Thompson Sampling for a Fatigue-aware Online Recommendation System

Yunjuan Wang and Theja Tulabandhula

University of Illinois at Chicago, USA

Motivation

Platform Sends out a daily/weekly digest

User Clicks on interesting links *or* marks sender as spam

Platform Schedules a series of notifications for engagement

User Clicks on notifications and engages with app *or* mutes notifications forever

Model

- The platform recommends a (sub)-sequence $\mathbf{S}$ of items.
- User's intrinsic preference for item $j \in [N]$ is $u_j \in [0, 1]$.
- After viewing each item, the user can abandon the platform with probability $1 - q > 0$.
- If they abandon, the platform incurs a penalty $c > 0$.
- If they select j and leave, platform gets revenue $r_j > 0$.
- If they don't select j and move to the next item in $\mathbf{S}$, platform gets nothing.
- Let $\mathbf{S} = (S_1, S_2, \ldots, S_m)$, where S_k denotes item in the k^{th} position
- Let $p_i(\mathbf{S})$ denote the probability of selecting item i in sequence $\mathbf{S}$.
- Let $p_a(\mathbf{S})$ denote the probability of total abandonment.

Offline Fatigue-aware Recommendations

$$p_i(\mathbf{S}) = \begin{cases} u_i & \text{if } i \in S_1, \\ q^{l-1}\prod_{k=1}^{l-1}(1-u_{S_k})u_i & \text{if } i \in S_l, l \geq 2, \\ 0 & \text{if } i \notin \mathbf{S}. \end{cases}$$

$$p_a(\mathbf{S}) = \sum_{k=1}^{m} q^{k-1}(1-q)\prod_{j=1}^{k}\left(1-u_{S_j}\right).$$

The goal is to find the optimal sequence of items that maximizes expected utility $\mathbb{E}[U(\mathbf{S};\mathbf{u},q)] = \sum_{i\in\mathbf{S}} p_i(\mathbf{S})r_i - cp_a(\mathbf{S})$:

$$\max_{\mathbf{S}} \quad \mathbb{E}[U(\mathbf{S};\mathbf{u},q)]$$
$$\text{s.t.} \quad S_i \cap S_j = \emptyset, \forall i \neq j,$$
$$\text{and other business constraints,}$$

where $\mathbb{E}[U(\mathbf{S};\mathbf{u},q)] = \sum_{i\in\mathbf{S}} p_i(\mathbf{S})r_i - cp_a(\mathbf{S})$.

TS-based Algorithm (Algo. 1, precursor to SBORS below)

Initialization: Set $c_i(t) = f_i(t) = 1$ for all $i \in X$; $n_e(t) = n_a(t) = 1$; $t = 1$;

while $t \leq T$ **do**

(a) *Posterior sampling*:

For each item $i = 1, \ldots, N$, sample $u_i'(t)$ and $q'(t)$

$u_i'(t) \sim Beta(c_i(t), f_i(t))$, $q'(t) \sim Beta(n_e(t), n_a(t))$

(b) *Sequence selection*:

Compute $\mathbf{S}^t = \arg\max_{\mathbf{S}} \mathbb{E}[U(\mathbf{S};\mathbf{u}'(t), q'(t))]$;

Observe feedback upon seeing the $k_t \leq |\mathbf{S}^t|$ items;

(c) *Posterior update*:

for $j = 1, \cdots, k_t$ **do**

Update

$$(c_{S_j^t}(t), f_{S_j^t}(t), n_e(t), n_a(t)) = \begin{cases} (c_{S_j^t}(t)+1, f_{S_j^t}(t), n_e(t), n_a(t)) \\ \quad \text{if select and leave} \\ (c_{S_j^t}(t), f_{S_j^t}(t)+1, n_e(t)+1, n_a(t)) \\ \quad \text{if not select and not abandon} \\ (c_{S_j^t}(t), f_{S_j^t}(t)+1, n_e(t), n_a(t)+1) \\ \quad \text{if not select and abandon} \end{cases}$$

$c_i(t+1) = c_i(t), f_i(t+1) = f_i(t)$ for all $i \in [N]$

$n_e(t+1) = n_e(t), n_a(t+1) = n_a(t)$

$t = t + 1$

SBORS Algorithm (Algo. 2)

Initialization: Set $c_i(t) = f_i(t) = 1$ for all $i \in X$; $n_e(t) = n_a(t) = 1$; $t = 1$;

while $t \leq T$ **do**

Update $\hat{u}_i(t) = \frac{c_i(t)}{c_i(t)+f_i(t)} = \frac{c_i(t)}{T_i(t)}$, $\hat{\sigma}_{u_i}(t) = \sqrt{\frac{\alpha\hat{u}_i(t)(1-\hat{u}_i(t))}{T_i(t)+1}} + \sqrt{\frac{\beta}{T_i(t)}}$,

$\hat{q}(t) = \frac{n_e(t)}{n_e(t)+n_a(t)} = \frac{n_e(t)}{N_q(t)}$, $\hat{\sigma}_q(t) = \sqrt{\frac{\alpha\hat{q}(t)(1-\hat{q}(t))}{N_q(t)+1}} + \sqrt{\frac{\beta}{N_q(t)}}$.

(a) *Correlated sampling*:

for $j = 1, \ldots, R$ **do**

Get $\theta^{(j)} \sim N(0, 1)$ and compute $u_i'^{(j)}(t)$,$q'^{(j)}(t)$

For each $i \leq N$, compute $u_i'(t) = \max_{j=1,\cdots,R} u_i'^{(j)}(t)$, $q'(t) = \max_{j=1,\cdots,R} q'^{(j)}(t)$.

(b) *Sequence selection*: Same as step (b) of Algo. 1.

(c) *Posterior update*: Same as step (c) of Algo. 1.

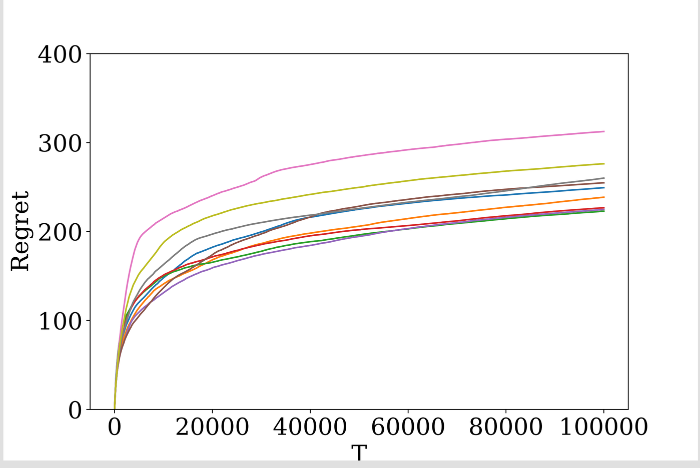
Regret Gaurantee

(**Main Result**) Over T rounds, the regret of SBORS is bounded as:

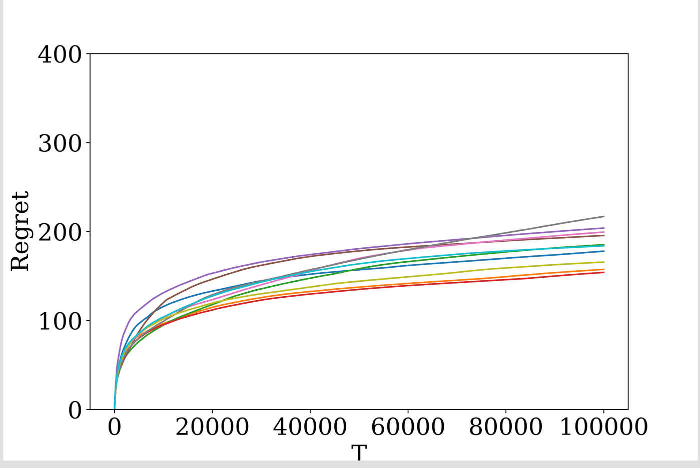
$$Reg(T;\mathbf{u},q) = \mathbb{E}\left[\sum_{t=1}^{T}\mathbb{E}[U(\mathbf{S}^*;\mathbf{u},q)] - \mathbb{E}[U(\mathbf{S}^t;\mathbf{u},q)]\right]$$
$$\leq C_1N^2\sqrt{NT\log TR} + C_2N\sqrt{T\log TR\cdot\log T} + \frac{C_3N}{R},$$

where C_1, C_2 and C_3 are constants and R is an algorithm parameter.

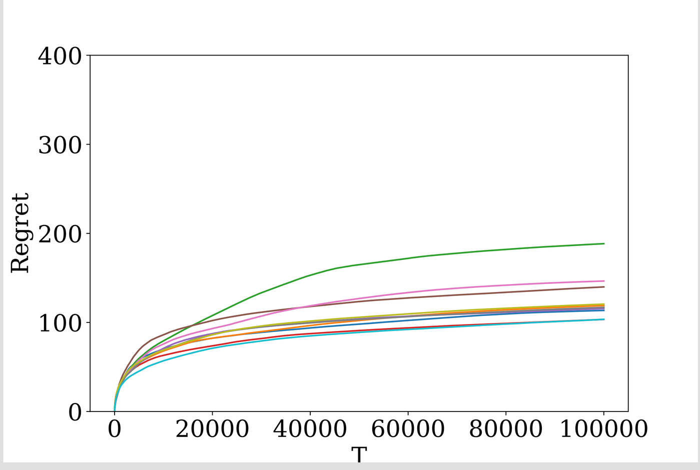
Experiments



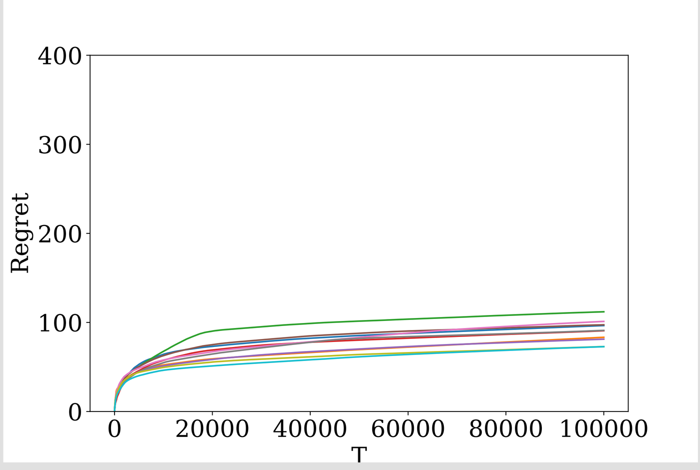
(a)



(b)

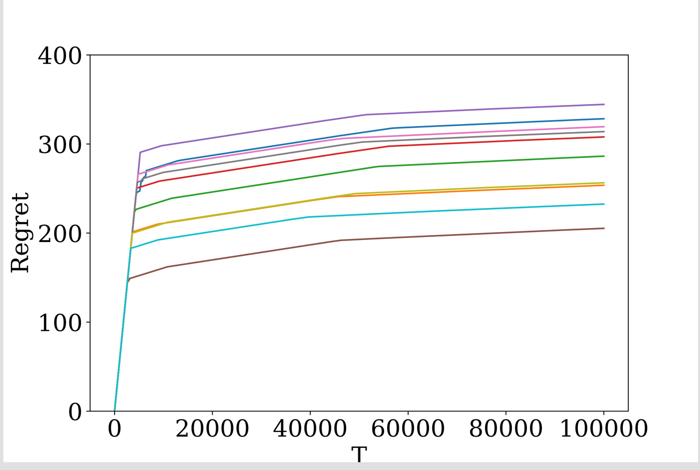


(c)

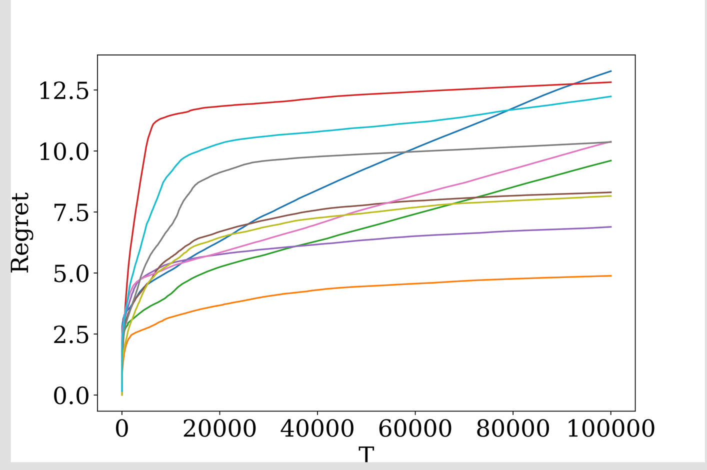


(d)

Figure: Comparison of Algorithm 1 when $\mathbf{u}$ is uniformly generated from (a) [0,0.1], (b) [0,0.2], (c) [0,0.3], and (d) [0,0.5].



(a) UCB-based algorithm



(b) Our algorithm

Figure: Comparison of UCB-based algorithm, UCB-V algorithm and Algorithm 1.

ArXiv: Yunjuan Wang and Theja Tulabandhula. Thompson Sampling for a Fatigue-aware Online Recommendation System, 2019.