BCORLE(λ): An Offline Reinforcement Learning and Evaluation Framework for Coupons Allocation in E-commerce Market

Yang Zhang[*]
Faculty of Electronics and Information
Xi'an Jiaotong University

Bo Tang
Alibaba Group

Qingyu Yang[†]
Faculty of Electronics and Information
Xi'an Jiaotong University

Dou An
Faculty of Electronics and Information
Xi'an Jiaotong University

Hongyin Tang
Institute of Software
Chinese Academy of Sciences

Chenyang Xi
Alibaba Group

Xueying Li
Alibaba Group

Feiyu Xiong
Alibaba Group

Abstract

Coupons allocation is an important tool for enterprises to increase the activity and loyalty of users on the e-commerce market. One fundamental problem related is how to allocate coupons within a fixed budget while maximizing users' retention on the e-commerce platform. The online e-commerce environment is complicated and ever changing, so it requires the coupons allocation policy learning can quickly adapt to the changes of the company's business strategy. Unfortunately, existing studies with a huge computation overhead can hardly satisfy the requirements of real-time and fast-response in the real world. Specifically, the problem of coupons allocation within a fixed budget is usually formulated as a Lagrangian problem. Existing solutions need to re-learn the policy once the value of Lagrangian multiplier variable λ is updated, causing a great computation overhead. Besides, a mature e-commerce market often faces tens of millions of users and dozens of types of coupons which construct the huge policy space, further increasing the difficulty of solving the problem. To tackle with above problems, we propose a budget constrained offline reinforcement learning and evaluation with λ-generalization (BCORLE(λ)) framework. The proposed method can help enterprises develop a coupons allocation policy which greatly improves users' retention rate on the platform while ensuring the cost does not exceed the budget. Specifically, λ-generalization method is proposed to lead the policy learning process can be executed according to different λ values adaptively, avoiding re-learning new polices from scratch. Thus the computation overhead is greatly reduced. Further, a novel offline reinforcement learning method and an off-policy evaluation algorithm are proposed for policy learning and policy evaluation, respectively. Finally, experiments on the simulation platform and real-world e-commerce market validate the effectiveness of our approach.

[*]Work was done during an internship at Alibaba Group.
[†]Correspondence:yangqingyu@mail.xjtu.edu.cn

1 Introduction

With the development of the internet industry, the business competition between e-commerce platforms is becoming more and more fierce. It is common for the e-commerce platform to provide users with incentives in the form of monetary prize to attract users to take actions of clicks or conversions [3, 35, 41]. For instance, on online e-commerce market, such as Taobao and Amazon, every day an e-commerce platform sends a coupon to each user who logs on the platform to keep them high retention on the platform. Obviously, the larger value of distributed coupons is, the higher retention rate of users it will gain. However, it may bring huge financial loss to the platform when coupons allocation is too costly. Thus, it is a key problem for the platform to decide an appropriate value of each coupon maximizing the users' retention while limiting the cost not to exceed a fixed budget. Besides, tens of millions of users and dozens of types of coupons lead to an extremely large policy space, which greatly increases the difficulty for solving the problem.

The budget constrained coupons allocation problem is usually formulated as a constrained Markov decision process (CMDP), and then can be converted into a Lagrangian dual problem. In related studies [20, 28, 35], bisection search or gradient descent is used to find the optimal value of Lagrangian multiplier variable λ, and the corresponding optimal coupons allocation policy for given λ is learned using reinforcement learning (RL) methods. Here, offline RL methods are used to avoid potential financial risks in learning process as the budget cannot be recovered once dispensed. Unfortunately, a key problem is that the policy needs to be re-learned every time when the value of λ is updated until the budget constraint is satisfied. Such a repetitive policy learning process brings a great computation overhead, making existing studies unable to satisfy the real-time and fast response requirements in the complicated industrial world. Thus the applications of existing methods are limited.

To address this problem, we propose a λ-generalization method which is also a technique of multi-objective RL [11]. The key idea behind our method is that the tasks of finding optimal value of λ and learning an optimal policy are combined by extending the reward function and training dataset with different values of λ. Thus, the optimal policy can be learned with different values of λ adaptively and the policy learning process only needs to be performed once. Besides, there is another advantage of our method is that there is no need to re-learn the policy when the budget changes. We only need to select an appropriate value of λ which makes its corresponding optimal policy satisfy the new budget constraint. It can help our method respond quickly to the changing business strategy of the company.

The contribution of this work is four-fold. First, a budget constrained offline reinforcement learning and evaluation with λ-generalization (BCORLE(λ)) framework is proposed to solve the CMDP problem of coupons allocation. BCORLE(λ) framework consists of λ-generalization method, an offline policy learning method and an off-policy evaluation method, in which λ-generalization method is proposed to reduce the computation overhead of policy learning. Second, for policy learning, an improved offline RL algorithm called resemble batch-constrained Q-learning (R-BCQ) is proposed in this paper. R-BCQ combines the advantages of two popular offline RL methods: batch-constrained Q-learning (BCQ) [13] and random ensemble mixture (REM) [1]. Specifically, R-BCQ addresses the problem of extrapolation error like BCQ, and retains strong generalization ability like REM. Third, to evaluate the performance of learned policy, a model-free off-policy evaluation method called random ensemble mixture evaluator (REME) is proposed. Finally, the experiments on a simulation platform and a real mobile shopping app validate the effectiveness of the proposed methods.

2 Related Work

Exiting works about the CMDP can be divided into three classes according to the constraint type:

State constrained MDP, also called safe RL [14, 33], refers that there are unsafe states in the full set of states. The objective of the agent is to maximize the cumulative rewards while avoiding falling into unsafe states. In recent studies, Wachi and Sui [32] explored the safe region using a step-wise approach, while Turchetta et al. [30] proposed a hierarchical RL method that a teacher instructs the student to take safe policy. Besides, Lyapunov function [5] was also used in exploring safe RL.

Action constrained MDP refers that the action taken by the agent is limited by a constraint. Apparently, the problem studied in this paper belongs to this CMDP type because selecting the value of distributed coupons is the action of agent and it is constrained by the budget. This kind of problem is commonly formulated as a Lagrangian problem. The aim of solving this problem is to find an optimal

Lagrangian multiplier variable λ and the corresponding optimal policy satisfying the constraint. For instance, a bisection search is developed to find an optimal value of λ, and a stochastic gradient descent method is proposed to learn the policy in [35]. Another approach is to use the gradient descent method to find the optimal value of λ, and the corresponding policy is developed by fitted-Q-learning method [20] or actor-critic method [28]. A key problem in existing approaches is that they both need to update the value of λ many times, and the policy needs to be re-learned once the value of λ is updated, which poses a great computation overhead of policy learning and limits their application in real world. In this paper, we propose a λ-generalization based method to solve this problem.

State-action constrained MDP is a generalization of first two CMDP types. In this CMDP type, a constraint is imposed both on the state and action. Existing studies about this type focus on finding a solution to the non-convex optimization problem in a Lagrangian dual form in theory [7, 23, 37], which is not strongly relevant to the work in this paper, so they will not be introduced in detail.

More related work about offline RL and off-policy evaluation are presented in Appendix A.

3 Problem Formulation

In this section, the problem of coupons allocation with budget is first formulated as a constrained Markov Decision Process, and then converted into a Lagrangian dual problem.

3.1 CMDP Problem of Coupons Distribution

In this paper, the time horizon of coupons allocation problem is divided into T days. Every day, each user receives a coupon after logging into the e-commerce platform. The problem of coupons allocation with a predefined budget is formulated as a CMDP, which is defined as $< S, A, P, R, C, \gamma, b >$. S denotes the state space that consists of a user features and historical retention information of the user on shopping platform. A denotes the action space containing the candidate values of a coupon. P denotes the probability of the state transition after taking some action $S \times A \times S \rightarrow [0, 1]$. R denotes the reward function defined as the user's retention information the next day after receiving a coupon; namely, the reward is 0 if the user logs in the platform tomorrow; otherwise, it is -1. C denotes the cost function defined as the value of the distributed coupon. $\gamma \in [0, 1]$ denotes the discount factor used to weight future rewards. b denotes the budget constraint of the coupons allocation in T days.

The aim of the platform is to develop an optimal policy maximizing the cumulative reward while preventing the cost from exceeding the budget. To avoid potential financial risks in the online interaction between agent and environment when using online RL methods, the CMDP problem is solved using an offline RL method in this paper. Namely, the optimal policy is developed from a fixed data D that is collected from the past interactions between the agent and the environment using a past behaviour policy π_b. The fixed dataset D consists of M tuples: $D = \{(s_i, a_i, r_i, c_i, s_{i+1})\}_{i=1}^{M}$. Accordingly, the CMDP problem can be formulated as:

$$\max_{\pi\in\Pi} J(\pi) = \mathbb{E}_{\tau\sim\pi,\mu}\left[\sum_{t=0}^{T}\gamma^t r_t\right] \quad \text{subject to} \quad C(\pi) = \mathbb{E}_{\tau\sim\pi,\mu}\left[\sum_{t=0}^{T}{\gamma'}^t c_t\right] \le b \tag{1}$$

where π denotes the learned policy, Π is the collection of all policies, and τ is the trajectories distribution that follows the policy π and an initial state distribution μ.

3.2 Lagrangian Dual Problem

To solve the CMDP problem defined by Eq. 1, first, the CMDP problem is converted into its Lagrangian problem as follows:

$$\max_{\pi\in\Pi} L(\pi, \lambda) \quad \text{subject to} \quad \lambda \ge 0 \tag{2}$$

where $L(\pi, \lambda) = J(\pi) - \lambda(C(\pi) - b)$, and λ represents the Lagrangian multiplier variable.

Assumption 1. *There exists a policy π that satisfies the constraint $C(\pi) < b$.*

This assumption is true in the real world; otherwise, the amount of distributed coupon can be reduced to satisfy the constraint. Since the Assumption 1, the Slater's condition [7] holds. Therefore the problem of Eq.2 is equivalent to its Lagrangian dual problem, which can be expressed as:

$$\min_{\lambda\in\mathbb{R}_+}\max_{\pi\in\Pi} L(\pi, \lambda) \tag{3}$$

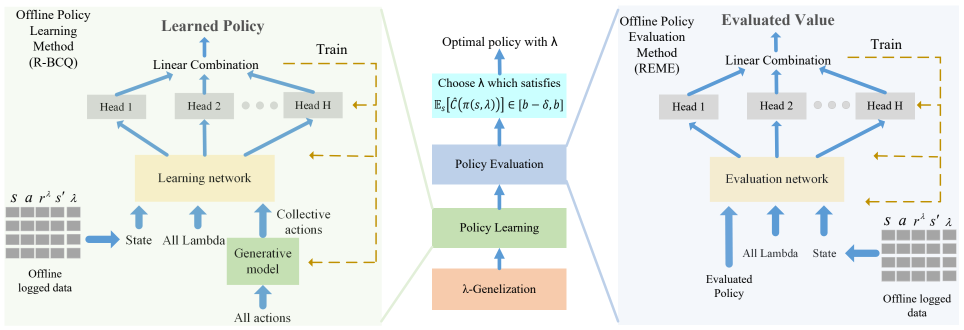


Figure 1: BCORLE(λ) framework consists of three methods: R-BCQ, REME and λ-generalization method. After policy learning and evaluation, a Lagrangian variable and its corresponding optimal policy that makes the cost closest to but not larger than the budget is selected as the final policy.

To solve the Lagrangian dual problem, it is needed to find an optimal value of Lagrangian variable λ^* and the corresponding optimal policy $\pi^*_{\lambda^*}$. And the proof of the existence of the solution of the Lagrangian dual problem is given in the Appendix B.

4 Our Approach

In this section, we propose BCORLE(λ) framework to solve the Lagrangian problem of budget constrained coupons allocation. BCORLE(λ) framework consists of three methods: R-BCQ method for offline policy learning, REME method for off-policy evaluation, and λ-generalization method for making the policy learning and evaluation process can be performed with different values of λ to reduce the computation overhead. Figure 1 provides a summary depiction of BCORLE(λ) framework.

4.1 λ-Generalization

To solve the problem of coupons allocation, the Lagrangian problem defined by Eq. 2 is transformed into a RL problem as follows.

$$\begin{aligned} L(\pi,\lambda) = J(\pi) - \lambda * (C(\pi) - b) &= \mathbb{E}\left[\sum_{t=1}^{T} r(s_t,a_t)\right] - \lambda\left(\mathbb{E}\left[\sum_{t=1}^{T} c(s_t,a_t)\right] - b\right) \\ &= \mathbb{E}\left[\sum_{t=1}^{T} r(s_t,a_t) - \lambda c(s_t,a_t)\right] + \lambda b = \mathbb{E}\left[\sum_{t=1}^{T} r^{\lambda}(s_t,a_t)\right] + \lambda b. \end{aligned} \tag{4}$$

Subsuming the cost into the reward function, we obtain the final reward function to be optimized $r^{\lambda} = r(s_t,a_t) - \lambda c(s_t,a_t)$. Therefore, the offline RL method can be used to develop an optimal offline policy maximizing $\mathbb{E}\left[\sum_{t=1}^{T} r^{\lambda}(s_t,a_t)\right]$ under the constraint of budget.

Different from common RL problem, there is a Lagrangian multiplier variable λ in the reward function. Therefore, the optimal policy changes with the value of λ. To learn the optimal policy while ensuring the cost not to exceed the budget, existing works [20, 35] both employ a λ-update based framework, which is shown in the left of Figure 2. Specifically, for given λ, existing methods both first conduct the policy learning process to find a corresponding optimal policy. Then the policy evaluation process is conducted to judge whether the cost of the learned policy falls within a small range of the budget without exceeding the budget (i.e. $\mathbb{E}_s\left[\widehat{C}(\pi(s,\lambda))\right] \in [b-\delta, b]$ where $b-\delta$ denotes the lower bound of cost). If not, the value of λ will be updated by bisection method [35] or gradient descent method [20], and then the policy learning process will be executed again until the termination condition is satisfied. However, it is necessary for existing methods to re-learn the policy once the value of λ is updated, causing a huge computation overhead.

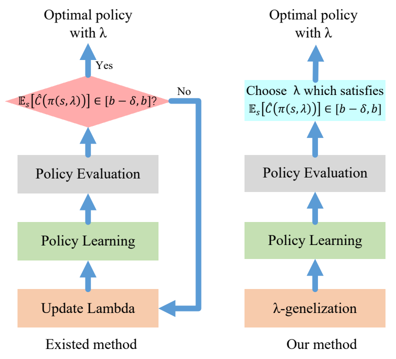


Figure 2: The comparison between the existing method in [20, 35] and our method.

To address this problem, we propose λ-generalization method based on the idea of multi-objective RL method [11]. The key idea of λ-generalization method is to combine two tasks of finding the optimal value of λ and policy learning together by extending the variable λ into the transition tuple, thus the policy learning process can be conducted to learn the optimal policy with different values of λ adaptively. Then, the policy evaluation process is conducted to select the value of λ and the corresponding optimal policy which satisfies the termination condition as the final policy. Requiring not updating the value of λ, the proposed method makes the policy learning and evaluation processes only need to be executed once. Thus, the computation overhead is greatly reduced in our method compared with existing methods. Besides, another advantage of our method is that when the budget changes, there is no need to re-learn the policy. We only need to reselect a value of λ which makes its corresponding policy satisfy the constraint of budget. It helps our method respond quickly to the changing company's business strategy. The flow chart of our method is shown in the right of Figure 2.

Specifically, the proposed λ-generalization method first increases the size of training dataset under different values of λ. For each transition tuple $(s_i, a_i, r_i, c_i, s_{i+1})$, we enlarge it into $\{(s_i, a_i, r_i^{\lambda_j}, c_i, s_{i+1}, \lambda_j)\}_{j=1}^{L}$ where $\lambda_j \in \{\lambda_1, \lambda_2, \dots, \lambda_L\}$ and $r_i^{\lambda_j}$ is defined in Eq. 4. Therefore, the original training dataset $D = \{(s_i, a_i, r_i, c_i, s_{i+1})\}_{i=1}^{M}$ is enlarged into $D' = \{(s_i, a_i, r_i^{\lambda_j}, c_i, s_{i+1}, \lambda_j)\}_{i,j=1,1}^{M,L}$, which is L times as the size of D. It is important to note that we give a detailed introduction about how to determine the candidate value set of λ in Appendix C.

The new dataset D' integrates the state, action and the variable λ, and ensures the processes of policy learning and evaluation are conducted with different values of λ. For each value of λ, the policy learning network generates a policy $\pi(s, \lambda)$, and then the policy evaluation network generates a value $\widehat{C}(\pi(s, \lambda))$ to evaluate the learned policy. Finally, we choose the value of λ and the corresponding policy $\pi(s, \lambda)$ which satisfies $\mathbb{E}_s\left[\widehat{C}(\pi(s, \lambda))\right] \in [b - \delta, b]$ as the final optimal policy.

4.2 Offline Policy Learning: R-BCQ

There are two types of popular offline RL methods: BCQ method and REM method. Two methods have their own advantages. BCQ addresses the problem of extrapolation error via ensuring the match between the state-action pairs generated using the learned policy and the fixed training dataset. REM has high robustness and strong generalization ability with the multi-head network. Unfortunately, the performances of two methods are significantly different on different datasets and it is difficult to judge which of two methods would perform better in the coupons allocation problem of e-commerce market. Thus in this paper, a new method that combines the advantages of two methods is proposed.

Due to the limited size of the fixed training dataset, some state-action pairs generated in learning may not exist in the training dataset. This mismatch causes the estimated value of some state-action pairs deviate greatly from the true value, thus damages the performance of learned policy. To address this problem, we introduce a generative model that imitates the policy in existing dataset. Specifically, the generative model is denoted as $G(a|s, \lambda; \omega)$ that represents an estimation of the behavior policy π_b for given state s and Lagrangian variable λ with parameters ω. When selecting action, we only choose the action which satisfies $G(a|s, \lambda; \omega) / \max_{a'} G(\hat{a}|s, \lambda; \omega) > \beta$ from the action space, where β is a threshold to drop state-action pairs which do not exist or seldom appear in the training dataset.

After choosing candidate action set using the generative model, the policy learning process is conducted using a multi-head network. Each head in the network represents an estimation of the Q-value of the state-action pair. Next, a convex combination of all heads of the network is used as the final estimated Q-value. Compared to common one-head network, multi-head network avoids the Q-value estimation bias problem, thus increases the robustness and generalization ability of the policy learning network. Combined with λ-generalization method, the policy learning network has three inputs: state s, action a and Lagrangian multiplier variable λ. Each head of the network is denoted as $Q_i(s, a, \lambda)$ and the final estimated Q-value is defined as $Q(s, a, \lambda) = \sum_i \alpha_i Q_i(s, a, \lambda)$ where α_i is the weight of each head and $\sum_i \alpha_i = 1, \alpha_i > 0, \forall i$.

Combined with the generative model $G(a|s, \lambda; \omega)$, the learned policy is denoted as follows.

$$\pi(s, \lambda) = \underset{a|G(a|s,\lambda;\omega)/\max_{\hat{a}} G(\hat{a}|s,\lambda;\omega)>\beta}{\arg\max} \sum\nolimits_i \alpha_i Q_i(s, a, \lambda) \tag{5}$$

And the loss of training the policy learning network with parameters θ is defined as follows.

$$L(\theta) = \mathbb{E}_{s,a,r^\lambda,s',\lambda\sim D'}\left[l_\kappa\left(r^\lambda + \gamma \max_{a'|G(a'|s')/\max G(\hat{a}|s')>\beta} \sum_i \alpha_i Q_i'\left(s',a',\lambda\right) - \sum_i \alpha_i Q_i\left(s,a,\lambda\right)\right)\right] \tag{6}$$

The generative model is updated with the loss $L(\omega) = \mathbb{E}_{s,a,\lambda\sim D'}\left[-\log G\left(a|s,\lambda;\omega\right)\right]$.

4.3 Policy Evaluation: REME

A great challenge for the application of offline RL method in real world is to develop an off-policy evaluation mechanism to evaluate the cost and expected utility of policy accurately. To tackle this challenge, in this section, a model-free direct evaluation method called REME is proposed.

The proposed REME method employs a multi-head evaluation network to evaluate the policy and uses an iterative update mechanism to train the evaluation network. Specifically, each head of the evaluation network represents an evaluated value of the given policy, and a convex combination of all heads is used as the final evaluated value. Unlike using the max operator when calculating the target value in the policy learning, we compute the target value using the evaluated policy π in policy evaluation. Besides, combined with the λ-generalization method, the policy evaluation network is able to evaluate the policy π with different values of λ. The evaluated network with parameters $\hat{\theta}$ is updated according to the loss:

$$L(\hat{\theta}) = \mathbb{E}_{s,a,r,s',\lambda\sim D'}\left[l_\kappa\left(r + \gamma \sum_i \alpha_i \widehat{Q}_i\left(s',\pi\left(s',\lambda\right),\lambda\right) - \sum_i \alpha_i \widehat{Q}_i\left(s,a,\lambda\right)\right)\right] \tag{7}$$

Note that when estimating the cost $\widehat{C}(\pi)$ of a given policy π, we will use the cost c to replace the reward r in Eq. 7.

We summarize the steps of the R-BCQ, REME and BCORLE(λ) algorithms in Appendix D.

5 Experiments

In experiments, we study four questions: (1) Does λ-generalization method help to reduce the computation overhead of policy training? (2) How does BCORLE(λ) framework with R-BCQ algorithm perform in comparison to other state-of-the-art offline RL algorithms? (3) How does REME algorithm perform in comparison to other OPE algorithms? (4) How do different values of λ in λ-generalization method affect the performance of proposed approach? To answer these questions, we conduct experiments both on simulation platform and real-world e-commerce platform.

5.1 Simulation Experiments

Before conducting experiments on the real-world platform, we perform the experiments on the simulation platform to ensure there is no risk or unaffordable cost when using the proposed method.

5.1.1 Simulation Environment Setup

According to the background knowledge of real-world e-commerce platform, we create a simulation environment that can be seen as a simplification of real-world e-commerce platform. There are 10000 users in the simulation environment. Each user may log into the simulation platform according to the user preference, and the user will receive a coupon after logging. We simulate the user preference using random distributions. For example, assuming there is a user named "Alice" in the simulation environment. There are two cases of how she would act in the environment: 1) One day when "Alice" logs into the simulation platform, she will receive a coupon c_i. And on the next day, "Alice" would logs into the platform with the probability P_{c_i} that is sampled randomly from a uniform distribution with a precondition: $P_{c_i} \geq P_{c_j}$ if $c_i \geq c_j$. 2) One day when "Alice" doesn't log in the simulation platform, the probability P' of whether she will log into the platform next day is sampled from another uniform distribution and satisfies $P' \leq P_{c_i}, \forall c_i$.

The time span of the simulation environment is 30 days. The state of a user is defined as $s = \{N_{c_1}, N_{c_2}, \ldots, N_{c_k}, M_{c_1}, M_{c_2}, \ldots, M_{c_k}\}$ where N_{c_i} is the number of days when the user receives a coupon c_i in the period of 30 days, and M_{c_i} is the number of days when the user logs into the platform the next day after receiving a coupon c_i the day before. There are 21 items of action that represents the value of coupons, i.e., $0.1, 0.2, 0.3, \ldots, 2.0$ and 2.1 Yuan. The reward is defined as 0 if the user logs into the platform the next day after receiving a coupon; otherwise, it is -1. Besides, there are 21 discrete values of λ: $0, 0.05, 0.1, \ldots, 0.95, 1.0$. Appendix F.1 provides all hyper-parameters of algorithms and more details about the simulation environments.

5.1.2 Evaluation Metrics

Three metrics are used to evaluate the performance of algorithms in the simulation environment. The first metric is **AvgLogins**, which is defined as the average of the number of days all users log on to the simulation platform in one episode of 30 days. The second metric is **AvgCost**, which is defined as the average cost of allocating coupons to all users in one episode of 30 days. The third metric is **return on investment (ROI)**, which is defined as the metric **AvgLogins** divided by the metric **AvgCost**. Obviously, larger values of **AvgLogins** and **ROI** and a smaller value of **AvgCost** indicate better performance.

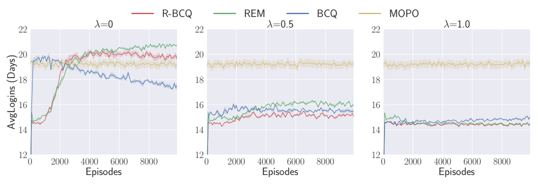
5.1.3 Results

To answer question (1) in simulation environment, we compare the computation overhead of λ-update based approach [20, 35] and our approach in Table 1. The tests are conducted in the same PC environment (Intel Core i7-8700k CPU and a single Nvidia GeForce GTX 1080 GPU) and the termination condition of the learning process for both two methods is the same that the AvgCost is 7.0 Yuan and the AvgLogins is 15 days. Both two methods employ R-BCQ as the offline RL algorithm in the experiment. As Table 1 shows, the required number of episodes for policy learning and the required training time using our approach are both significantly less than those using λ-based mechanism. The results show the effectiveness of our approach in reducing the computation overhead of policy learning.

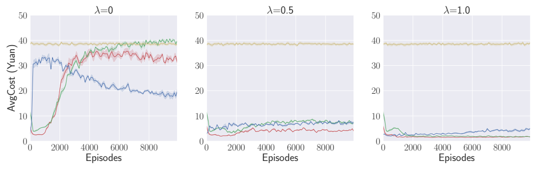
Table 1: The overhead of λ-update based mechanism [20, 35] vs. our approach in terms of the required number of episodes in policy learning and the required training time.

Method	Episodes	Time cost
λ-update approach	14000	2.187h
Our approach	3000	0.439h

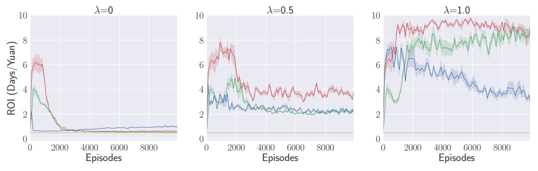
To answer question (2) in simulation environment, we compare R-BCQ method to BCQ [13], REM [1] and another model-based offline RL method: MOPO [38]. Figure 3 shows the result when the value of λ is 0, 0.5 and 1. We first study the performance of R-BCQ, BCQ and REM algorithms. Recall the definition of reward function r^{λ}, it does not include the cost when λ is 0. Thus an optimal strategy must lead to the greatest cost in theory. As shown in Figure 3a and Figure 3b, when λ is 0, R-BCQ method and REM method generate similar policies with higher AvgCost than that of BCQ method, and their AvgLogins values are also higher. This indicates that R-BCQ method and REM method exhibit better performance because the strategies they learned are more in accordance with the expectation. Regarding the ROI, R-BCQ method exhibits a superior performance over REM and BCQ method that the ROI of R-BCQ is higher than that of REM or BCQ method in Figure 3c. This result demonstrates the effectiveness of R-BCQ method that combines the advantages of REM and BCQ methods. Next, we study the performance of MOPO algorithm. Figure 3b show that the AvgCost using MOPO method is significantly higher than that of other methods. The results illustrate MOPO method is not suit for the environment of coupons allocation. We speculate the reason for this result is that it is difficult to estimate the transition and reward function in such a complicated coupons allocation environment which is full of highly randomness, e.g., the



(a) The comparison results of AvgLogins



(b) The comparison results of AvgCost



(c) The comparison results of ROI

Figure 3: We evaluate the performance of BCQ, REM and R-BCQ algorithms on the simulation experiments. The solid and dashed lines show the mean and standard deviation of results over five runs, respectively.

retention information of a user depends on the user's own preference and we can not judge whether the user will logs in the platform the next day accurately. Thus, as a model-based method, MOPO may not gain desired performance in our environment. Based on above results, we consider only deploying R-BCQ, BCQ and REM methods in the e-commerce market of real world.

To answer question (3) in simulation environment, we compare REME method with IS [24], DM [31], DR [17] and FQE method [8, 20]. Results of the errors between the evaluated Q-value of AvgLogins and the real Q-value of AvgLogins are given in Table 2. Results of the errors of evaluating AvgCost are presented in Appendix G. Note that we take the learning step (ls) of evaluated policy (R-BCQ) as 1000, 2000 and 4000 and the value of λ as 0, 0.5 and 1, thus there are nine cases for each OPE method. As show in Table 2, REME method achieves the least error among all methods, which illustrates its high evaluation accuracy. Besides, DM and FQE method achieves sub-optimal performances, while DR and IS method have the worst performances. The results show the superiority of direct evaluation methods that do not require predicting the action distribution of behavior policy (More related work about the direct and non-direct evaluation methods is provided in Appendix A.2).

Table 2: The errors between evaluated value of AvgLogins and real value of AvgLogins using different OPE methods. The value of λ in evaluated policy is 0, 0.5 or 1, and the learning step of evaluated policy is 1000, 2000 or 4000. The bold values represent the best method with least errors.

OPE Method	Errors (Days) when $\lambda = 0$			Errors (Days) when $\lambda = 0.5$			Errors (Days) when $\lambda = 1$		
	ls=1000	ls=2000	ls=4000	ls=1000	ls=2000	ls=4000	ls=1000	ls=2000	ls=4000
IS	3.7837	3.1353	2.1883	2.7559	2.9531	3.2068	0.4299	0.4299	0.4301
DM	0.0201	0.0204	0.0191	0.0204	0.0201	0.0195	0.0192	0.0197	0.0194
DR	1.0126	0.8014	0.5139	2.1328	2.2951	2.5462	0.2258	0.2252	0.2258
FQE	0.2064	0.1483	0.0644	0.2102	0.1844	0.1686	0.0196	0.0192	0.0193
REME	**0.0135**	**0.0127**	**0.0158**	**0.0118**	**0.0085**	**0.0093**	**0.0100**	**0.0082**	**0.0091**

5.1.4 Ablation Study

To answer the question (4), we show the changes of AvgLogins, AvgCost and ROI with regard to various Lagrangian variable λ values under the R-BCQ method. As displayed in Figure 4, the metrics AvgCost and AvgLogins both decrease with the increase of the value of λ. This result is reasonable because the greater the value of λ is, the greater the weight of the cost in the reward function will be. Besides, the result verifies the Assumption 2 introduced in Appendix B, which says that the greater the cost of distributed coupons is, the greater retention rate of users will be. Also, the index ROI increases with the value of λ, which indicates that the metric AvgCost has a greater impact on ROI than AvgLogins.

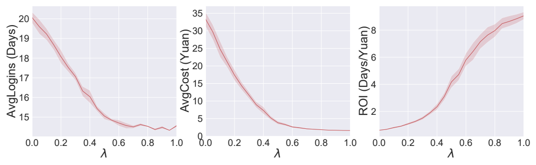


Figure 4: Effect of Lagrangian variable λ on the performance of R-BCQ algorithm in terms of AvgLogins (Left), AvgCost (Mid) and ROI (right)

5.2 Real-world Experiments

5.2.1 Real-world Platform

We apply our methods in the scenario of daily check-in of Taobao Deals, which is a mobile shopping app launched by Alibaba Group in 2020 with over 10 million daily active users. Each user receives a coupon when opening the app for the first time in a day. Users can use the coupon while shopping to make a subtraction of payment. Each coupon is valid for 24 hours and can be used only once. The screen views of the shopping app and the daily check-in scenario are shown in the Appendix E.

We collect the real data sampled from the Taobao Deals app to create the training dataset that consists of over 2 million users' daily check-in records from January 2021 to March 2021.

The state in real environment consists of the contextual information and behavior information of a user. The contextual information of a user includes the gender, age, and location, etc. The behavior information is the retention information of users in the app, just like the state we defined in the simulation environment. The action set consists of 13 items of coupons: $0.5, 0.55, 0.6, 0.65, 0.7, 0.75, 0.8, 0.85, 0.9, 0.95, 1.0, 1.05$ and 1.1 Yuan. The reward is defined as 0 if the user logs into the app the next day after receiving a coupon; otherwise it is -1. The length of one episode is 7 days. The seven-day budget for the coupon allocation per user is 4.1 Yuan. The value of δ is 0.1. Appendix F.2 provides all hyper-parameters of the real-world experiments environment.

5.2.2 Baselines and Evaluation Metrics

In addition to BCQ and REM methods, we compare our method with two uplift model based baselines that are currently adopted to allocate coupons in Taobao Deals: Logistics regression + Linear programming (LR+LP) [25] and Gradient boost decision tree + Linear programming (GBDT+LP) [26, 27]. Two methods both consists of two modules: a prediction model to predict users' retention intent after receiving different coupons using LR or GBDT method and an action selection model using LP method. These two methods are proved to be able to gain promising results in the e-commerce market.

In addition to metrics **AvgLogins**, **AvgCost** and **ROI**, we also employ a metric **ROI Imp**, which is defined as the ROI improvement rate using other four methods compared to using LR+LP method.

5.2.3 Results

We first choose the policy with the value of λ which makes the cost satisfy the condition $\mathbb{E}_{s\sim D'}\left[\widehat{C}\left(s,\pi\left(s,\lambda\right),\lambda\right)\right]\in[b-\delta,b]$. As shown in the Figure 5, we choose REM policy with $\lambda=0.25$, BCQ policy with $\lambda=0.35$ and R-BCQ policy with $\lambda=0.55$ as coupons allocation policies in the e-commerce platform. The results also verify that λ-generalization method can scale the performance of offline RL algorithms with different values of λ in question (4).

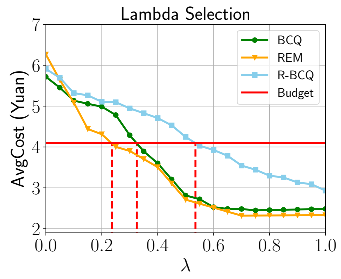


Figure 5: The results of REM, BCQ and R-BCQ methods with the value of λ ranging from 0 to 1

To answer question (2) in real environment, we deploy all methods in Taobao Deals app for two weeks from March 9th to March 22nd. The experiments follow the online A/B testing methodology and the results are shown in Table 3. From the table, we observe when AvgCost of all methods is similar, the evaluation metrics AvgLogins and ROI of BCQ, REM and R-BCQ methods are obviously higher than those of LR+LP and GBDT+LP methods. This result shows that offline RL methods gain superior performance over common uplift model based methods. The reason for this result is that the uplift model based method aims to maximize the benefit in one day while ignoring future benefits in making decision [35]. Besides, we observe that R-BCQ method gains the biggest ROI in three RL methods. This conclusion is consistent with the conclusion drawn in the simulation experiments. Experimental results illustrate the effectiveness of R-BCQ method in increasing users' activity within the limitation of budget.

Table 3: The online results in Taobao Special Offer Edition app during two weeks. The bold values represent the method with the greatest AvgLogins, greatest ROI and least AvgCost.

Method	Results in first week				Results in second week			
	AvgLogins (Days)	AvgCost (Yuan)	ROI (Days/Yuan)	ROI Imp	AvgLogins (Days)	AvgCost (Yuan)	ROI (Days/Yuan)	ROI Imp
LR+LP	5.0416	**4.0628**	1.2409	0	5.3099	4.0684	1.3052	0
GBDT+LP	5.0442	4.0776	1.2371	-0.31%	5.3802	4.0756	1.3201	1.14%
BCQ	5.6832	4.0740	1.3950	12.42%	5.8789	4.0698	1.4445	10.67%
REM	5.7108	4.0644	1.4051	13.23%	5.9092	4.0729	1.4509	11.16%
R-BCQ	**5.8252**	4.0654	**1.4329**	**15.47%**	**5.9871**	**4.0528**	**1.4773**	**13.18%**

To answer question (3) in real environment, we compare REME algorithm with IS, DM, DR and FQE evaluation methods in the online experiments. The evaluated policy is R-BCQ policy deployed in the real platform. Experimental results of the errors between the evaluated and real value of AvgLogins and the errors between the evaluated and real value of AvgCost are shown in Table 4. As shown in table, REME gains the best performance among all methods with the least evaluation errors of AvgLogins and AvgCost. Besides, we also observe that DM and FQE perform worse than REME, and IS performs worst among all methods. These results are roughly consistent with the conclusion drawn in the simulation experiments.

Table 4: The errors of REME vs. other methods when evaluating AvgLogins and AvgCost. Bold values represent the best method with the least errors.

OPE Method	Errors of AvgLogins (Days)	Errors of AvgCost (Yuan)
IS	0.2203	0.2754
DM	0.1417	0.1923
DR	0.1848	0.1881
FQE	0.1933	0.1515
REME	**0.0443**	**0.0221**

Note that the great computation overhead makes the λ-update based approach cannot be deployed in the real shopping app. Thus there is no need to conduct repeated experiments to answer question (1).

6 Conclusion

In this paper, we propose a BCORLE(λ) framework to solve the problem of budget constrained coupons allocation in e-commerce market. The proposed λ-generalization method contributes to developing an optimal coupons allocation policy with a small computation overhead of policy learning compared with existing methods. For policy learning, we propose R-BCQ method that addresses the problem of extrapolation error and retains the strong ability of generalization of learned policy. Further, we propose a novel off-policy evaluation method called REME to evaluate the performance of policies accurately. Experimental results obtained using simulation environment and online real-world e-commerce market verify the effectiveness of our approach.

7 Acknowledgments and Disclosure of Funding

We thank the members of the business platform of Alibaba Group and Chengwei Zhang for their insightful feedback and comments. This work was supported in part by the National Natural Science Foundation of China under Grant 61973247 and 62173268, China Postdoctoral Science Foundation 2021M692566.

References

[1] Rishabh Agarwal, Dale Schuurmans, and Mohammad Norouzi. An optimistic perspective on offline reinforcement learning. In *International Conference on Machine Learning*, pages 104–114. PMLR, 2020.

[2] Oron Anschel, Nir Baram, and Nahum Shimkin. Averaged-dqn: Variance reduction and stabilization for deep reinforcement learning. In *International Conference on Machine Learning*, pages 176–185. PMLR, 2017.

[3] Nikolay Archak, Vahab Mirrokni, and Shanmugavelayutham Muthukrishnan. Budget optimization for online campaigns with positive carryover effects. In *International Workshop on Internet and Network Economics*, pages 86–99. Springer, 2012.

[4] Xinyue Chen, Zijian Zhou, Zheng Wang, Che Wang, Yanqiu Wu, and Keith Ross. Bail: Best-action imitation learning for batch deep reinforcement learning. *arXiv preprint arXiv:1910.12179*, 2019.

[5] Yinlam Chow, Ofir Nachum, Edgar Duenez-Guzman, and Mohammad Ghavamzadeh. A lyapunov-based approach to safe reinforcement learning. *arXiv preprint arXiv:1805.07708*, 2018.

[6] Will Dabney, Mark Rowland, Marc Bellemare, and Rémi Munos. Distributional reinforcement learning with quantile regression. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[7] Dongsheng Ding, Kaiqing Zhang, Tamer Basar, and Mihailo Jovanovic. Natural policy gradient primal-dual method for constrained markov decision processes. *Advances in Neural Information Processing Systems*, 33, 2020.

[8] Yaqi Duan, Zeyu Jia, and Mengdi Wang. Minimax-optimal off-policy evaluation with linear function approximation. In *International Conference on Machine Learning*, pages 2701–2709. PMLR, 2020.

[9] Mehrdad Farajtabar, Yinlam Chow, and Mohammad Ghavamzadeh. More robust doubly robust off-policy evaluation. In *International Conference on Machine Learning*, pages 1447–1456. PMLR, 2018.

[10] Stefan Faußer and Friedhelm Schwenker. Neural network ensembles in reinforcement learning. *Neural Processing Letters*, 41(1):55–69, 2015.

[11] Eli Friedman and Fred Fontaine. Generalizing across multi-objective reward functions in deep reinforcement learning. *arXiv preprint arXiv:1809.06364*, 2018.

[12] Scott Fujimoto, Edoardo Conti, Mohammad Ghavamzadeh, and Joelle Pineau. Benchmarking batch deep reinforcement learning algorithms. *arXiv preprint arXiv:1910.01708*, 2019.

[13] Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In *International Conference on Machine Learning*, pages 2052–2062. PMLR, 2019.

[14] Javier Garcıa and Fernando Fernández. A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research*, 16(1):1437–1480, 2015.

[15] Anna Harutyunyan, Marc G Bellemare, Tom Stepleton, and Rémi Munos. Q (lambda) with off-policy corrections. In *International Conference on Algorithmic Learning Theory*, pages 305–320. Springer, 2016.

[16] Nan Jiang and Jiawei Huang. Minimax value interval for off-policy evaluation and policy optimization. *Advances in Neural Information Processing Systems*, 33, 2020.

[17] Nan Jiang and Lihong Li. Doubly robust off-policy value evaluation for reinforcement learning. In *International Conference on Machine Learning*, pages 652–661. PMLR, 2016.

[18] Aviral Kumar, Justin Fu, George Tucker, and Sergey Levine. Stabilizing off-policy q-learning via bootstrapping error reduction. *arXiv preprint arXiv:1906.00949*, 2019.

[19] Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. *arXiv preprint arXiv:2006.04779*, 2020.

[20] Hoang Le, Cameron Voloshin, and Yisong Yue. Batch policy learning under constraints. In *International Conference on Machine Learning*, pages 3703–3712. PMLR, 2019.

[21] Rémi Munos, Tom Stepleton, Anna Harutyunyan, and Marc G Bellemare. Safe and efficient off-policy reinforcement learning. *arXiv preprint arXiv:1606.02647*, 2016.

[22] Cosmin Paduraru. *Off-policy evaluation in Markov decision processes*. PhD thesis, Ph. D. Dissertation. McGill University, 2012.

[23] Santiago Paternain, Luiz FO Chamon, Miguel Calvo-Fullana, and Alejandro Ribeiro. Constrained reinforcement learning has zero duality gap. *arXiv preprint arXiv:1910.13393*, 2019.

[24] Doina Precup. Eligibility traces for off-policy policy evaluation. *Computer Science Department Faculty Publication Series*, page 80, 2000.

[25] Krzysztof Rudaś and Szymon Jaroszewicz. Linear regression for uplift modeling. *Data Mining and Knowledge Discovery*, 32(5):1275–1305, 2018.

[26] Piotr Rzepakowski and Szymon Jaroszewicz. Decision trees for uplift modeling. In *2010 IEEE International Conference on Data Mining*, pages 441–450. IEEE, 2010.

[27] Piotr Rzepakowski and Szymon Jaroszewicz. Decision trees for uplift modeling with single and multiple treatments. *Knowledge and Information Systems*, 32(2):303–327, 2012.

[28] Chen Tessler, Daniel J Mankowitz, and Shie Mannor. Reward constrained policy optimization. *arXiv preprint arXiv:1805.11074*, 2018.

[29] Philip Thomas and Emma Brunskill. Data-efficient off-policy policy evaluation for reinforcement learning. In *International Conference on Machine Learning*, pages 2139–2148. PMLR, 2016.

[30] Matteo Turchetta, Andrey Kolobov, Shital Shah, Andreas Krause, and Alekh Agarwal. Safe reinforcement learning via curriculum induction. *arXiv preprint arXiv:2006.12136*, 2020.

[31] Cameron Voloshin, Hoang M Le, Nan Jiang, and Yisong Yue. Empirical study of off-policy policy evaluation for reinforcement learning. *arXiv preprint arXiv:1911.06854*, 2019.

[32] Akifumi Wachi and Yanan Sui. Safe reinforcement learning in constrained markov decision processes. In *International Conference on Machine Learning*, pages 9797–9806. PMLR, 2020.

[33] Akifumi Wachi, Yanan Sui, Yisong Yue, and Masahiro Ono. Safe exploration and optimization of constrained mdps using gaussian processes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[34] Yifan Wu, George Tucker, and Ofir Nachum. Behavior regularized offline reinforcement learning. *arXiv preprint arXiv:1911.11361*, 2019.

[35] Shuai Xiao, Le Guo, Zaifan Jiang, Lei Lv, Yuanbo Chen, Jun Zhu, and Shuang Yang. Model-based constrained mdp for budget allocation in sequential incentive marketing. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 971–980, 2019.

[36] Ming Yin and Yu-Xiang Wang. Asymptotically efficient off-policy evaluation for tabular reinforcement learning. In *International Conference on Artificial Intelligence and Statistics*, pages 3948–3958. PMLR, 2020.

[37] Ming Yu, Zhuoran Yang, Mladen Kolar, and Zhaoran Wang. Convergent policy optimization for safe reinforcement learning. *arXiv preprint arXiv:1910.12156*, 2019.

[38] Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. Mopo: Model-based offline policy optimization. *arXiv preprint arXiv:2005.13239*, 2020.

[39] Ruiyi Zhang, Bo Dai, Lihong Li, and Dale Schuurmans. Gendice: Generalized offline estimation of stationary values. *arXiv preprint arXiv:2002.09072*, 2020.

[40] Shangtong Zhang, Bo Liu, and Shimon Whiteson. Gradientdice: Rethinking generalized offline estimation of stationary values. In *International Conference on Machine Learning*, pages 11194–11203. PMLR, 2020.

[41] Kui Zhao, Junhao Hua, Ling Yan, Qi Zhang, Huan Xu, and Cheng Yang. A unified framework for marketing budget allocation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1820–1830, 2019.

Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [Yes] See Section **??**.
- Did you include the license to the code and datasets? [No] The code and the data are proprietary.
- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] We believe there is no negative societal impact.

   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]

2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] See Section 3.2, Appendix B and Appendix C
   (b) Did you include complete proofs of all theoretical results? [Yes] See Appendix B and Appendix C

3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [N/A] Our works uses own data, and the data used in the simulation experiment is released in supplementary material.
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]

5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]