Multi-Preview Recommendation via Reinforcement Learning

Yang Xu
Department of Statistics, North Carolina State University
Raleigh, North Carolina, USA
yxu63@ncsu.edu

Kuan-Ting Lai
Microsoft
Redmond, Washington, USA
kuantinglai@microsoft.com

Pengcheng Xiong
Microsoft
Redmond, Washington, USA
peterxiong@microsoft.com

Zhong Wu
Microsoft
Redmond, Washington, USA
zhonwu@microsoft.com

ABSTRACT

Preview recommendations serve as a crucial shortcut for attracting users' attention on various systems, platforms, and webpages, significantly boosting user engagement. However, the variability of preview types and the flexibility of preview duration make it challenging to use an integrated framework for multi-preview recommendations under resource constraints. In this paper, we present an approach that incorporates constrained Q-learning into a notification recommendation system, effectively handling both multi-preview ranking and duration orchestration by targeting long-term user retention. Our method bridges the gap between combinatorial reinforcement learning, which often remains too theoretical for practical use, and segmented modules in production, where model performance is typically compromised due to over-simplification. We demonstrate the superiority of our approach through off-policy evaluation and online A/B testing using Microsoft data.

CCS CONCEPTS

• **Information systems** → **Web interfaces**; **Data analytics**; • **Computing methodologies** → **Model verification and validation**; • **Mathematics of computing** → *Mathematical analysis.*

KEYWORDS

Notification Recommendation System, Constrained Q-learning, Multi-Preview Ranking, Duration, Combinatorial Reinforcement Learning

ACM Reference Format:
Yang Xu, Kuan-Ting Lai, Pengcheng Xiong, and Zhong Wu. 2024. Multi-Preview Recommendation via Reinforcement Learning. In *18th ACM Conference on Recommender Systems (RecSys '24), October 14–18, 2024, Bari, Italy.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3640457.3691698

1 INTRODUCTION

On Windows systems, a preview tab is usually displayed in the bottom left corner, providing users with a brief overview of the latest news and notifications before they click for more details. The system regularly aggregates incoming notifications from various sources, typically every 30 minutes (with possible variations for different users). It then selects a subset of these previews and determines their display duration. Due to time constraints, not all candidate previews can be shown to each user. Therefore, optimizing the selection and duration of these previews is essential to maximize the effectiveness of the recommendation system.

Reinforcement Learning (RL) is a powerful tool for learning from past experiences and continuously updating actions to optimize long-term rewards. While it has been extensively studied in academic research, its adoption in industrial applications remains relatively limited. In recommendation systems involving multiple previews, the decision-making process has an action space with two dimensions: (1) selecting K previews with the highest potential from N candidates, and (2) determining the duration for each preview type. This problem, often known as combinatorial RL, has garnered increasing attention from researchers in recent years [2, 4, 7, 10, 12]. However, the complexity of action space makes it challenging to implement at a production level.

In industry, complex decision-making problems involving multi-dimensional action manipulation are often decomposed into several simpler, more robust steps, each supported by a separate model. These steps typically include: (1) ranking, where all incoming previews are rated by a model based on their potential for each user, and (2) duration time selection, where durations are chosen from a set of possible values $\{0, 1, 2, \dots, S\}$ minutes. Each step uses either a separate modeling or rule-based algorithm to support personalized preview recommendations.

Given the significant differences in handling complex sequential decision-making problems with an intricate action space, we aim to propose a more integrated framework that bridges the gap between theoretical research and real-world applications, so as to improve overall user engagement and long-term retention. Our proposed method extends classical Q-learning to handle both ranking and duration time selection simultaneously. This approach allows us to address complex decision-making problems within an integrated RL framework in a more implementable way, while being more motivated by long-term rewards than traditional multi-step decomposition methods used in the industry.

2 PROBLEM FORMULATION

At each round t (i.e. every 30 minutes), a total of $\mathcal{P}$ incoming previews are available for selection. We denote the user information available at this time as s_t. For simplicity, we discretize

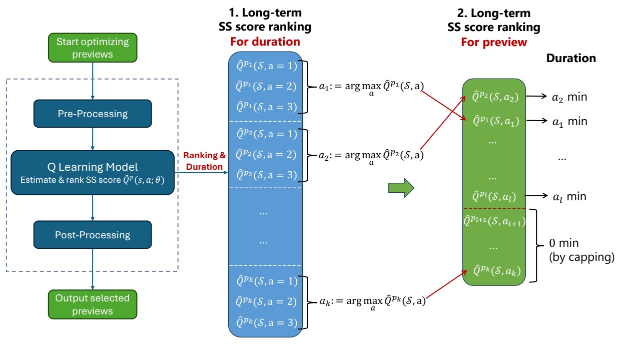

Start optimizing previews
Pre-Processing
Q Learning Model
Estimate & rank SS score $\hat{Q}^{p}(s,a;\theta)$
Post-Processing
Output selected previews
Ranking & Duration
1. Long-term SS score ranking
For duration
$\hat{Q}^{p_1}(\mathcal{S}, a=1)$
$\hat{Q}^{p_1}(\mathcal{S}, a=2)$
$\hat{Q}^{p_1}(\mathcal{S}, a=3)$
$a_1 := \arg\max_a \hat{Q}^{p_1}(\mathcal{S}, a)$
$\hat{Q}^{p_2}(\mathcal{S}, a=1)$
$\hat{Q}^{p_2}(\mathcal{S}, a=2)$
$\hat{Q}^{p_2}(\mathcal{S}, a=3)$
$a_2 := \arg\max_a \hat{Q}^{p_2}(\mathcal{S}, a)$
$\hat{Q}^{p_k}(\mathcal{S}, a=1)$
$\hat{Q}^{p_k}(\mathcal{S}, a=2)$
$\hat{Q}^{p_k}(\mathcal{S}, a=3)$
$a_k := \arg\max_a \hat{Q}^{p_k}(\mathcal{S}, a)$
2. Long-term SS score ranking
For preview
Duration
$\hat{Q}^{p_2}(\mathcal{S}, a_2)$
$\hat{Q}^{p_1}(\mathcal{S}, a_1)$
$\hat{Q}^{p_l}(\mathcal{S}, a_l)$
$\hat{Q}^{p_{l+1}}(\mathcal{S}, a_{l+1})$
$\hat{Q}^{p_k}(\mathcal{S}, a_k)$
a_2 min
a_1 min
a_l min
0 min
(by capping)

Figure 1: Flowchart of ranking and duration orchestration

the duration for which each preview can be shown into values $a_t \in \mathcal{A} = \{0, 1, \ldots, S\}$, with $S = 3$ representing the maximum number of minutes preview p will be displayed to a user characterized by the state $s_t \in \mathcal{S}$. The reward for showing preview p for a_t minutes, denoted as R_t, is determined based on multi-dimensional signals, including the user's clicks, scrolls, and dwell time on both the preview banner and detail page.

We define a policy $\pi : \mathcal{S} \longrightarrow \mathcal{A}$ as a mapping from the state space to the action space, which suggests the duration time for each user based on specific information s. Since multiple previews $p \in \mathcal{P}$ are involved at a given stage, the Q-function for preview p is defined as $Q^{\pi}(s, a, p)$, where $Q^{\pi}(s, a, p) = \mathbb{E}\big[\sum_{k=0}^{\infty} \gamma^k R_{t+k} \mid S_t = s, A_t = a, p \in \mathcal{P}\big]$. We refer to this as the long-term successful session (LTSS) score specific to each preview p, user information s, and duration time a. Thus, the aggregated Q-function is $Q^{\pi}(s, a) = \sum_{p \in \mathcal{P}} \mathbb{E}\big[\sum_{k=0}^{\infty} \gamma^k R_{t+k} \mid S_t = s, A_t = a\big]$. Similarly, we define the value function for preview p as $V^{\pi}(s, p) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k} \mid S_t = s, p \in \mathcal{P}\right]$, and the aggregated value function as $V^{\pi}(s) = \sum_{p \in \mathcal{P}} \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k} \mid S_t = s\right]$.

Our goal is to learn an optimal policy $\pi^* = \{\pi_p^*\}_{p \in \mathcal{P}}$, so as to maximize

$$\begin{aligned} \max_{\pi} \quad & \mathbb{E}_{s \in \mathcal{S}} \left[V^{\pi}(s)\right] \\ \text{s.t.} \quad & \mathbb{E}[A_t] \le C. \end{aligned} \tag{1}$$

For simplicity of implementation, we set time constraint C to be the same across all stages. There is existing work on constrained Q-learning in the current literature [1, 3, 5, 9, 11]. In this paper, we adapt the Lagrangian method to convert the constrained problem into an unconstrained optimization problem by introducing a penalty term controlled by λ. Specifically, we maximize the following objective function:

$$\max_{\pi} \quad \mathbb{E}_{s \in \mathcal{S}} \left[\sum_{p \in \mathcal{P}} \mathbb{E}\left\{ \sum_{k=0}^{\infty} \gamma^k (R_{t+k} - \lambda A_{t+k}) \mid S_t = s \right\} \right], \tag{2}$$

which effectively redefines the reward function by penalizing with $R_t - \lambda A_t$, allowing us to control the overall duration time. Here, λ represents the expected increase in the SS score conversion rate per unit of time, and can be adjusted by the user based on the specific problem context and desired level of capping. Generally, a larger λ will bias actions towards the shortest duration time, i.e., $A_t = 0$. Since the optimization problem in Equation (2) involves the reward summation across all preview types $\mathcal{P}$, in practice, model training and parameter updates can be conducted separately for each preview type to enhance training speed. This separate training procedure also provides an additional level of flexibility by allowing λ to be set differently for each preview type, enabling practitioners to tailor the importance of previews according to specific needs. The detailed approach is outlined in Algorithm 1.

Algorithm 1 Multi-Preview Optimization via Constrained RL

```
 1: Input: offline data $D = \{S_{i,t}, A_{i,t}, R_{i,t}, S_{i,t+1}\}_{1 \le i \le N, 1 \le t \le T}$, discount factor $\gamma$, initialized parameter $\theta_0$, exploration probability $\epsilon_t$, target network updating frequency $E_0$, number of epochs $E$.

 2: [Step 1: Offline Warm-Up]
 3: for each preview $p \in \mathcal{P}$ do
 4:     for epoch $e \in \{0, \ldots, E\}$ do
 5:         Sample a batch of data tuple $D_e \subset D$
 6:         For each data tuple $(s, a, r, s') \in D_e$, calculate
            $y(e) \longleftarrow r - \lambda a + \gamma \max_{a'} Q(s', a'; \theta_e)$
 7:         Calculate $Loss(\theta) = \sum_{i,t} \{y_{it}(e) - Q(s_{it}, a_{it}; \theta)\}^2$
 8:         Update $\theta_{e+1} \longleftarrow \arg\min_{\theta} Loss(\theta)$
 9:     end for
10: end for
11: [Step 2: Online Update]
12: Update the initial parameter in online testing by $\theta_0 \longleftarrow \theta_E$
13: for each stage $t \in \{0, \ldots, T\}$ do
14:     for each preview $p \in \mathcal{P}$ do
15:         Select $a_t = \arg\max_a Q(s_t, a, p; \theta_t)$ with probability $1 - \epsilon_t$
            and a random policy with probability $\epsilon_t$
16:         Interact with the environment to obtain $r_t, s_{t+1}$
17:         Update data replay buffer $D \longleftarrow D \cup (s_t, a_t, r_t, s_{t+1})$
18:         Repeat step 5-8 to update $\theta_t \longleftarrow \arg\min_{\theta} Loss(\theta)$
19:     end for
20: end for
21: return  estimated Q function parameter $\theta_T$
```

Figure 1 illustrates the use of constrained Q-learning for multi-preview ranking, duration orchestration, and capping. At each stage t, we start with incoming previews $p \in \mathcal{P}$, estimate the long-term SS score $\hat{Q}(s, a, p; \theta_t)$, and proceed as follows:

a. Step 1: For each preview $p_k \in \mathcal{P}$, determine the optimal duration $a_k := \arg\max_{a \in \mathcal{A}} \hat{Q}(s, a, p_k; \theta_t)$.
b. Step 2: Rank previews by their long-term SS scores under the chosen durations a_k. Previews are then presented to users based on these rankings until a capping threshold is reached.

Within each Q-learning module, we update θ_t as described in Algorithm 1. The process is divided into two phases: (1) using offline data to initially train a constrained Q-learning model; and (2) using the pre-trained model, with parameters initialized to $\theta_0 \longleftarrow \theta_E$, for online testing. If no offline data is available, training can begin directly online with a pre-selected parameter θ_0.

3 OFFLINE EVALUATION

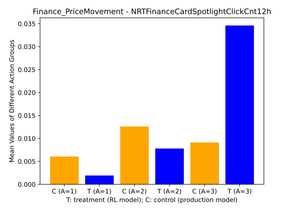
To evaluate performance, we first conduct Off-Policy Evaluation (OPE), then online A/B testing to compare the existing production model, represented by the behavior policy b, with the constrained RL model obtained in Algorithm 1, represented by the estimated optimal policy π.

In OPE, we first learn the behavior policy $\hat{b}(s|a,p)$ for each preview type p from historical data to estimate the value function under existing production model. We then use fitted-Q evaluation [6, 8] to estimate $\hat{Q}^b(s,a,p)$. Table 1 compares $\sum_{s\in\mathcal{S}} \hat{V}(s,p)$ across four popular preview types. By comparison, Algorithm 1 demonstrates a long-term successful session (LTSS) gain for all preview types trained through offline learning, highlighting its effectiveness in optimizing long-term rewards under specific resource constraints.

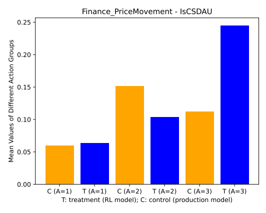
Preview Type	$LTSS^C$	$LTSS^T$	$LTSS$ gain
$Finance\_DefaultIndex$	0.7258	0.7536	3.83%
$Finance\_PriceMovement$	1.1407	1.1734	2.87%
$Sports\_SportsMatch$	0.5561	0.6083	9.39%
$Weather\_TeaserTempRecord$	0.4124	0.4737	14.86%

Table 1: The comparison of of LTSS in production (group C) and constrained Q-learning (group T).

To further illustrate the shift in user cohort from the production model to the RL model, we examine the $Finance\_PriceMovement$ preview type, as shown in Figure 2. In the left bar plot, yellow bars represent the mean click counts for finance cards in the existing production model, while blue bars represent the mean click counts in the constrained RL model. Notably, in the last two columns, where the duration time $A = 3$, the RL model assigns users with higher finance card click counts to longer duration time buckets. Similarly, the right bar plot indicates that the RL model assigns more active users to longer duration time buckets compared to the existing model. These results demonstrate the RL model's effectiveness in identifying and prioritizing the most engaging users based on their preview preferences.



(a) Finance Card Click Counts



(b) Daily Active Users

Figure 2: The bar plots of preview $Finance\_PriceMovement$

4 ONLINE A/B TESTING

In online testing, we tested the constrained RL model in two groups, T_1 with $\gamma = 0$, i.e. focusing on short-term rewards, and T_2 with $\gamma = 0.9$, which prioritize long-term rewards, and compare with existing production model with Microsoft data. Since the algorithm relies solely on a constraint-free loss function calculation and corresponding parameter updates, it is scalable and can be trained using any supervised learning models, such as ensemble methods or neural networks, depending on user preferences. Neural networks may offer more flexibility for parameter updates in batched training due to their compatibility with gradient descent. For privacy reasons, we are unable to share the code and raw data, but the summary of the online testing results is provided.

Group	DAU	DAU gain
C	0.4107	0
T_1	0.4109	0.03%
T_2	0.4151	1.05%

The result of online testing is shown in the table above. Group C serves as the control group, representing the production model with separate ranking and duration orchestration. Group T_1 uses the RL model with $\gamma = 0$ to emphasize short-term rewards, while Group T_2 employs the RL model with $\gamma = 0.9$ to prioritize long-term rewards. The results from the online A/B testing reveal a consistent increase in daily active users (DAU) when long-term rewards are prioritized with the RL model. This underscores the significance of focusing on long-term user retention to enhance daily active user volume, and demonstrates the advantages of integrating ranking and duration within the constrained RL model.

5 SUMMARY AND FUTURE WORK

In this paper, we adapt a constrained RL algorithm to handle a multi-preview recommendation system, enabling simultaneous preview selection and duration setup with the unified goal of enhancing long-term user retention. Both offline evaluation and online testing demonstrate the superiority of RL-based approaches.

There are several potential areas for future research. Although the algorithm benefits from training each constrained RL model separately for each preview type, there is potential for sharing information across different preview types. Exploring a global model that can better adapt penalty weight selection, accommodate variations in penalty λ across preview types, and facilitate information sharing through initial layers while training the RL models could be a promising direction. This approach would help capture the commonalities among different preview types, enhancing the overall effectiveness of the system.

ACKNOWLEDGMENTS

We would like to express our sincere gratitude to Microsoft for providing the data used in this research. We also extend our appreciation to Bin Bin, Xi Chen, Yongzhi Li, and Tiannan Wang for their invaluable assistance, which were crucial to this study.

REFERENCES

[1] Eitan Altman. 2021. *Constrained Markov decision processes.* Routledge.
[2] Irwan Bello, Hieu Pham, Quoc V Le, Mohammad Norouzi, and Samy Bengio. 2016. Neural combinatorial optimization with reinforcement learning. *arXiv preprint arXiv:1611.09940* (2016).
[3] Steven Bohez, Abbas Abdolmaleki, Michael Neunert, Jonas Buchli, Nicolas Heess, and Raia Hadsell. 2019. Value constrained model-free continuous control. *arXiv preprint arXiv:1902.04623* (2019).

[4] Arthur Delarue, Ross Anderson, and Christian Tjandraatmadja. 2020. Reinforcement learning with combinatorial actions: An application to vehicle routing. *Advances in Neural Information Processing Systems* 33 (2020), 609–620.

[5] Arnob Ghosh, Xingyu Zhou, and Ness Shroff. 2022. Provably efficient model-free constrained rl with linear function approximation. *Advances in Neural Information Processing Systems* 35 (2022), 13303–13315.

[6] Botao Hao, Xiang Ji, Yaqi Duan, Hao Lu, Csaba Szepesvari, and Mengdi Wang. 2021. Bootstrapping fitted q-evaluation for off-policy inference. In *International Conference on Machine Learning*. PMLR, 4074–4084.

[7] Alexandre Laterre, Yunguan Fu, Mohamed Khalil Jabri, Alain-Sam Cohen, David Kas, Karl Hajjar, Torbjorn S Dahl, Amine Kerkeni, and Karim Beguir. 2018. Ranked reward: Enabling self-play reinforcement learning for combinatorial optimization. *arXiv preprint arXiv:1807.01672* (2018).

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

[9] Yongshuai Liu, Avishai Halev, and Xin Liu. 2021. Policy learning with constraints in model-free reinforcement learning: A survey. In *The 30th international joint conference on artificial intelligence (ijcai)*.

[10] Nina Mazyavkina, Sergey Sviridov, Sergei Ivanov, and Evgeny Burnaev. 2021. Reinforcement learning for combinatorial optimization: A survey. *Computers & Operations Research* 134 (2021), 105400.

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

[12] Yunhao Yang and Andrew Whinston. 2023. A survey on reinforcement learning for combinatorial optimization. In *2023 IEEE World Conference on Applied Intelligence and Computing (AIC)*. IEEE, 131–136.

Received 31 July 2024