Two-Stage Constrained Actor-Critic for Short Video Recommendation

Qingpeng Cai
Kuaishou Technology
Beijing, China
caiqingpeng@kuaishou.com

Zhenghai Xue
Kuaishou Technology
Beijing, China
xuezhenghai@kuaishou.com

Chi Zhang
Kuaishou Technology
Beijing, China
zhangchi08@kuaishou.com

Wanqi Xue
Kuaishou Technology
Beijing, China
xuewanqi@kuaishou.com

Shuchang Liu
Kuaishou Technology
Beijing, China
liushuchang@kuaishou.com

Ruohan Zhan
Hong Kong University of Science and Technology
Hong Kong, China
rhzhan@ust.hk

Xueliang Wang
Kuaishou Technology
Beijing, China
wangxueliang03@kuaishou.com

Tianyou Zuo
Kuaishou Technology
Beijing, China
zuotianyou@kuaishou.com

Wentao Xie
Kuaishou Technology
Beijing, China
xiewentao@kuaishou.com

Dong Zheng
Kuaishou Technology
Beijing, China
zhengdong@kuaishou.com

Peng Jiang*
Kuaishou Technology
Beijing, China
jiangpeng@kuaishou.com

Kun Gai
Unaffiliated
Beijing, China
gai.kun@qq.com

ABSTRACT

The wide popularity of short videos on social media poses new opportunities and challenges to optimize recommender systems on the video-sharing platforms. Users sequentially interact with the system and provide complex and multi-faceted responses, including `WatchTime` and various types of interactions with multiple videos. On the one hand, the platforms aim at optimizing the users' cumulative `WatchTime` (main goal) in the long term, which can be effectively optimized by Reinforcement Learning. On the other hand, the platforms also need to satisfy the constraint of accommodating the responses of multiple user interactions (auxiliary goals) such as `Like`, `Follow`, `Share`, etc. In this paper, we formulate the problem of short video recommendation as a Constrained Markov Decision Process (CMDP). We find that traditional constrained reinforcement learning algorithms fail to work well in this setting. We propose a novel two-stage constrained actor-critic method: At stage one, we learn individual policies to optimize each auxiliary signal. In stage two, we learn a policy to (i) optimize the main signal and (ii) stay close to policies learned in the first stage, which effectively guarantees the performance of this main policy on the auxiliaries. Through extensive offline evaluations, we demonstrate the effectiveness of our method over alternatives in both optimizing the main goal as well as balancing the others. We further show the advantage of our method in live experiments of short video recommendations, where it significantly outperforms other baselines in terms of both `WatchTime` and interactions. Our approach has been fully launched in the production system to optimize user experiences on the platform.

CCS CONCEPTS

• **Information systems** → **Recommender systems**; • **Computing methodologies** → **Reinforcement learning**.

KEYWORDS

constrained reinforcement learning, recommender systems, short video recommendation

ACM Reference Format:
Qingpeng Cai, Zhenghai Xue, Chi Zhang, Wanqi Xue, Shuchang Liu, Ruohan Zhan, Xueliang Wang, Tianyou Zuo, Wentao Xie, Dong Zheng, Peng Jiang, and Kun Gai. 2023. Two-Stage Constrained Actor-Critic for Short Video Recommendation. In *Proceedings of the ACM Web Conference 2023 (WWW '23), May 1–5, 2023, Austin, TX, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3543507.3583259

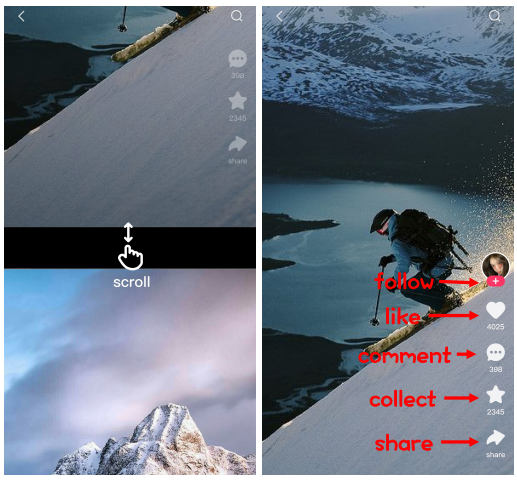
*Corresponding author

WWW '23, May 1–5, 2023, Austin, TX, USA

ACM ISBN 978-1-4503-9416-1/23/04…$15.00
https://doi.org/10.1145/3543507.3583259

1 INTRODUCTION

The surging popularity of short videos has been changing the status quo of social media. Short video consumption has brought in huge business opportunities for organizations. As a result, there has been an increasing interest in optimizing recommendation strategies [Gong et al. 2022; Lin et al. 2022; Wang et al. 2022a; Zhan et al.



(a) (b)

Figure 1: An example of a popular short video (TikTok, Kuaishou, etc) platform.

2022] for short video platforms. Users interact with the platform by scrolling up and down and watching multiple videos as shown in Figure 1(a). Users provide multi-dimensional responses at each video. As shown in the left part of Figure 1(b), potential responses from a user after consuming a video include `WatchTime` (the time spent on watching the video), and several types of interactions: `Follow` (follow the author of the video), `Like` (Like this video), `Comment` (provide comments on the video), `Collect` (Collect this video), `Share` (share this video with his/her friends), etc.

On the one hand, the main goal of the platform is to optimize the cumulative `WatchTime` of multiple videos, as `WatchTime` reflects user attention and is highly related to daily active users (DAU). Recently, a growing literature has focused on applying reinforcement learning (RL) to recommender systems, due to its ability to improve cumulative reward [Afsar et al. 2021; Chen et al. 2019b, 2018; Gao et al. 2022a; Ge et al. 2021; Liu and Yang 2019; Ma et al. 2020; Nemati et al. 2016; Wang et al. 2022b; Xian et al. 2019; Xin et al. 2022; Zhao et al. 2018, 2017; Zou et al. 2019]. In particular, `WatchTime` , can be effectively cumulatively maximized to increase user spent time across multiple videos with RL approaches. On the other hand, other responses such as `Like`/`Follow`/`Share` also reflect user satisfaction levels. Thus the platform needs to satisfy the constraints of user interactions. Thereby, established recommender systems that exclusively optimize a single objective (such as gross merchandise volume for e-commerce platforms [Pi et al. 2020]) is no longer sufficient—the applied systems should take all aspects of responses into consideration to optimize user experiences.

In this paper, we model the problem of short video recommendation as a Constrained Markov Decision Process: users serve as the environments, and the recommendation algorithm is the agent; at each time step the agent plays an action (recommend a video to the user), the environment sends multiple rewards (responses) to the agent. The objective of the agent is to maximize the cumulative `WatchTime` (main goal) subject to the constraints of other interaction responses (auxiliary goals). Our aim is different from Pareto optimality that aims to find a Pareto optimal solution [Chen et al. 2021; Lin et al. 2019; Sener and Koltun 2018], which may not prioritize the main goal of the system.

The problem of this constrained policy optimization is much more challenging as compared to its unconstrained counterpart. A natural idea would be applying standard constrained reinforcement learning algorithms that maximize the Lagrangian with prespecified multipliers [Tessler et al. 2018]. However, such method can not apply to our setting for the following two reasons:

First, it is not sufficient to use a single policy evaluation model to estimate the Lagrangian dual objective due to different types of responses from the user. Such response combination is not adequate, particularly for responses with their own discount factors—the formulation of temporal difference error in value-based models only allows for a single discount value. In scenarios where one discount factor suffices, it can still be difficult for a single value model to evaluate the policy accurately, especially when different responses are observed at various frequencies, as typical for short video recommendations. The `WatchTime` response is dense and observed from each video view, while the interaction-signal such as `Like`/`Follow`/`Share` is much more sparse and may not be provided within dozens of views. The signal from the sparse responses will be weakened by the dense responses when naively summing them up together. To address this multi-response evaluation difficulty, we separately evaluate each response via its own value model, which allows for response-specific discount factors and mitigates the interference on evaluation from one response on another. Experiments in Section 4.1 validates the effectiveness of this method.

Second, different from only one constraint is considered in [Tessler et al. 2018], multiple constraints exist in recommender systems, especially in short video systems. We find that it is more difficult for algorithms that maximize the Lagrangian to optimize due to larger search space of multi-dimensional Lagrangian multipliers. It is time costly to grid search on the Lagrangian multipliers as the training of reinforcement learning algorithms takes long time. On account of this, we propose to firstly learn policies to optimize each auxiliary response and then "softly" regularize the policy of the main response to be close to others instead of searching optimal value of Lagrangian multipliers. We theoretically prove the closed form of the optimal solution. We demonstrate empirically that our approach can better maximize the main response and balance other responses in both offline and live experiments.

Together, we summarize our contributions as below:

- **Constrained Optimization in Short Video Recommendations**: We formalize the problem of constrained policy learning in short video recommendations, where different responses may be observed at various frequencies, and the agent maximizes one with the constraint of balancing others.
- **Two-Stage Constrained Actor-Critic Algorithm** We propose a novel two-stage constrained actor-critic algorithm that effectively tackles the challenge: (1) Multi-Critic Policy Estimation: To better evaluate policy on multiple responses that may differ in discount factors and observation frequencies, we propose to separately learn a value model to evaluate each response. (2) Two-Stage Actor Learning: We propose a

two-stage actor learning method which firstly learns a policy to optimize each auxiliary response and secondly softly regularizes the policy of the main response to be not far from others, which we demonstrate to be a more effective way in constrained optimization with multiple constraints as compared with other alternatives.
- **Significant Gains in Offline and Live Experiments**: We demonstrate the effectiveness of our method in both offline and live experiments.
- **Deployment in real world short video application**: We fully launch our method in a popular short video platform.

2 RELATED WORK

Reinforcement Learning for Recommendation. There is a growing literature in applying RL to recommender systems, for its ability to optimize user long-term satisfaction [Afsar et al. 2021]. Value-based approaches estimate user satisfaction of being recommended an item from the available candidate set and then select the one with the largest predicted satisfaction [Chen et al. 2018; Liu and Yang 2019; Nemati et al. 2016; Zhao et al. 2018]. Policy-based methods directly learn the policy (which item to recommend) and optimize it in the direction of increasing user satisfaction [Chen et al. 2019b,a; Ma et al. 2020; Xian et al. 2019]. Recently, growing attention has been paid to adapting reinforcement learning for more complex recommendation applications beyond optimizing one single objective, such as promoting equal exposure opportunities for content items [Ge et al. 2021], increasing diversity and novelty of recommendations [Stamenkovic et al. 2021], and characterizing more comprehensive user dynamics with representational reward shaping [Chen et al. 2021]; we view our work as complementary to the third line. In face of the multi-faceted user responses, the system in real applications often has preferences on different types of user responses, for which we propose the constrained optimization problem in contrast to pursuing the Pareto optimality as proposed in [Chen et al. 2021] and [Ge et al. 2022].

Constrained Reinforcement Learning. Our work is also closely related to the literature of constrained reinforcement learning, where the sequential decision making problem is formulated into a constrained Markov Decision Process [Sutton and Barto 2018], and the policy learning procedure is expected to respect the constraints[Chow et al. 2017; Dalal et al. 2018; Garcıa and Fernández 2015; Liu et al. 2021; Tessler et al. 2018]. As an example, [Tessler et al. 2018] propose to update the policy and the Lagrangian multiplier alternatively and prove the convergence of their algorithm to a fixed point. This approach however only models one constraint, and can not scale well on problems with multiple constraints. In contrast, for each auxiliary response, we learn a policy to maximize it specifically, then we "softly" regularize the main policy to be close to others. We show empirically that this is a more effective way for constrained policy learning when dealing with multiple responses in recommender systems. Different from [Nair et al. 2020] that studies in offline RL and regularizes the learned policy to be near to one behavior policy, we softly restrict the policy within other policies maximizing other auxiliary responses.

Multi-objective Optimization. We also discuss a relevant line on multi-objective optimization. To trade off different objectives, methods in this field can be broadly categorized into two classes: the Pareto optimization and the joint optimization with pre-specified weights. The goal of Pareto optimization is to find a solution such that no other solutions can concurrently improve all objectives, named as *Pareto optimality* [Chen et al. 2021; Ge et al. 2022; Nguyen et al. 2020; Sener and Koltun 2018]. However, a Pareto optimal solution may not prioritize the objective that is most valued in applications. The other method combines different objectives together into a single one via pre-specifying the weights [Mossalam et al. 2016; White et al. 1980]. However, it is difficult to quantify these weights that can accurately reflect preferences in real applications [Tessler et al. 2018].

3 CONSTRAINED MARKOV DECISION PROCESS FOR SHORT VIDEO RECOMMENDATION

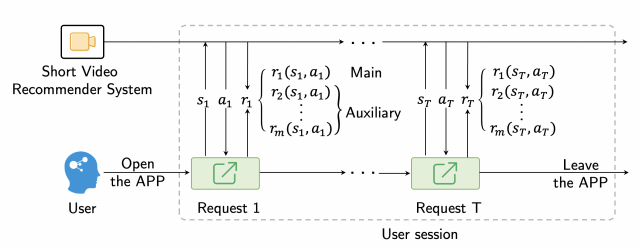


Figure 2: The MDP of short video recommendation.

We start by formulating the problem of short video recommendation, which is shown in Figure 2. When a user u opens the app, a new *session* starts. A session consists of multiple *requests*. At each request t the recommender system (agent) takes an *action* a_t that recommends the user a video based on the user current *state*. Then the user provides *multi-faceted* responses (such as `WatchTime`, `Like`, `Share`, and `Follow`) on the shown video, which are received by the agent as vector-valued *reward* signal. After the user leaves the app, the session ends. The goal of the recommender system is to optimize cumulative reward of the main response (*e.g.*, `WatchTime`), with the constraint of not sacrificing others much.

We model the above procedure as a Constrained Markov Decision Process (CMDP) [Sutton and Barto 2018] $(S, A, P, R, C, \rho_0, \Gamma)$, where S is the state space of user current representation s_t, A is the action space (and each action a_t corresponds to a recommended video for one request), $P : S \times A \to \Delta(S)$ captures the state transition, $R : S \times A \to \mathbb{R}^m$ defines the vector-valued reward function that yields m different rewards $r(s_t, a_t) = \big(r_1(s_t, a_t), \ldots, r_m(s_t, a_t)\big)$, ρ_0 is the initial state distribution, $\Gamma = (\gamma_1, \ldots, \gamma_m) \in (0, 1)^m$ denotes the vector of discount factor for reward of each response. C specifies the constraints on the auxiliary responses, which denotes the lower bound of the total numbers of signals of other objectives.

Define the vector-valued discounted cumulative reward R_t as $R_t = \sum_{t'=t}^{T} \Gamma^{t'-t} \cdot r(s_{t'}, a_{t'})$, where T is the session length (i.e., the number of requests), $\Gamma^b = \big(\gamma_1^b, \ldots, \gamma_m^b\big)$, and $\mathbf{x} \cdot \mathbf{y}$ denotes the pointwise product. Let $V^\pi(s) = \big(V_1^\pi(s), \ldots, V_m^\pi(s)\big)$ be the state value

$E_\pi[R_t|s_t = s]$ under actions sampled in accordance with policy π and $Q(s,a) = \left(Q_1^\pi(s,a), \ldots, Q_m^\pi(s,a)\right)$ be its state-action value $E_\pi[R_t|s_t = s, a_t = a]$. Denote ρ_π as the state distribution induced by policy π. Without loss of generality, we set the first response as our main response. The goal is to learn a recommendation policy $\pi(\cdot|s)$ to solve the following optimization problem:

$$\begin{aligned} \max_{\pi} \quad & E_{\rho_\pi}\left[V_1^\pi(s)\right] \\ \text{s.t.} \quad & E_{\rho_\pi}\left[V_i^\pi(s)\right] \geq C_i, \quad i = 2, \ldots, m \end{aligned} \tag{1}$$

where C_i is constraint on the *auxiliary* response i.

4 TWO-STAGE CONSTRAINED ACTOR-CRITIC

In this section, we propose a novel two-stage constrained actor-critic method, addressing the learning challenges in the context of short video recommendation:

- **Multi-Critic Policy Estimation** We propose to estimate the responses separately to better estimate dense and sparse signals.
- **Stage One** For each auxiliary response, we learn a policy to optimize its cumulative reward.
- **Stage Two** For the main response, we learn a policy to optimize its cumulative reward, while softly limiting it to be close to other policies that are learned to optimize the auxiliary.

We first discuss the advantage of evaluating different policies separately over estimating jointly. Secondly, we elaborate our method in the settings of online learning with stochastic policies in Sections 4.2 and 4.3. We then discuss its extensions to the offline setting and deterministic policies.

4.1 Multi-Critic Policy Estimation

We showcase the advantage of separate evaluation for each response over a joint evaluation of summed response. Specifically, we consider two types of responses from each video view: `WatchTime` and interactions (which is an indicator function of whether the interactions happen during the view).

- For the joint evaluation, we learn a value model V_{joint} with reward as a sum of `WatchTime` and interactions.
- For the separate evaluation, we learn two value models V_w and V_i with reward as `WatchTime` and interactions respectively. Define the value of separate evaluation as $V_{separate} = V_w + V_i$

For fair comparison, we share the same discount factor 0.95 for all value models and train them on the same data collected from a popular short video platform for one day. To evaluate the accuracy of the value model in terms of `WatchTime` and interactions, we compute the correlation between model values V_{joint} and $V_{separate}$ with the Monte Carlo value of the sum of the corresponding responses in each session. As compared to V_{joint}, $V_{separate}$ is more correlated with `WatchTime` and interactions by 0.19% and 0.14% respectively(a 0.1% improvement on `WatchTime` and interactions is significant), demonstrating that the separate evaluation better learns different reward responses than jointly learning.

4.2 Stage One: Policy Learning for Auxiliary Responses

At this stage, we learn policies to optimize the cumulative reward of each auxiliary response separately. For completeness, we write out our procedure for stochastic policies [Williams 1992]. Considering response i, let the learned actor and the critic be parameterized by π_{θ_i} and V_{ϕ_i} respectively. At iteration k, we observe sample (s, a, s') collected by $\pi_{\theta_i^{(k)}}$, *i.e.*, $s \sim \rho_{\pi_{\theta_i^{(k)}}}, a \sim \pi_{\theta_i^{(k)}}(\cdot|s)$ and $s' \sim P(\cdot|s,a)$. We update the critic to minimize the Bellman equation:

$$\phi_i^{(k+1)} \leftarrow \arg\min_{\phi} E_{\pi_{\theta_i^{(k)}}}\left[\left(r_i(s,a) + \gamma_i V_{\phi_i^{(k)}}(s') - V_\phi(s)\right)^2\right]. \tag{2}$$

We update the actor to maximize the advantage:

$$\begin{aligned} \theta_i^{(k+1)} &\leftarrow \arg\max_{\theta} E_{\pi_{\theta_i^{(k)}}}\left[A_i^{(k)} \log\left(\pi_\theta(a|s)\right)\right] \\ \text{where} \quad A_i^{(k)} &= r_i(s,a) + \gamma_i V_{\phi_i^{(k)}}(s') - V_{\phi_i^{(k)}}(s). \end{aligned} \tag{3}$$

4.3 Stage Two: Softly Constrained Optimization of the Main Response

After pre-training the policies $\pi_{\theta_2}, \ldots, \pi_{\theta_m}$ that optimize the auxiliary responses, we now move onto the second stage of learning the policy to optimize the main response. We propose a new constrained policy optimization method with multiple constraints.

Let the actor and the critic be π_{θ_1} and V_{ϕ_1} respectively. At iteration k, we similarly update the critic to minimize the Bellman equation:

$$\phi_1^{(k+1)} \leftarrow \arg\min_{\phi} E_{\pi_{\theta_1^{(k)}}}\left[\left(r_1(s,a) + \gamma_1 V_{\phi_1^{(k)}}(s') - V_\phi(s)\right)^2\right]. \tag{4}$$

The principle of updating the actor is two-fold: (i) maximizing the advantage; (ii) restricting the policy to the domain that is not far from other policies. The optimization is formalized below:

$$\begin{aligned} \max_{\pi} \quad & E_\pi[A_1^{(k)}] \\ \text{s.t.} \quad & D_{KL}(\pi||\pi_{\theta_i}) \leq \epsilon_i, \quad i = 2, \ldots, m, \\ \text{where} \quad & A_1^{(k)} = r_1(s,a) + \gamma_1 V_{\phi_1^{(k)}}(s') - V_{\phi_1^{(k)}}(s). \end{aligned} \tag{5}$$

We get the closed form solution of the Lagrangian of Eq. (5) in the following theorem. We omit the proof due to lack of space, please refer to Appendix A.

Theorem 1. *The Lagrangian of Eq. (5) has the closed form solution*

$$\pi^*(a|s) \propto \prod_{i=2}^{m} \left(\pi_{\theta_i}(a|s)\right)^{\frac{\lambda_i}{\sum_{j=2}^m \lambda_j}} \exp\left(\frac{A_1^{(k)}}{\sum_{j=2}^m \lambda_j}\right), \tag{6}$$

where λ_i with $i = 2, \ldots, m$ are Lagrangian multipliers.

Given data collected by $\pi_{\theta_1^{(k)}}$, we learn the policy π_{θ_1} by minimizing its KL divergence from the optimal policy π^*:

$$\begin{aligned} \theta_1^{(k+1)} &\leftarrow \arg\min_{\theta} E_{\pi_{\theta_1^{(k)}}}\left[D_{KL}(\pi^*(a|s)||\pi_\theta(a|s))\right] \\ &= \arg\max_{\theta} E_{\pi_{\theta_1^{(k)}}}\Big[\frac{\prod_{i=2}^m \left(\pi_{\theta_i}(a|s)\right)^{\frac{\lambda_i}{\sum_{j=2}^m \lambda_j}}}{\pi_{\theta_1^{(k)}}(a|s)} \exp\left(\frac{A_1^{(k)}}{\sum_{j=2}^m \lambda_j}\right) \log \pi_\theta(a|s)\Big]. \end{aligned} \tag{7}$$

The procedure of the two-stage constrained actor-critic algorithm is shown in Appendix B, and we name it as TSCAC for short. We here provide some intuition behind actor updating in (7). The term $\pi_{\theta_i}(a|s)$ denotes the probability the action selected by policy i and serves as an importance, which softly regularizes the learned policy π_{θ_1} to be close to other policies π_{θ_i}. Smaller Lagrangian multipliers λ_i indicate weaker constraints, and when $\lambda_i = 0$, we allow the learned policy π_{θ_1} to be irrelevant of the constraint policy π_{θ_i}. Note that we set the value of λ to be the same, which is more practical for the production system. The performance of TSCAC would be better if we fine-tune it with different Lagrangian multiplier value. But the effectiveness of TSCAC with the same value of λ is validated in both offline and live experiments, as we will see in following sections.

Offline Learning We now discuss adapting our constrained actor-critic method to the offline setting, i.e., a fixed dataset. The main change when moving from the online learning to the offline learning is the bias correction on the policy gradient. The actor is no longer updated on data collected by current policy but by another behavior policy π_β, which may result in a different data distribution induced by the policy being updated. To address the distribution mismatch when estimating the policy gradient, a common strategy is to apply bias-correction ratio via importance sampling [Precup 2000; Precup et al. 2001]. Given a trajectory $\tau = (s_1, a_1, s_2, a_2, \dots)$, the bias-correction ratio on the policy gradient for policy π_{θ_i} is $w(s_t, a_t) = \prod_{t'=1}^{t} \frac{\pi_{\theta_i}(s_{t'}|a_{t'})}{\pi_\beta(s_{t'}|a_{t'})}$, which gives an unbiased estimation, but the variance can be huge. Therefore, we suggest using a first-order approximation, and using the current action-selection ratio when optimizing the actors of auxiliary responses,

$$\theta_i^{(k+1)} \leftarrow \arg\max_\theta E_{\pi_\beta}\left[\frac{\pi_{\theta_i^{(k)}}(a|s)}{\pi_\beta(a|s)} A_i^{(k)} \log(\pi_\theta(a|s))\right]. \tag{8}$$

When updating the actor of the main response, we have

$$\theta_1^{(k+1)} \leftarrow \arg\max_\theta E_{\pi_\beta}\Bigg[\frac{\prod_{i=2}^{m}\left(\pi_{\theta_i}(a|s)\right)^{\frac{\lambda_i}{\sum_{j=2}^{m}\lambda_j}}}{\pi_\beta(a|s)} \times \exp\left(\frac{A_1^{(k)}}{\sum_{j=2}^{m}\lambda_j}\right)\log(\pi_\theta(a|s))\Bigg]. \tag{9}$$

Deterministic Policies We now discuss the extension of TSCAC to deterministic policies[Lillicrap et al. 2015], inspired by the updating rule for the actor of constrained policy discussed in (7). Similarly, at stage one, for each auxiliary response i, we learn separate critic models $Q_{\phi_i}(s, a)$ and actor models $\pi_{\theta_i}(s)$. At stage two, for the main response, we learn critic $Q_{\phi_1}(s, a)$ via temporal learning, and for actor $\pi_{\theta_1}(s)$, the updating rule follows the form:

$$\max_\theta \quad \prod_{i=2}^{m}\left(h(\pi_{\theta_i}(s), \pi_{\theta_1}(s))\right)^{\lambda_i} Q_{\phi_1}(s, \pi_\theta(s)), \tag{10}$$

where $h(a_1, a_2)$ scores high when two actions a_1, a_2 are close to each other and scores low vice versa, and $h(\pi_{\theta_i}(s), \pi_{\theta_1}(s))$ scores high when the actions selected by policy π_{θ_1} and π_{θ_i} are close. $\lambda_i \geq 0$ plays a similar role as the constraint Lagrangian multiplier—larger λ_i denotes stronger constraint. As an example, given n dimensional action space, one can choose $h(a_1, a_2) = \sum_{d=1}^{n} \exp\left(-\frac{(a_{1d}-a_{2d})^2}{2}\right)$. The deterministic version of TSCAC can apply to the setting with continuous actions, such as the embedding of the user preference.

Table 1: The statistics of KuaiRand.

Dimension	Number	Sparse Ratio
users	26858	-
items	10,221,515	-
samples	68,148,288	-
click	25,693,008	37.70%
like	1094434	1.61%
comment	163977	0.24%
hate	32449	0.048%

5 OFFLINE EXPERIMENTS

In this section, we evaluate our method on a public dataset about short video recommendation via extensive offline learning simulations. We demonstrate the effectiveness of our approach as compared to existing baselines in both achieving the main goal and balancing the auxiliaries. We also test the versatility of our method on another public recommendation dataset, please refer to Appendix C due to lack of space.

5.1 Setup

Dataset. We consider a public dataset for short video recommendation named *KuaiRand* (https://kuairand.com/) [Gao et al. 2022b], which is collected from a famous video-sharing mobile app and suitable for the offline evaluation of RL methods as it is unbiased. This dataset collects not only the overall `WatchTime` of the videos, but also the interaction behavior of the users including `Click`, `Like`, `Comment` and `Hate`. The statistics of the dataset are illustrated in Table 1. It shows that `Like`, `Comment`, and `Hate` are sparse signals. Note that `Hate` is extremely sparse. Logs provided by the same user are concatenated to form a trajectory; we choose top 150 videos that are most frequently viewed.

MDP.

- state s_t: A 1044 dimension vector, which is a concatenation of user features(user property), the last 20 video features viewed by the user(user history) and all the 150 candidate video features(context).
- action a_t: the video ID to be recommended currently.
- reward r_t: a vector of five scores the user provided for the viewed videos in terms of `Click`, `Like`, `Comment`, `Hate`, and `WatchTime`.
- episode: a sequence of users' video viewing history.
- discount factor γ: 0.99
- objective: We set the main goal to be maximizing the video `WatchTime`, and treat others as the auxiliaries.

Evaluation. We use the *Normalised Capped Importance Sampling* (NCIS) approach to evaluate different policies, which is a standard offline evaluation approach for RL methods in recommender systems [Zou et al. 2019]. We also evaluate our method in terms of

other metrics, please refer to Appendix D. The NCIS score is defined:

$$N(\pi) = \frac{\sum_{s,a\in D} w(s,a) r(s,a)}{\sum_{s,a\in D} w(s,a)}, w(s,a) = \min\{c, \frac{\pi(a|s)}{\pi_\beta(a|s)}\}, \quad (11)$$

where D is the dataset, $w(s,a)$ is the clipped importance sampling ratio, π_β denotes the behavior policy, c is a positive constant.

Baselines. We compare TSCAC with the following baselines.

- **BC**: A supervised behavior-cloning policy π_β to mimic the recommendation policy in the dataset, which inputs the user state and outputs the video ID.
- **Wide&Deep** [Cheng et al. 2016]: A supervised model which utilizes wide and deep layers to balance both memorization and generalization, which inputs the user state, outputs the item id, and the weight of each sample is set to be the weighted sum of all responses of this item.
- **DeepFM** [Guo et al. 2017]: a supervised recommendation model which combines deep neural network and factorization machine, which inputs the user state, outputs the item id, and the weight of each sample is set to be the weighted sum of all responses of this item.
- **RCPO** [Tessler et al. 2018] : A constrained actor-critic approach called reward-constrained policy optimization which optimizes the policy to maximize the Lagrange dual function of the constrained program. Specifically, the reward function is defined as $r = r_0 + \sum_{i=1}^{n} \lambda_i * r_i$, where r_0 is main objective, `WatchTime` and r_i denotes other feedback, and λ_i is the Lagrangian Multiplier.
- **RCPO-Multi-Critic**: We test an improved version of RCPO with multiple critics. We separately learn multiple critic models to evaluate the cumulative rewards of each feedback. Then when optimizing the actor, we maximize a linear combination of critics, weighted by the Lagrangian multipliers.
- **Pareto** [Chen et al. 2021]: A multi-objective RL algorithm that finds the Pareto optimal solution for recommender systems.
- **TSCAC**: our two-stage constrained actor-critic algorithm.

5.2 Overall Performance

Table 2 presents the performance of different algorithms in terms of five scores. We can see that our TSCAC algorithm significantly outperforms other algorithms including both constrained reinforcement learning and supervised learning methods: for the main goal (`WatchTime`), TSCAC achieves the highest performance 13.14(2.23%); for the auxiliary goal, TSCAC also ranks highest for 3 out of 4 scores (`Click`, `Like`, `Comment`). Note that TSCAC outperforms BC and RCPO at each dimension. The Pareto algorithm indeed learns a Pareto optimal solution that achieves best performance at `Hate`, but gets the lowest performance 11.90(−7.4%), i.e., it does not satisfy the setting with the main goal to optimize the `WatchTime`. The RCPO algorithm achieves the second highest performance at `WatchTime`, 13.07(1.70%), but the score at `Hate` is the worst as the sparse signals are dominated by dense signals in a single evaluation model. Compared with RCPO, RCPO-Multi-Critic achieves much better score at `Hate`, which demonstrates the effectiveness of the multi-critic policy estimation method. TSCAC also outperforms RCPO-Multi-Critic at each dimension, which shows that the ability of our two-stage actor learning method to deal with multiple responses.

5.3 Ablation Study

We investigate how the value of Lagrangian multiplier affects the performance. As we set the value of λ of all constraints to be the same in the second stage, we vary λ across $[1e-1, 1e-2, 1e-3, 1e-4, 1e-5]$ and present performance of TSCAC in terms of all responses. Recall that larger λ denotes stronger constraints of auxiliary responses. Figure 3 shows that with λ increasing, the main goal, `WatchTime` decreases as the constraints of auxiliary responses become stronger. As shown in Figure 3, the performance of interactions drops with small λ $1e-5$ as the constraints are weak. Interestingly, the performance of interactions also decreases with larger λ, which shows that too strong constraints affect the learning of the policy. The value of $1e-4$ achieves the best performance at interactions, and improve `WatchTime` significantly compared with other baselines.

6 LIVE EXPERIMENTS

To demonstrate the effectiveness of our algorithm, we test its performance as well as other alternatives via live experiments in a popular short video platform. Algorithms are embodied in a candidate-ranking system used in production at a popular short video platform, that is, when a user arrives, these algorithms are expected to rank the candidate videos, and the system will recommend the top video to the user. We show that the proposed TSCAC algorithm is able to learn a policy that maximizes the main goal while also effectively balancing the auxiliary goal, and in particular, we set the main one as maximizing the `WatchTime` and the auxiliary one as improving the interactions between users and videos.

6.1 Setup

Evaluation metrics. We use online metrics to evaluate policy performance. For the main goal, we look at the total amount of time user spend on the videos, referred to as `WatchTime`. For the auxiliary goal, users can interact with videos through multiple ways, such as sharing the video to friends, downloading it, or providing comments. Here, we focus on the three online metrics associated with the user-video interactions—the total number of `Share`, `Download`, `Comment` interactions.

MDP. Following the formulation in Section 3, we present the details of the Constrained MDP for short video recommendation.

- state s_t: user historical interactions (the list of items recommended to users at previous rounds and corresponding user feedbacks), user property (such as device and location) and the feature (the embeddings and statistics) of candidate videos at time t.
- action a_t: a vector embedding of algorithm-predicted user preferences on different video topics, which determines the actual recommendation action(the video to be recommended) via a ranking function described below:
  the ranking function: for each candidate video, this function calculates the dot product between the predicted user

Table 2: Performance of different algorithms on KuaiRand.

Algorithm	Click↑	Like↑(e-2)	Comment↑(e-3)	Hate↓(e-4)	WatchTime↑
BC	0.5338	1.231	3.225	2.304	12.85
Wide&Deep	0.5544	1.244	3.344	2.011	12.84
	3.86%	1.07%	3.69%	−12.7%	−0.08%
DeepFM	0.5549*	1.388*	3.310	2.112	12.92
	3.95%*	12.76%*	2.64%	−8.31%	0.53%
RCPO	0.5510	1.386	3.628*	2.951	13.07*
	3.23%	12.57%	12.5%*	28.1%	1.70%*
RCPO-Multi-Critic	0.5519	1.367	3.413	2.108	13.00
	3.41%	11.04%	5.83%	−8.49%	1.14%
Pareto	0.5438	1.171	3.393	0.9915*	11.90
	1.87%	−4.85%	5.22%	−56.96%*	−7.4%
TSCAC	**0.5570**	**1.462**	**3.728**	1.870	**13.14**
	4.35%	**18.80%**	**15.6%**	−18.83%	**2.23%**

The number in the bracket stands for the unit of this column; The number in the first row of each algorithm is the NCIS score.
The percentage in the second row means the performance gap between the algorithm and the BC algorithm.
The numbers with $*$ denote the best performance among all baseline methods in each response dimension.
The last row is marked by bold font when TSCAC achieves the best performance at each response dimension.

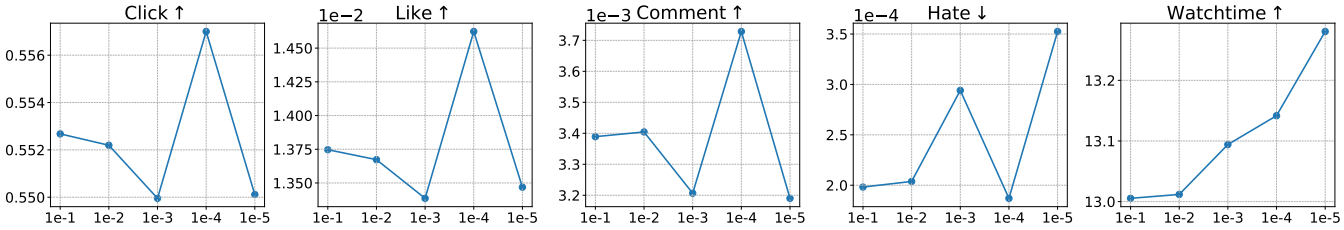


Figure 3: Effect of the value of the Lagrangian multiplier on the performance.

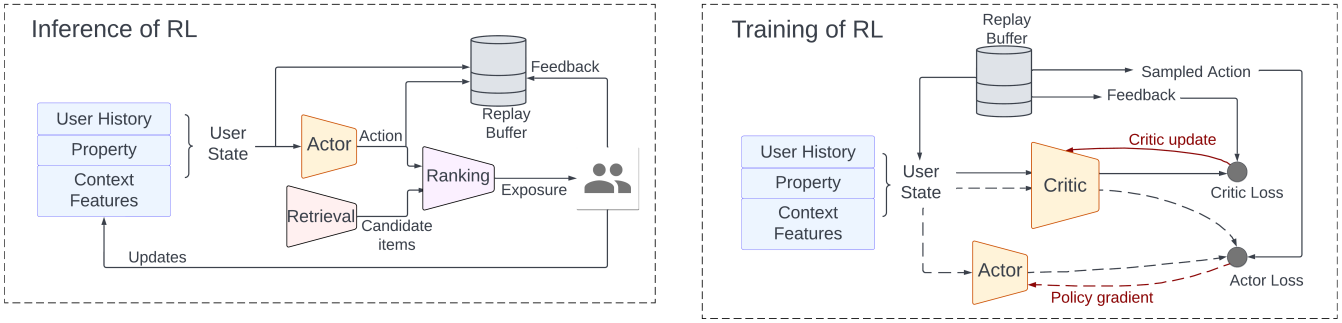


Figure 4: The workflow of RL in production system.

preference vector (a_t) and the video embedding (representing its topic and quality) as in [Dulac-Arnold et al. 2015]. Then the video with the largest score is recommended.

- reward $r_t = (l_t, i_t)$: after each recommendation, the system observes how long the user spent on the video, `WatchTime` , denoted as l_t, and whether the user has interacted with the video (`Share`/`Download`/`Comment`), denoted as i_t.
- episode: a trajectory starts when a user opens the app and ends when the user leaves.
- policy: we choose to learn a Gaussian policy in the live experiments. Specifically, the action a_t is sampled from a multivariate Gaussian distribution whose mean and variance are output of the actor model.

Workflow. As shown in Figure 4, RL runs as follows:

- **Inference** When the user comes, the user state are sent to the actor network, the actor network sample action by the Gaussian distribution. Then the ranking function inputs both

Table 3: Performance comparison of different algorithms with the LTR baseline in live experiments.

Algorithm	WatchTime	Share	Download	Comment
RCPO	+0.309%	−0.707%	0.153%	−1.313%
Interaction-AC	+0.117%	+5.008%	+1.952%	−0.101%
TSCAC	+0.379%	+3.376%	+1.733%	−0.619%

the action and the embedding of candidates, calculates the dot product between the action and the video embeddings as scores, and output the item with the highest score to the user. After that, (state, action, rewards, next state) are saved in the replay buffer.

- **Training** The actor and the critic networks are trained with a mini-batch (state, action, rewards, next state), sampled from the replay buffer.

Compared algorithms. We complement our evaluation with a supervised learning-to-rank (LTR) baseline, which is the default model run on the platform.

- **RCPO**: Following [Tessler et al. 2018], we define a combined reward $l_t + \lambda i_t$ and learn a policy to maximize the cumulative combined reward with discount factor 0.95, where λ is the Lagrangian multiplier.
- **TSCAC**: We first learn a policy π_2 to optimize the auxiliary goal. Then we learn a policy π_1 to optimize the main goal with the soft constraint that π_1 is close to π_2.
  - **Interaction-AC**: At the first stage, we learn a policy π_2 to maximize the interaction reward, with critic update following (2) and actor update following (3).
  - **TSCAC** At the second stage, we learn a main policy π_1 to maximize the cumulative reward of WatchTime and softly regularize π_1 to be close to π_2, with critic update following (4) and actor update following (7).
- **LTR (Baseline)**: The learning-to-rank model [Liu et al. 2009] that takes user state embedding and video embedding as input and fits the sum of responses.

Experimental details. To test different algorithms, we randomly split users on the platform into several buckets. The first bucket runs the baseline LTR model, and the remaining buckets run models RCPO, Interaction-AC, and TSCAC. Models are trained for a couple of days and then are fixed to test performance within one day.

6.2 Results

Table 3 shows the performance improvement of algorithm comparison with the LTR baseline regarding metrics WatchTime, Share, Download, and Comment. As we can see, RCPO can learn to improve the WatchTime as compared to the baseline; but interaction-signals are too sparse with respect to WatchTime, such that when combining these responses together, it cannot effectively balance the interaction well. Performance of the Interaction-AC algorithm is as expected: with signal from only the interaction reward, it learns to improve the interaction-related metrics (Share, Download, Comment); such interactions between users and videos also improve

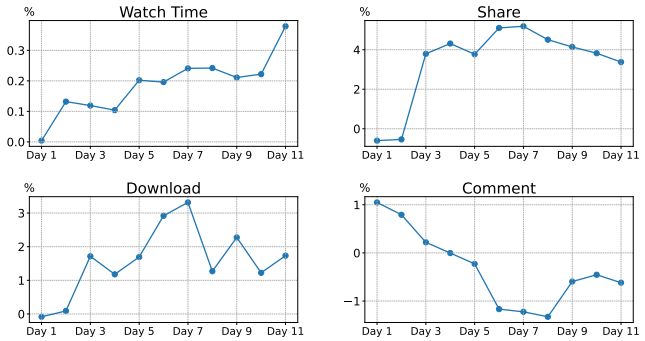


Figure 5: Online performance gap of TSCAC over the LTR baseline of each day.

the user WatchTime, since more interesting videos with high potential of invoking interactions are recommended, which optimizes user whole experience. Finally, The TSCAC algorithm achieves the best performance: as compared to RCPO, it has better WatchTime and does much better on interaction metrics, thanks to the effective softly regularization during training that it should not be too far from the Interaction-AC policy. Note that 0.1% improvement of WatchTime and 1% improvement of interactions are statistically significant in the short video platform. That is, the performance improvement of our proposed method over baselines is significant. The universal drop of Comment for all RL methods is due to the natural trade-off between WatchTime and Comment.

To understand how the TSCAC algorithm learns to balance the main and auxiliary goal, Figure 5 plots the online performance gap of the second stage over the LTR baseline on both WatchTime and interactions. As shown, the algorithm quickly learns to improve the interaction metrics Share and Comment at the beginning, with the constraint of Interaction-AC policy. Then gradually, the model learns to improve WatchTime over time with sacrificing interactions a little. Note that the live performance of TSCAC outperforms RCPO significantly at each dimension, which demonstrates the effectiveness of our method.

7 CONCLUSION

In this paper we study the problem to optimize main cumulative responses with multiple auxiliary sparse constraints in short video platforms. To tackle the challenge of multiple constraints, we propose a novel constrained reinforcement learning method, called TSCAC, that optimizes the main goal as well as balancing the others for short video platforms. Our method consists of multiple critic estimation and two learning stages. At stage one, for each auxiliary response, we learn a policy to optimize its cumulative reward respectively. At stage two, we learn the major policy to optimize the cumulative main response, with a soft constraint that restricts the policy to be close to policies maximized for other responses. We demonstrate the advantages of our method over existing alternatives via extensive offline evaluations as well as live experiments. For the future work, it is promising to apply our method to other recommender systems. It is also an interesting future work to study the performance of the deterministic version of TSCAC.

REFERENCES

M Mehdi Afsar, Trafford Crump, and Behrouz Far. 2021. Reinforcement learning based recommender systems: A survey. *arXiv preprint arXiv:2101.06286* (2021).

Md Hijbul Alam, Woo-Jong Ryu, and SangKeun Lee. 2016. Joint multi-grain topic sentiment: modeling semantic aspects for online reviews. *Information Sciences* 339 (2016), 206–223.

Haokun Chen, Xinyi Dai, Han Cai, Weinan Zhang, Xuejian Wang, Ruiming Tang, Yuzhou Zhang, and Yong Yu. 2019b. Large-scale interactive recommendation with tree-structured policy gradient. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 3312–3320.

Minmin Chen, Alex Beutel, Paul Covington, Sagar Jain, Francois Belletti, and Ed H Chi. 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.

Shi-Yong Chen, Yang Yu, Qing Da, Jun Tan, Hai-Kuan Huang, and Hai-Hong Tang. 2018. Stabilizing reinforcement learning in dynamic environment with application to online recommendation. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1187–1196.

Xu Chen, Yali Du, Long Xia, and Jun Wang. 2021. Reinforcement Recommendation with User Multi-aspect Preference. In *Proceedings of the Web Conference 2021*. 425–435.

Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. 2016. Wide & deep learning for recommender systems. In *Proceedings of the 1st workshop on deep learning for recommender systems*. 7–10.

Yinlam Chow, Mohammad Ghavamzadeh, Lucas Janson, and Marco Pavone. 2017. Risk-constrained reinforcement learning with percentile risk criteria. *The Journal of Machine Learning Research* 18, 1 (2017), 6070–6120.

Gal Dalal, Krishnamurthy Dvijotham, Matej Vecerik, Todd Hester, Cosmin Paduraru, and Yuval Tassa. 2018. Safe exploration in continuous action spaces. *arXiv preprint arXiv:1801.08757* (2018).

Gabriel Dulac-Arnold, Richard Evans, Hado van Hasselt, Peter Sunehag, Timothy Lillicrap, Jonathan Hunt, Timothy Mann, Theophane Weber, Thomas Degris, and Ben Coppin. 2015. Deep reinforcement learning in large discrete action spaces. *arXiv preprint arXiv:1512.07679* (2015).

Chongming Gao, Wenqiang Lei, Jiawei Chen, Shiqi Wang, Xiangnan He, Shijun Li, Biao Li, Yuan Zhang, and Peng Jiang. 2022a. CIRS: Bursting Filter Bubbles by Counterfactual Interactive Recommender System. *arXiv preprint arXiv:2204.01266* (2022).

Chongming Gao, Shijun Li, Yuan Zhang, Jiawei Chen, Biao Li, Wenqiang Lei, Peng Jiang, and Xiangnan He. 2022b. KuaiRand: An Unbiased Sequential Recommendation Dataset with Randomly Exposed Videos. In *Proceedings of the 31st ACM International Conference on Information and Knowledge Management* (Atlanta, GA, USA) *(CIKM '22)*. 5 pages. https://doi.org/10.1145/3511808.3557624

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

Yingqiang Ge, Shuchang Liu, Ruoyuan Gao, Yikun Xian, Yunqi Li, Xiangyu Zhao, Changhua Pei, Fei Sun, Junfeng Ge, Wenwu Ou, et al. 2021. Towards Long-term Fairness in Recommendation. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*. 445–453.

Yingqiang Ge, Xiaoting Zhao, Lucia Yu, Saurabh Paul, Diane Hu, Chu-Cheng Hsieh, and Yongfeng Zhang. 2022. Toward Pareto Efficient Fairness-Utility Trade-off in Recommendation through Reinforcement Learning. *arXiv preprint arXiv:2201.00140* (2022).

Xudong Gong, Qinlin Feng, Yuan Zhang, Jiangling Qin, Weijie Ding, Biao Li, and Peng Jiang. 2022. Real-time Short Video Recommendation on Mobile Devices. In *Proceedings of the 31st ACM International Conference on Information and Knowledge Management* (Atlanta, GA, USA) *(CIKM '22)*.

Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: a factorization-machine based neural network for CTR prediction. *arXiv preprint arXiv:1703.04247* (2017).

Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. 2015. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971* (2015).

Xiao Lin, Hongjie Chen, Changhua Pei, Fei Sun, Xuanji Xiao, Hanxiao Sun, Yongfeng Zhang, Wenwu Ou, and Peng Jiang. 2019. A pareto-efficient algorithm for multiple objective optimization in e-commerce recommendation. In *Proceedings of the 13th ACM Conference on recommender systems*. 20–28.

Zihan Lin, Hui Wang, Jingshu Mao, Wayne Xin Zhao, Cheng Wang, Peng Jiang, and Ji-Rong Wen. 2022. Feature-aware Diversified Re-ranking with Disentangled Representations for Relevant Recommendation. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 3327–3335.

Dong Liu and Chenyang Yang. 2019. A deep reinforcement learning approach to proactive content pushing and recommendation for mobile users. *IEEE Access* 7 (2019), 83120–83136.

Tie-Yan Liu et al. 2009. Learning to rank for information retrieval. *Foundations and Trends® in Information Retrieval* 3, 3 (2009), 225–331.

Yongshuai Liu, Avishai Halev, and Xin Liu. 2021. Policy learning with constraints in model-free reinforcement learning: A survey. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*.

Jiaqi Ma, Zhe Zhao, Xinyang Yi, Ji Yang, Minmin Chen, Jiaxi Tang, Lichan Hong, and Ed H Chi. 2020. Off-policy learning in two-stage recommender systems. In *Proceedings of The Web Conference 2020*. 463–473.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. 2016. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*. PMLR, 1928–1937.

Hossam Mossalam, Yannis M Assael, Diederik M Roijers, and Shimon Whiteson. 2016. Multi-objective deep reinforcement learning. *arXiv preprint arXiv:1610.02707* (2016).

Ashvin Nair, Murtaza Dalal, Abhishek Gupta, and Sergey Levine. 2020. AWAC: Accelerating Online Reinforcement Learning with Offline Datasets. (2020).

Shamim Nemati, Mohammad M Ghassemi, and Gari D Clifford. 2016. Optimal medication dosing from suboptimal clinical examples: A deep reinforcement learning approach. In *2016 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. IEEE, 2978–2981.

Thanh Thi Nguyen, Ngoc Duy Nguyen, Peter Vamplew, Saeid Nahavandi, Richard Dazeley, and Chee Peng Lim. 2020. A multi-objective deep reinforcement learning framework. *Engineering Applications of Artificial Intelligence* 96 (2020), 103915.

Qi Pi, Guorui Zhou, Yujing Zhang, Zhe Wang, Lejian Ren, Ying Fan, Xiaoqiang Zhu, and Kun Gai. 2020. Search-based user interest modeling with lifelong sequential behavior data for click-through rate prediction. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 2685–2692.

Doina Precup. 2000. Eligibility traces for off-policy policy evaluation. *Computer Science Department Faculty Publication Series* (2000), 80.

Doina Precup, Richard S Sutton, and Sanjoy Dasgupta. 2001. Off-policy temporal-difference learning with function approximation. In *ICML*. 417–424.

Ozan Sener and Vladlen Koltun. 2018. Multi-task learning as multi-objective optimization. *arXiv preprint arXiv:1810.04650* (2018).

Dusan Stamenkovic, Alexandros Karatzoglou, Ioannis Arapakis, Xin Xin, and Kleomenis Katevas. 2021. Choosing the Best of Both Worlds: Diverse and Novel Recommendations through Multi-Objective Reinforcement Learning. *arXiv preprint arXiv:2110.15097* (2021).

Richard S Sutton and Andrew G Barto. 2018. *Reinforcement learning: An introduction*. MIT press.

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

Jiayin Wang, Weizhi Ma, Jiayu Li, Hongyu Lu, Min Zhang, Biao Li, Yiqun Liu, Peng Jiang, and Shaoping Ma. 2022a. Make Fairness More Fair: Fair Item Utility Estimation and Exposure Re-Distribution. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 1868–1877.

Yuyan Wang, Mohit Sharma, Can Xu, Sriraj Badam, Qian Sun, Lee Richardson, Lisa Chung, Ed H Chi, and Minmin Chen. 2022b. Surrogate for Long-Term User Experience in Recommender Systems. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 4100–4109.

C Ch White, CC III WHITE, and KIM KW. 1980. Solution procedures for vector criterion Markov decision processes. (1980).

Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Reinforcement learning* (1992), 5–32.

Yikun Xian, Zuohui Fu, Shan Muthukrishnan, Gerard De Melo, and Yongfeng Zhang. 2019. Reinforcement knowledge graph reasoning for explainable recommendation. In *Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval*. 285–294.

Xin Xin, Alexandros Karatzoglou, Ioannis Arapakis, and Joemon M Jose. 2022. Supervised Advantage Actor-Critic for Recommender Systems. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*. 1186–1196.

Ruohan Zhan, Changhua Pei, Qiang Su, Jianfeng Wen, Xueliang Wang, Guanyu Mu, Dong Zheng, Peng Jiang, and Kun Gai. 2022. Deconfounding Duration Bias in Watch-time Prediction for Video Recommendation. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 4472–4481.

Xiangyu Zhao, Liang Zhang, Zhuoye Ding, Long Xia, Jiliang Tang, and Dawei Yin. 2018. 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.

Xiangyu Zhao, Liang Zhang, Long Xia, Zhuoye Ding, Dawei Yin, and Jiliang Tang. 2017. Deep reinforcement learning for list-wise recommendations. *arXiv preprint arXiv:1801.00209* (2017).

Lixin Zou, Long Xia, Zhuoye Ding, Jiaxing Song, Weidong Liu, and Dawei Yin. 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.

A PROOF OF THEOREM 1

Theorem 2. *The Lagrangian of Eq. (5) has the closed form solution*

$$\pi^*(a|s) \propto \prod_{i=2}^{m} \left(\pi_{\theta_i}(a|s)\right)^{\frac{\lambda_i}{\sum_{j=2}^{m}\lambda_j}} \exp\left(\frac{A_1^{(k)}}{\sum_{j=2}^{m}\lambda_j}\right), \tag{12}$$

where λ_i with $i = 2, \ldots, m$ are Lagrangian multipliers.

Proof. The Lagrangian of Eq. (5) is

$$\mathcal{L}(\pi, \lambda_2, \ldots, \lambda_m) = -E_\pi[A_1^{(k)}] + \sum_{i=2}^{m} \lambda_i \big(D_{KL}(\pi||\pi_{\theta_i}) - \epsilon_i\big). \tag{13}$$

Compute the gradient of $\mathcal{L}(\pi, \lambda_2, \ldots, \lambda_m)$ with respect to π,

$$\frac{\partial \mathcal{L}}{\partial \pi} = -A_1^{(k)} + \sum_{i=2}^{m} \lambda_i \big(1 + \log(\pi(a|s)) - \log(\pi_{\theta_i}(a|s))\big). \tag{14}$$

Setting $\frac{\partial \mathcal{L}}{\partial \pi} = 0$, we have the solution π^* satisfies

$$\pi^*(a|s) = \frac{1}{Z(s)} \prod_{i=2}^{m} \left(\pi_{\theta_i}(a|s)\right)^{\frac{\lambda_i}{\sum_{j=2}^{m}\lambda_j}} \exp\left(\frac{A_1^{(k)}}{\sum_{j=2}^{m}\lambda_j}\right), \tag{15}$$

where $Z(s)$ is the partition function to such that $\int_a \pi^*(a|s) = 1$. □

B THE TWO-STAGE CONSTRAINED ACTOR-CRITIC ALGORITHM

Algorithm 1: Two-Stage Constrained Actor-Critic (TSCAC)

Stage One: For each auxiliary response $i = 2, \ldots, m$, learn a policy to optimize the response i, with π_{θ_i} denoting actor and V_{ϕ_i} for critic.

While not converged, at iteration k:

$$\phi_i^{(k+1)} \leftarrow \arg\min_{\phi} E_{\pi_{\theta_i^{(k)}}} \Big[\big(r_i(s,a) + \gamma_i V_{\phi_i^{(k)}}(s') - V_\phi(s)\big)^2\Big],$$

$$\theta_i^{(k+1)} \leftarrow \arg\max_{\theta} E_{\pi_{\theta_i^{(k)}}} \Big[A_i^{(k)} \log\big(\pi_\theta(a|s)\big)\Big].$$

Stage Two: For the main response, learn a policy to both optimize the main response and restrict its domain close to the policies $\{\pi_{\theta_i}\}_{i=2}^m$ of auxiliary responses, with π_{θ_1} denoting actor and V_{ϕ_1} for critic.

While not converged, at iteration k:

$$\phi_1^{(k+1)} \leftarrow \arg\min_{\phi} E_{\pi_{\theta_1^{(k)}}} \Big[\big(r_1(s,a) + \gamma_1 V_{\phi_1^{(k)}}(s') - V_\phi(s)\big)^2\Big],$$

$$\theta_1^{(k+1)} \leftarrow \arg\max_{\theta} E_{\pi_{\theta_1^{(k)}}} \Big[\frac{\prod_{i=2}^{m}\left(\pi_{\theta_i}(a|s)\right)^{\frac{\lambda_i}{\sum_{j=2}^{m}\lambda_j}}}{\pi_{\theta_1^{(k)}}(a|s)} \times \exp\left(\frac{A_1^{(k)}}{\sum_{j=2}^{m}\lambda_j}\right) \log \pi_\theta(a|s)\Big].$$

Output: the constrained policy π_1.

C OFFLINE EXPERIMENTS ON TRIPADVISOR

Our code is refered to https://anonymous.4open.science/r/www23-TSCAC-2D2D/.

In this section, we evaluate our approach on another public dataset via extensive offline learning simulations. We demonstrate the effectiveness of our approach as compared to existing baselines in both achieving the main goal and balancing the auxiliaries.

Dataset. We consider a hotel-review dataset named *TripAdvisor*, which is a standard dataset for studying policy optimization in recommender system with multiple responses in [Chen et al. 2021]. In this data, customers not only provide an *overall* rating for hotels but also score hotels in multiple aspects including *service*, *business*, *cleanliness*, *check-in*, *value*, *rooms*, and *location* [Alam et al. 2016]. [1] Reviews provided by the same user are concatenated chronologically to form a trajectory; we filter trajectories with length smaller than 20. In total, we have 20277 customers, 150 hotels, and 257932 reviews.

MDP. A trajectory tracks a customer hotel-reviewing history. For each review, we have state s_t: customer ID and the last three reviewed hotel IDs as well as corresponding multi-aspect review scores; action a_t: currently reviewed hotel ID; reward r_t: a vector of eight scores the customer provided for the reviewed hotel in terms of *service*, *business*, *cleanliness*, *check-in*, *value*, *rooms*, *location*, and *overall rating*; discount factor γ: 0.99. We set the main goal to be maximizing the cumulative overall rating, and treat others as the auxiliaries.

Performance. Table 4 presents the results of different algorithms in terms of eight scores. We can see that our TSCAC algorithm performs the best among all algorithms: for the main goal, TSCAC achieves the highest overall rating 3.99; for the auxiliary goal, TSCAC also ranks highest for 5 out of 7 scores (service, cleanliness, value, rooms, location). The Pareto algorithm indeed learns a Pareto optimal solution that achieves best performance on the check-in score and business score, which however does not satisfy the setting here with the main goal to optimize the overall rating. RCPO-Multi-Critic outperforms RCPO in terms of 6 scores, which validates the effect of multi-critic estimation compared with joint-critic estimation. The RCPO-Multi-Critic algorithm achieves the same best overall score as our approach, but they sacrifice much on the others, and in particular, the location score is even lower than that from the BC algorithm.

D OFFLINE EVALUATION IN TERMS OF OTHER METRICS

We also evaluate the performance of TSCAC and baseline methods in terms of the Discounted Cumulative Gain (DCG) measure, as shown in Table 5.

[1] The dataset consists of both the main objective and other responses, which can also be used to evaluate constrained policy optimization in recommender system.

Table 4: Performance of different algorithms on TripAdvisor

Algorithms	Service	Business	Cleanliness	Check-in	Value	Rooms	Location	Overall Rating
BC	3.38	-1.86	3.57	-0.73	3.32	2.92	2.93*	3.92
Wide&Deep	3.41*	-1.86	3.62*	-0.75	3.36*	2.96	2.88	3.98
RCPO	3.40	-1.82	3.61	-0.71	3.34	2.95	2.91	3.97
RCPO-Multi-Critic	3.41*	-1.82	3.62*	-0.68	3.35	2.97*	2.87	3.99*
Pareto	3.36	−1.79*	3.57	−0.62*	3.29	2.93	2.86	3.95
TSCAC	**3.43**	-1.82	**3.64**	-0.68	**3.37**	**3.00**	**2.98**	**3.99**

The results with ∗ denote the best performance among all baseline methods in each response dimension
the data in last row is marked by bold font when TSCAC achieves the best performance.

Table 5: Performance of different algorithms in terms of DCG.

Algorithm	Click↑ (e+2)	Like↑	Comment↑	Hate↓(e-2)	WatchTime↑(e+3)
BC	4.617	8.492	2.320	8.137	1.079
Wide&Deep	4.576	8.158	2.274	7.503	1.073
	−0.88%	−3.93%	−1.97%	−7.79%*	−0.56%
DeepFM	4.594	8.439	2.274	8.334	1.083
	−0.50%	−0.63%	−1.97%	2.43%	0.41%*
RCPO	4.603	8.537	2.282	8.336	1.080
	−0.30%	0.53%*	−1.60%	2.45%	0.12%
RCPO-Multi-Critic	4.569	8.168	2.272	7.538	1.069
	−1.03%	−3.81%	−2.09%	−7.36%	−0.88%
Pareto	4.605	8.494	2.304	8.449	1.074
	−0.26%*	0.02%	−0.69%*	3.83%	−0.48%
TSCAC	**4.617**	**8.652**	**2.378**	8.036	1.083
	0.01%	**1.88%**	**2.49%**	−1.24%	0.39%

↑: higher is better; ↓: lower is better.
The number in the bracket stands for the unit of this column; The number in the first row of each algorithm is the DCG score.
The percentage in the second row means the performance gap between the algorithm and the BC algorithm.
The numbers with ∗ denote the best performance among all baseline methods in each response dimension.
The last row is marked by bold font when TSCAC achieves the best performance at each response dimension.