. **Different backbones and sparse feedbacks (RQ2)**. In the **RQ1**, we implement all methods based on the RNN backbone. To further evaluate the effectiveness of the pro-

[1]https://recsys.acm.org/recsys15/challenge/

[2]https://www.kaggle.com/retailrocket/ecommerce-dataset

posed offline learning methods, we consider the case where the learning methods is implemented using other neural network architectures. We consider other two state-of-the-art neural architectures in recommendation as backbones, i.e., temporal CNN (Caser) (Tang and Wang 2018) and Transformer (SASRec) (Kang and McAuley 2018). We only show the results on CNN due to space limitations, and the results of Transformer are similar to CNN and are thus omitted. As shown in Table 2, our offline learning methods perform better than all the compared methods in most cases, which once again proves the effectiveness of our methods and also shows that our offline learning methods is robust to different backbones. To evaluate the effectiveness of proposed offline learning methods on more sparse logged feedbacks, we consider the purchase, a more sparse feedback compared to click. RecSys dataset contains 43,946 purchases of users. For Kaggle dataset, we consider the behavior of adding to cart as the purchase feedback, resulting 57,269 purchase feedbacks. Table 3 shows the purchase performance comparison of different learning algorithms. According to the table: (1) Our offline learning methods such as DC, SR and RE still dominate other baselines, which confirms that our methods can also perform well on sparse logged feedbacks. (2) Different from the Table 1, RE becomes a competitive method and outperforms SR, which demonstrates that control the divergence of recommendation policy and logging policy at future time steps is helpful on sparse feedbacks.
Adaptive Regularization (RQ3) To evaluate the effects of the adaptive β updates in the SR method (see §5.5), we compare policies trained with different fixed values of β and policies where β is updated adaptively to enforce a desired distribution constraint $\epsilon = 1$. Figure 1 shows the performance comparison with different β. We can find that policies trained using dual gradient descent to adaptively update β consistently achieves the best performance overall.
Sensitivity Analysis (RQ4). We study how trade-off hyper-parameters in proposed offline learning methods affect the performance. For SC, we have trade-off hyper-parameter δ, as shown in Eq. 17, SC becomes SDAC when $\delta = 0$ or $\delta = 1$. For PC, if the parameter $\beta \to \infty$, then PC resembles the supervised learning. For DC and RA, the choice of α also creates a trade-off between supervised learning and our reinforcement learning algorithm. Figure 2 illustrates the NDCG@10 of SC, PC, DC and RE with different hyper-parameters. The effect of β in SR has been studied in RQ3, and is omitted here. Figure 2 shows that we can balance the supervised signal and task reward by varying these parameters, leading to better performance under the offline setting.

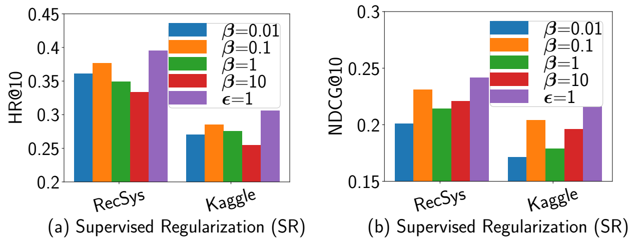

Figure 1: Comparison of supervised regularization on two datasets with different β utilizing RNN as backbone.

	RecSys				Kaggle			
	H@5	N@5	H@10	N@10	H@5	N@5	H@10	N@10
SL	.3994	.2824	.5183	.3204	.4608	.3834	.5107	.3995
DQN	.3478	.2417	.4820	.2843	.4087	.3218	.4524	.3401
PG	.3514	.2576	.4883	.2941	.4172	.3324	.4612	.3517
SL+DQN	.4228	.3016	.5333	.3376	.5069	.4130	.5589	.4289
SL+PG	.4325	.3071	.5412	.3414	.5087	.4172	.5602	.4340
SDAC	.3671	.2624	.4917	.3012	.4236	.3476	.4721	.3632
SC	.4216	.2978	.5279	.3351	.4982	.4052	.5517	.4149
SR	.4341	.3086	.5458	.3516	.5111	.4239	.5641	.4418
PC	.4356	.3074	.5401	.3396	.5105	.4155	.5627	.4340
DC	**.4427***	**.3219***	**.5571***	**.3587***	**.5341***	**.4339***	**.5868***	**.4687***
RE	.4372	.3102	.5487	.3527	.5201	.4278	.5743	.4547

Table 3: Purchase performance comparison utilizing RNN as the backbone. The best and the second best performance are marked with boldfaces and underlined, respectively.

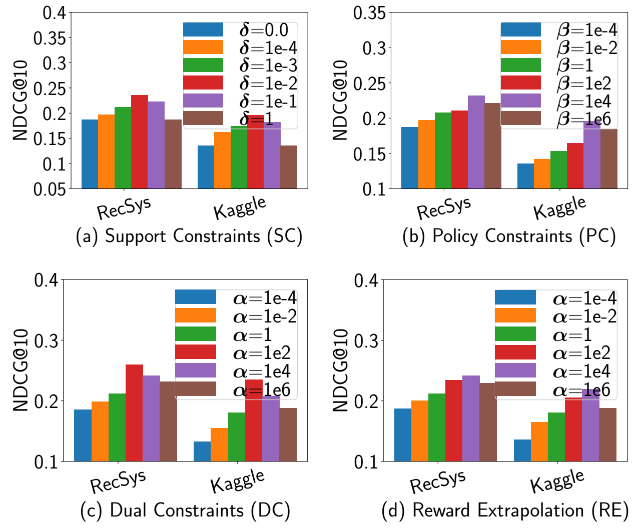

Figure 2: Performance comparison of NDCG when varying the trade-off parameters in four methods on two datasets.

7 Conclusions

In this paper, we presented the first comprehensive analysis of learning interactive recommendation offline. We first formalized the interactive recommendation as a probabilistic inference problem, and then proposed a stochastic and discrete RL algorithm to maximize user cumulative rewards. To perform offline learning effectively, we proposed a general offline learning framework to minimize the distribution mismatch between the logging policy and learning policy, including support constraints, supervised regularization, policy constraints, dual constraints and reward extrapolation. We conducted extensive experiments on two real-world datasets, demonstrating that the proposed methods can achieve better performance over existing methods.

Acknowledgments

The authors would like to thank the Westlake University and Bright Dream Robotics Joint Institute for the funding support. The authors also would like to thank the anonymous reviewers for thoughtful and constructive feedbacks.

References

Bai, X.; Guan, J.; and Wang, H. 2019. A Model-Based Reinforcement Learning with Adversarial Training for Online Recommendation. In *Advances in Neural Information Processing Systems*, 10734–10745.

Blei, D. M.; Kucukelbir, A.; and McAuliffe, J. D. 2017. Variational inference: A review for statisticians. *Journal of the American statistical Association* 112(518): 859–877.

Chen, M.; Beutel, A.; Covington, P.; Jain, S.; Belletti, F.; and Chi, E. H. 2019a. Top-K Off-Policy Correction for a REINFORCE Recommender System. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, 456–464.

Chen, X.; Li, S.; Li, H.; Jiang, S.; Qi, Y.; and Song, L. 2019b. Generative Adversarial User Model for Reinforcement Learning Based Recommendation System. In *Proceedings of the 36th International Conference on Machine Learning*, 1052–1061.

Chen, Z.; Gai, S.; and Wang, D. 2019. Deep Tensor Factorization for Multi-Criteria Recommender Systems. In *2019 IEEE International Conference on Big Data*, 1046–1051.

Chen, Z.; Wang, D.; and Yin, S. 2021. Improving Cold-Start Recommendation via Multi-Prior Meta-Learning. In *43rd European Conference on IR Research*.

Chen, Z.; Xu, Z.; and Wang, D. 2021. Deep Transfer Tensor Decomposition with Orthogonal Constraint for Recommender Systems. In *The Thirty-Fifth AAAI Conference on Artificial Intelligence*.

Christodoulou, P. 2019. Soft actor-critic for discrete action settings. *arXiv preprint arXiv:1910.07207* .

Covington, P.; Adams, J.; and Sargin, E. 2016. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM conference on recommender systems*, 191–198.

Fujimoto, S.; Meger, D.; and Precup, D. 2019. Off-policy deep reinforcement learning without exploration. In *International Conference on Machine Learning*, 2052–2062.

Gong, Y.; Zhu, Y.; Duan, L.; Liu, Q.; Guan, Z.; Sun, F.; Ou, W.; and Zhu, K. Q. 2019. Exact-K Recommendation via Maximal Clique Optimization. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 617–626.

Haarnoja, T.; Hartikainen, K.; Abbeel, P.; and Levine, S. 2018a. Latent space policies for hierarchical reinforcement learning. *arXiv preprint arXiv:1804.02808* .

Haarnoja, T.; Tang, H.; Abbeel, P.; and Levine, S. 2017. Reinforcement learning with deep energy-based policies. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, 1352–1361.

Haarnoja, T.; Zhou, A.; Abbeel, P.; and Levine, S. 2018b. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. In *International Conference on Machine Learning*, 1861–1870.

Hasselt, H. v.; Guez, A.; and Silver, D. 2016. Deep reinforcement learning with double Q-Learning. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, 2094–2100.

Heskes, T.; and Zoeter, O. 2002. Expectation propagation for approximate inference in dynamic bayesian networks. In *Proceedings of the Eighteenth conference on Uncertainty in artificial intelligence*, 216–223.

Hidasi, B.; Karatzoglou, A.; Baltrunas, L.; and Tikk, D. 2016. Session-based recommendations with recurrent neural networks. In *4th International Conference on Learning Representations*.

Hinton, G.; Vinyals, O.; and Dean, J. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531* .

Jang, E.; Gu, S.; and Poole, B. 2016. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144* .

Joachims, T.; Swaminathan, A.; and de Rijke, M. 2018. Deep Learning with Logged Bandit Feedback. In *6th International Conference on Learning Representations*.

Kakade, S. 2003. On the Sample Complexity of Reinforcement Learning. *PhD thesis, Gatsby Computational Neuroscience Unit, University College London* .

Kakade, S.; and Langford, J. 2002. Approximately Optimal Approximate Reinforcement Learning. In *Proceedings of the Nineteenth International Conference on Machine Learning*, 267–274.

Kang, W.-C.; and McAuley, J. 2018. Self-attentive sequential recommendation. In *2018 IEEE International Conference on Data Mining (ICDM)*, 197–206. IEEE.

Kumar, A.; Fu, J.; Soh, M.; Tucker, G.; and Levine, S. 2019. Stabilizing off-policy q-learning via bootstrapping error reduction. In *Advances in Neural Information Processing Systems*, 11784–11794.

Levine, S. 2018. Reinforcement learning and control as probabilistic inference: Tutorial and review. *arXiv preprint arXiv:1805.00909* .

Liu, Y.; Swaminathan, A.; Agarwal, A.; and Brunskill, E. 2019. Off-Policy Policy Gradient with Stationary Distribution Correction. In *Proceedings of the Thirty-Fifth Conference on Uncertainty in Artificial Intelligence*, 440.

Meng, Z.; Liang, S.; Fang, J.; and Xiao, T. 2019. Semi-supervisedly co-embedding attributed networks. In *Advances in Neural Information Processing Systems*, 6507–6516.

Rendle, S.; Freudenthaler, C.; and Schmidt-Thieme, L. 2010. Factorizing personalized markov chains for next-basket recommendation. In *Proceedings of the 19th international conference on World wide web*, 811–820.

Sachdeva, N.; Su, Y.; and Joachims, T. 2020. Off-policy Bandits with Deficient Support. In *The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 965–975.

Saito, Y.; Yaginuma, S.; Nishino, Y.; Sakata, H.; and Nakata, K. 2020. Unbiased Recommender Learning from Missing-Not-At-Random Implicit Feedback. In *Proceedings of the 13th International Conference on Web Search and Data Mining*, 501–509.

Siegel, N.; Springenberg, J. T.; Berkenkamp, F.; Abdolmaleki, A.; Neunert, M.; Lampe, T.; Hafner, R.; Heess, N.; and Riedmiller, M. 2019. Keep Doing What Worked: Behavior Modelling Priors for Offline Reinforcement Learning. In *International Conference on Learning Representations*.

Song, J.; Shen, H.; Ou, Z.; Zhang, J.; Xiao, T.; and Liang, S. 2019. ISLF: interest shift and latent factors combination model for session-based recommendation. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, 5765–5771.

Sutton, R. S.; and Barto, A. G. 2018. *Reinforcement learning: An introduction*. MIT press.

Swaminathan, A.; and Joachims, T. 2015a. Batch learning from logged bandit feedback through counterfactual risk minimization. *J. Mach. Learn. Res.* 16: 1731–1755.

Swaminathan, A.; and Joachims, T. 2015b. The Self-Normalized Estimator for Counterfactual Learning. In *Advances in Neural Information Processing Systems*, 3231–3239.

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

Vinyals, O.; Blundell, C.; Lillicrap, T.; Wierstra, D.; et al. 2016. Matching networks for one shot learning. In *Advances in neural information processing systems*, 3630–3638.

Wainwright, M. J.; and Jordan, M. I. 2008. *Graphical models, exponential families, and variational inference*. Now Publishers Inc.

Wang, L.; Zhang, W.; He, X.; and Zha, H. 2018. Supervised Reinforcement Learning with Recurrent Neural Network for Dynamic Treatment Recommendation. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2447–2456.

Xiao, T.; Liang, S.; and Meng, Z. 2019a. Dynamic Collaborative Recurrent Learning. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, 1151–1160.

Xiao, T.; Liang, S.; and Meng, Z. 2019b. Hierarchical neural variational model for personalized sequential recommendation. In *The World Wide Web Conference*, 3377–3383.

Xiao, T.; Liang, S.; Shen, W.; and Meng, Z. 2019a. Bayesian deep collaborative matrix factorization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 5474–5481.

Xiao, T.; Ren, J.; Meng, Z.; Sun, H.; and Liang, S. 2019b. Dynamic Bayesian Metric Learning for Personalized Product Search. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, 1693–1702.

Xiao, T.; and Shen, H. 2019. Neural variational matrix factorization with side information for collaborative filtering. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 414–425. Springer.

Xin, X.; Karatzoglou, A.; Arapakis, I.; and Jose, J. M. 2020. Self-Supervised Reinforcement Learning for Recommender Systems. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, 931–940.

Yuan, F.; Karatzoglou, A.; Arapakis, I.; Jose, J. M.; and He, X. 2019. A simple convolutional generative network for next item recommendation. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, 582–590.

Zhao, X.; Xia, L.; Zhang, L.; Ding, Z.; Yin, D.; and Tang, J. 2018a. Deep reinforcement learning for page-wise recommendations. In *Proceedings of the 12th ACM Conference on Recommender Systems*, 95–103.

Zhao, X.; Zhang, L.; Ding, Z.; Xia, L.; Tang, J.; and Yin, D. 2018b. Recommendations with Negative Feedback via Pairwise Deep Reinforcement Learning. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 1040–1048.

Zhao, X.; Zheng, X.; Yang, X.; Liu, X.; and Tang, J. 2020. Jointly Learning to Recommend and Advertise. In *The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 3319–3327.

Zheng, G.; Zhang, F.; Zheng, Z.; Xiang, Y.; Yuan, N. J.; Xie, X.; and Li, Z. 2018. DRN: A Deep Reinforcement Learning Framework for News Recommendation. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*, 167–176.

Zhou, S.; Dai, X.; Chen, H.; Zhang, W.; Ren, K.; Tang, R.; He, X.; and Yu, Y. 2020. Interactive Recommender System via Knowledge Graph-enhanced Reinforcement Learning. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, 179–188.

Zou, L.; Xia, L.; Ding, Z.; Song, J.; Liu, W.; and Yin, D. 2019. Reinforcement Learning to Optimize Long-term User Engagement in Recommender Systems. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2810–2818.

Zou, L.; Xia, L.; Du, P.; Zhang, Z.; Bai, T.; Liu, W.; Nie, J.; and Yin, D. 2020. Pseudo Dyna-Q: A Reinforcement Learning Framework for Interactive Recommendation. In *The Thirteenth ACM International Conference on Web Search and Data Mining*, 816–824.