中国人民大學
RENMIN UNIVERSITY OF CHINA

Towards More Realistic User Long Term Engagement Modeling in Recommender Systems

Xu Chen

CONTENTS

Background

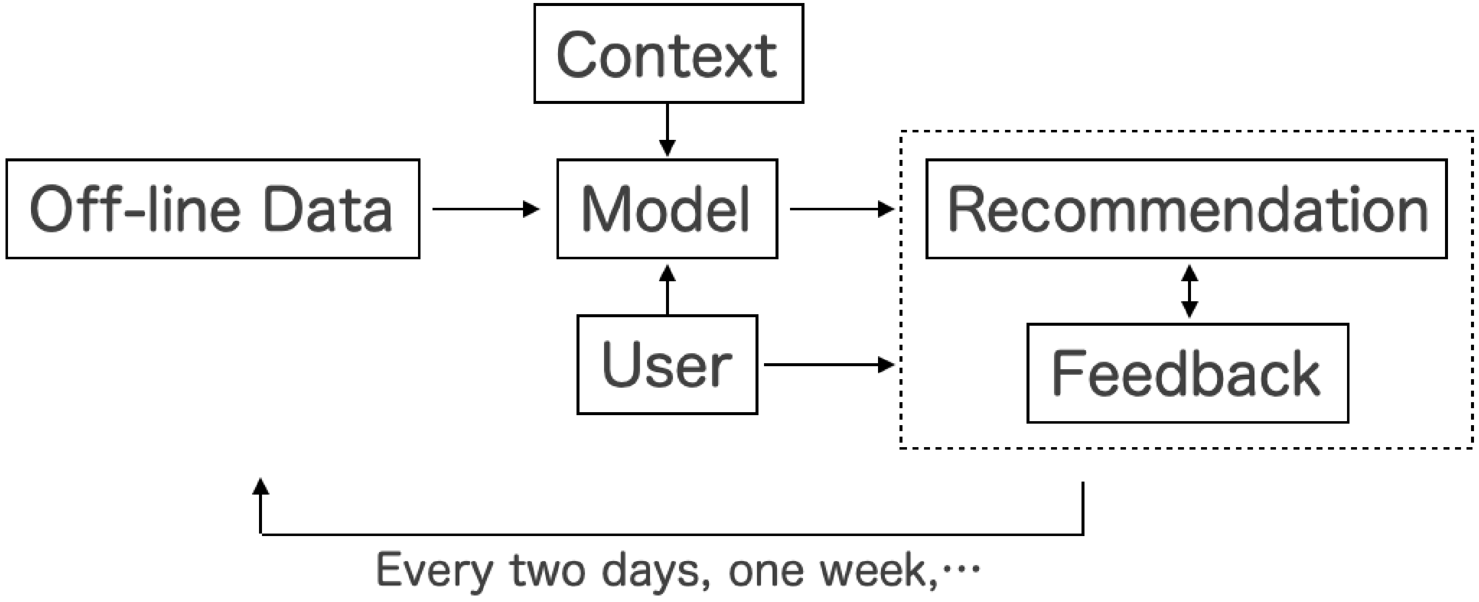
Information Overload

SIGIR 2021

头条 今日头条 YOUKU 优酷 YouTube 網易 NETEASE

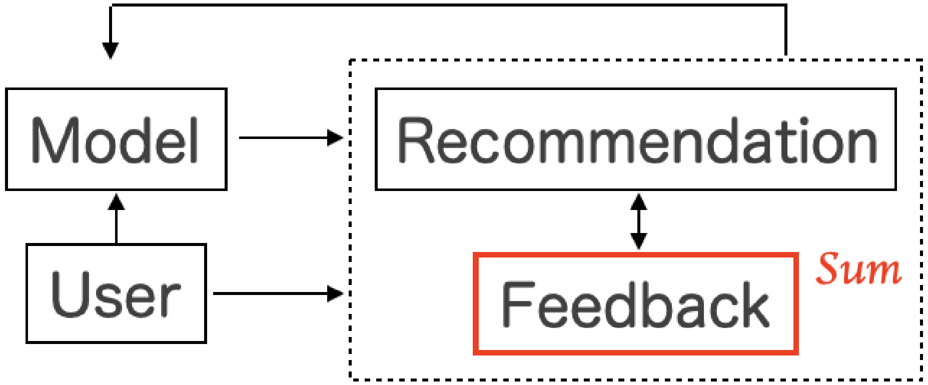
amazon 淘宝网 Taobao.com 快手 一亿人都在拼多多

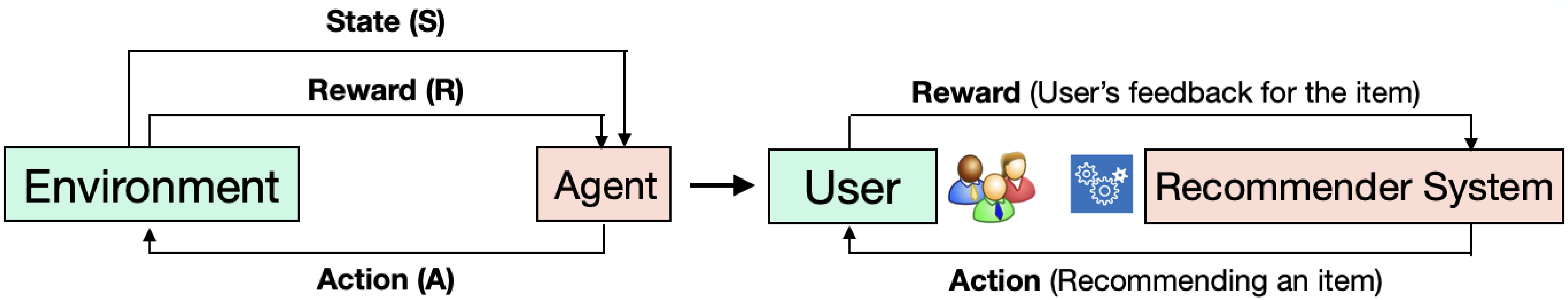
JD.COM 京东 iQIYI 爱奇艺 腾讯视频



1. Interactive Nature

2. Long-term Utilities





Recommender -> Agent

User -> Environment

Recommend an item -> Action

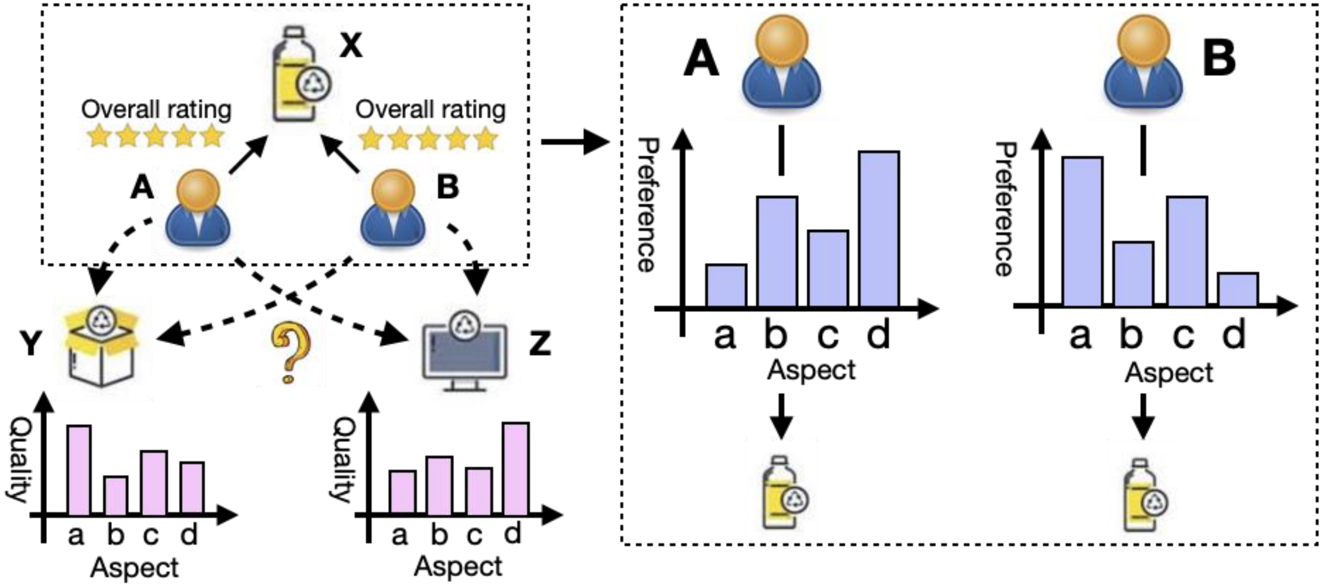
User's historical behaviour -> State

User's rating on the item -> Reward

How to consider the characters of the user preference? E.g., Diversity, Dynamic, …

Modelling User Diverse Preference

Diverse user preference



Different aspect preferences are not always aligned!

Definition 1. Pareto dominance. Suppose we have two parameters θ_A and θ_B, we say θ_A can dominant θ_B (denoted by $\theta_A \succ \theta_B$), if and only if $\mathcal{L}_i(\theta_A) \leq \mathcal{L}_i(\theta_B)$, $\forall i \in \{1, 2, \ldots M\}$ and $\mathcal{L}_i(\theta_A) < \mathcal{L}_i(\theta_B)$, $\exists i \in \{1, 2, \ldots M\}$.

Definition 2. Pareto efficiency. For a parameter θ^*, if there is no other $\hat{\theta}$, such that $\hat{\theta} \succ \theta^*$, then we say θ^* is a Pareto efficient solution.

Critic learning:

$$\arg\min_{\phi_m} \sum_{i=1}^{N} (y_{i,m} - Q_m(s_i, a_i|\phi_m))^2, \; m = 1, 2, \ldots M$$

Actor learning:

$$l(\theta) = -\sum_{m=1}^{M} w_m \sum_{i=1}^{N} Q_m(s_i, \mu(s_i|\theta))$$

$$L(\theta) = \sum_{i=1}^{N} y_i \log \sigma(\boldsymbol{q}_o^T \mu(s_i|\theta)) + (1 - y_i) \log(1 - \sigma(\boldsymbol{q}_o^T \mu(s_i|\theta))) \longrightarrow L(\theta) = \tilde{Q}((o, y_i), \mu(s_i|\theta))$$

$$\min_{\boldsymbol{w}} || \sum_{m=1}^{M} w_m \nabla_\theta \sum_{i=1}^{N} Q_m(s_i, \mu(s_i|\theta))||_2^2$$
$$s.t. \; \boldsymbol{e}_k^T \boldsymbol{w} \geq b_k, \; \forall \, k \in [1, K]$$
$$\boldsymbol{1}^T \boldsymbol{w} = 1, \; w_m \geq 0, \; \forall \, m \in [1, M]$$

Theorem 1. *If $\boldsymbol{w}$ is determined by solving the quadratic programming (QP) problem of (5), then either one of the following holds:*
i) The solution to the optimization problem is 0, then the local Pareto efficient solution is achieved.
ii) $\boldsymbol{d} = \sum_{m=1}^{M} w_m \nabla_\theta \sum_{i=1}^{N} Q_m(s_i, \mu(s_i|\theta))$ is a gradient direction which does not decrease any Q function.

Algorithm 1: Pareto Deterministic Policy Gradient

Initialize Actor parameter $\boldsymbol{\theta}$ and Target Actor parameter $\boldsymbol{\theta}' \leftarrow \boldsymbol{\theta}$.
Initialize Critic parameter $\boldsymbol{\phi}_m$ and Target Critic parameter $\boldsymbol{\phi}'_m \leftarrow \boldsymbol{\phi}_m, \forall m \in [1, M]$.
Initialize Pareto weights $\boldsymbol{w} = \{\frac{1}{M+1}, \frac{1}{M+1}, \dots, \frac{1}{M+1}\}$ and replay buffer B.
for *episode number in [1, K]* **do**
  i) Trajectory Generation
  Get start state s_1
  for *step t in [1, T]* **do**
    Select an action according to $a_t = \mu(s_t|\boldsymbol{\theta}) + N_t$, N_t is an exploration noise.
    Execute a_t to obtain the new state s_{t+1} and the reward vector $\boldsymbol{r}_t = \{r_{t,1}, r_{t,2}, \dots, r_{t,M}\}$.
    Push $\{s_t, a_t, \boldsymbol{r}_t, s_{t+1}\}$ into the replay buffer B
  end
  ii) Update Critic
  Sample Z instances $\{s_i, a_i, \boldsymbol{r}_i, s_{i+1}\}$ from B
  for *critic m in [1, M]* **do**
    for *i in [1, Z]* **do**
      Compute $y_i = r_{i,m} + \gamma Q_m(s_{i+1}, \mu(s_{i+1}|\boldsymbol{\theta}')|\boldsymbol{\phi}'_m)$.
    end
    $\boldsymbol{\phi}_m \leftarrow \boldsymbol{\phi}_m - \alpha_\phi \nabla_{\boldsymbol{\phi}_m}\{\frac{1}{Z}\sum_{i=1}^{Z}(y_i - Q_m(s_i, a_i|\boldsymbol{\phi}_m))^2\}$.
  end
  iii) Update Pareto Weight
  for *i in [1, Z]* **do**
    for *m in [1, M]* **do**
      $\boldsymbol{p}_{i,m} = \nabla_a Q_m(s,a)|_{s=s_i, a=\mu_\theta(s_i)} \nabla_{\boldsymbol{\theta}}\mu(s|\boldsymbol{\theta})|_{s=s_i}$.
    end
    $\tilde{\boldsymbol{p}}_i = \nabla_a \tilde{Q}(s,a))|_{s=(o,y_i), a=\mu_\theta(s_i)} \nabla_{\boldsymbol{\theta}}\mu(s|\boldsymbol{\theta})|_{s=s_i}$.
  end
  Update $\boldsymbol{w} = \{w_1, w_2, \dots, w_M, \tilde{w}\}$ by Solving (5).
  iv) Update Actor
  $\boldsymbol{d} = \frac{1}{Z}\sum_{i=1}^{Z}\sum_{m=1}^{M} w_m \boldsymbol{p}_{i,m} + \tilde{w}\tilde{\boldsymbol{p}}_i$.
  $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha_\theta \boldsymbol{d}$.
  $\boldsymbol{\theta}' \leftarrow \tau\boldsymbol{\theta} + (1-\tau)\boldsymbol{\theta}'$.
  $\boldsymbol{\phi}'_m \leftarrow \tau\boldsymbol{\phi}_m + (1-\tau)\boldsymbol{\phi}'_m \quad \forall m \in [1, M]$.
end

$$G = \left|\mathbb{E}_{\boldsymbol{B}_i}\left[\sum_{m=1}^{M+1} w_m(\boldsymbol{B}_i)\left(\frac{1}{Z}\sum_{s_b \in \boldsymbol{B}_i} f_m(s_b;\boldsymbol{\theta}) - \mathbb{E}_s[f_m(s;\boldsymbol{\theta})]\right)\right]\right|$$

Theorem 2. *Suppose i) $\nabla_a Q_m(s,a)$ and $\nabla_{\boldsymbol{\theta}}\mu(s|\boldsymbol{\theta})$ are bounded by X_m and Y, that is, $||\nabla_a Q_m(s,a)||_2 \le X_m$ and $||\nabla_{\boldsymbol{\theta}}\mu(s|\boldsymbol{\theta})||_2 \le Y$. ii) The batched gradient of the action-value function for each objective is unbiased, that is: $\mathbb{E}_{\boldsymbol{B}_i}[\frac{1}{Z}\sum_{s_b \in \boldsymbol{B}_i} f_m(s_b;\boldsymbol{\theta})] = \mathbb{E}_s[f_m(s;\boldsymbol{\theta})]$. iii) $f_m(s_b;\boldsymbol{\theta})$ follows a normal distribution $\mathcal{N}(\mathbb{E}_s[f_m(s;\boldsymbol{\theta})], \sigma^2 I)$, where $I \in \mathbb{R}^{d\times d}$ is an identity matrix and σ is a scalar. Then we have:*

$$G \le \sum_{m=1}^{M+1}\mathbb{E}_{\boldsymbol{B}_i}\left[w_m(\boldsymbol{B}_i)\left(\left|\frac{1}{Z}\sum_{s_b \in \boldsymbol{B}_i} f_m(s_b;\boldsymbol{\theta}) - \mathbb{E}_s[f_m(s;\boldsymbol{\theta})]\right|\right)\right] \quad (14)$$

$$\le \frac{X_{m*} Y \sqrt{d}}{\sqrt{Z}} \quad (15)$$

where $m^* = \arg\max_m |\frac{1}{Z}\sum_{s_b \in \boldsymbol{B}_i} f_m(s_b;\boldsymbol{\theta}) - \mathbb{E}_s[f_m(s;\boldsymbol{\theta})]|$.

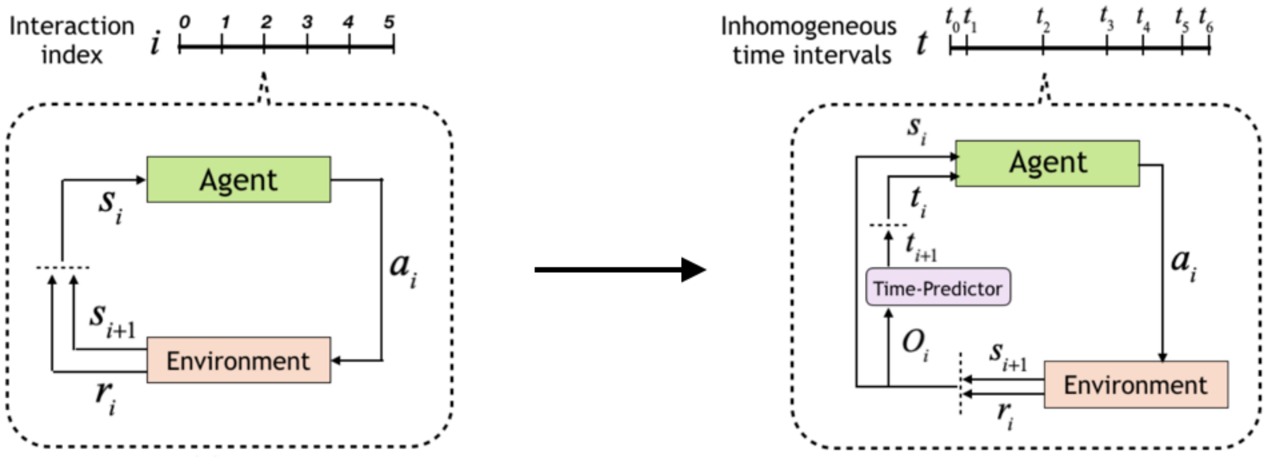
Weight-reuse mechanism. In this method, we introduce a container $\boldsymbol{W} \in \mathbb{R}^{L\times(M+1)}$ for storing previously derived Pareto weights. For each training batch $\boldsymbol{B}_i$, $\boldsymbol{w} \in \mathbb{R}^{M+1}$ is not always computed by solving problem (5). We firstly check the weights in the container:

(1) If there is a candidate $\boldsymbol{w}^* \in \boldsymbol{W}$, such that its corresponding $\boldsymbol{d}^* = \sum_{m=1}^{M+1} w_m^* \nabla_{\boldsymbol{\theta}} \left(\frac{1}{Z} \sum_{s_b \in \boldsymbol{B}_i} Q_m(s_b, \mu(s_b|\boldsymbol{\theta})) \right)$ can increase all the Q functions, that is, $(\boldsymbol{d}^*)^T \nabla_{\boldsymbol{\theta}} \left(\frac{1}{Z} \sum_{s_b \in \boldsymbol{B}_i} Q_m(s_b, \mu(s_b|\boldsymbol{\theta})) \right) > 0, \forall m \in [1, M+1]$, then we set $\boldsymbol{w} = \boldsymbol{w}^{*3}$. Since the weights in $\boldsymbol{W}$ is not derived from $\boldsymbol{B}_i$, the bias G becomes 0 at this moment.

(2) If there is no such weight in $\boldsymbol{W}$, we solve problem (5) to derive $\boldsymbol{w}$, which is then pushed into the container for future "reuse". In this scenario, G is not 0, which is bounded by equation (15).

Modelling User Dynamic Preference

Dynamic user preference



Inhomogeneous DQN

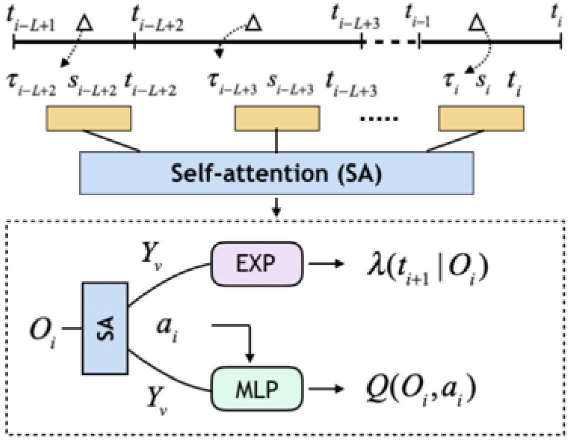
Trajectory $\tau = \{s_0, a_0, t_0, s_1, a_1, t_1, ..., s_T, a_T, t_T\}$

Objective $J = \mathbb{E}_{\tau \sim p_{\theta,\psi}(\tau)}[\sum_{i=0}^{T} \gamma^i \kappa(t_0 - t_i) r(s_i, a_i, t_i))]$

Bellman Operator

$$(TQ)(s_i, a_i, t_i)$$
$$= r(s_i, a_i, t_i) + \sum_{s_{i+1} \in \mathcal{S}} \mathcal{P}(s_{i+1}|s_i, a_i)\{\mathbb{E}_{t_{i+1} \sim \mathcal{T}(\cdot|O_i)}[\gamma \kappa(t_i - t_{i+1})$$
$$\max_{a_{i+1} \in \mathcal{A}} Q(s_{i+1}, a_{i+1}, t_{i+1})]\}$$

Theorem 1 (T is a contractive operator.). *Let Q_1 and Q_2 be two value functions. Then, the Lipschitz condition $||TQ_1 - TQ_2||_\infty \leq \alpha||Q_1 - Q_2||_\infty$ holds, where $\alpha \in [0, 1)$ is a constant.*



$$Q(O_i, a_i|\boldsymbol{\phi}_Q, \boldsymbol{\phi}_Y) = f_Q^n(f_Q^{n-1}(\ldots(f_Q^1(Y_v \boldsymbol{w}_3, \boldsymbol{p}_i))\ldots))$$

$$\lambda^*(t_{i+1}|\boldsymbol{\phi}_\lambda, \boldsymbol{\phi}_Y) = \lambda(t_{i+1}|O_i; \boldsymbol{\phi}_\lambda, \boldsymbol{\phi}_Y)$$
$$= \exp\big(\underbrace{\boldsymbol{w}_1^T Y_v \boldsymbol{w}_2}_{A} + \underbrace{w_t(t_{i+1} - t_i)}_{B} + \underbrace{b}_{C}\big)$$

$$\boldsymbol{x}_j = [\boldsymbol{e}_j; \boldsymbol{\tau}_j] + \boldsymbol{v}_j$$

$$\boldsymbol{Q} = \boldsymbol{X}^T \boldsymbol{W}_Q, \boldsymbol{K} = \boldsymbol{X}^T \boldsymbol{W}_K, \boldsymbol{V} = \boldsymbol{X}^T \boldsymbol{W}_V,$$
$$\boldsymbol{X}_1 = \mathrm{SOFTMAX}(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{d_K}})\boldsymbol{V},$$
$$Y_1 = \boldsymbol{W}_2^F \mathrm{ReLU}(\boldsymbol{W}_1^F \boldsymbol{X}_1^T + \boldsymbol{b}_1^F) + \boldsymbol{b}_2^F,$$

$$L(\phi_Q, \phi_Y, \phi_\lambda) = E_{(O_i, a_i, r_i, s_{i+1}, t_{i+1})}[(y_i - Q(O_i, a_i))^2]$$

$$= \int_{O_i} \underbrace{p(O_i)}_{A} \underbrace{E_{(a_i, r_i, s_{i+1}, t_{i+1} | O_i)}[(y_i - Q(O_i, a_i))^2]}_{B} \mathrm{d}O_i$$

A

$$L_P(\phi_\lambda, \phi_Y) = \sum_{O_i} \sum_{j=1}^{i} \log p(t_j | O_{j-1}) = \sum_{O_i} \sum_{j=1}^{i} \log f^*(t_j | O_{j-1})$$

$$= \sum_{O_i} \sum_{j=1}^{i} \{\log \lambda^*(t_j | \phi_\lambda, \phi_Y) - \int_{t_{j-1}}^{t_j} \lambda^*(\tau | \phi_\lambda, \phi_Y) \mathrm{d}\tau\}$$

B

$$L_Q(\phi_Q, \phi_Y) = \sum_{D} (y_i - Q(O_i, a_i | \phi_Q, \phi_Y))^2$$

Theorem 2 (Finite-time bound for IDQN). *Suppose we have K training iterations in the optimization process, and for the kth iteration, the Q-network is updated from Q_{k-1} to Q_k as follows:*

$$Q_k = \arg\min_{f \in \mathcal{F}} \sum_{i=1}^{N} [y_i - f(O_i, a_i)]^2 \\ y_i = r_i + \gamma\kappa(t_i - t_{i+1}) \max_{a_{i+1}} Q_{k-1}(O_{i+1}, a_{i+1}) \tag{14}$$

where we have N training samples, and (O_i, a_i) in each instance is drawn from a distribution σ.

Suppose (i) $k(m; \mu, \sigma)$ is the concentration coefficient as defined in [37], where μ is the distribution on (O_i, a_i) after m MDP steps, and $k(m; \mu, \sigma)$ measures the similarity between μ and σ. (ii) l^k is the minimum time interval spanning k steps in all trajectories, that is, $l^k = \min_{i,\tau}(t^{\tau}_{i+k} - t^{\tau}_{i})$, where t^{τ}_{i} is the agent-environment interaction time for the ith step in trajectory τ (iii) $e_{k+1} = TQ_k - Q_{k+1}$ and $s = \max_i ||e_i||_{\sigma}$. We assume (i) the immediate reward is bounded by R_{max}, (ii) $\sum_{m\geq 1}[\gamma^{m-1} m k(m; \mu, \sigma)]^2 \leq \phi_{\mu,\sigma}$, and (iii) $\sum_{k=0}^{\infty} \kappa(-l^k)^2 \leq \phi_{\kappa}$.

Let π_K be the one-step greedy policy of Q_K, and Q^{π_K} be the action-value function corresponding to π_K. $Q^*(O, a) = sup_{\pi} Q^{\pi}(O, a)$ is the optimal Q-function. Then, the upper bound of the error between Q^* and Q^{π_K} is:

$$|Q^* - Q^{\pi_K}|_{1,\mu} \leq 2\gamma s(\phi_{\mu,\sigma}\phi_{\kappa})^{\frac{1}{2}} + \frac{4\gamma^{K+1} R_{max}}{(1-\gamma)^2} \tag{15}$$

Outlook

What if the logged user preference is biased?

Counterfactual trajectory generation

Debiased user preference learning

...

How to build high reliable simulator?

Not limited to RL-based Recsys

Potential to promote the Recsys research

...

Thanks & QA