Reinforcement Learning for Industrial Recommender Systems

Qingpeng Cai

July. 15, 2022

Outline

1 Reinforcement Learning

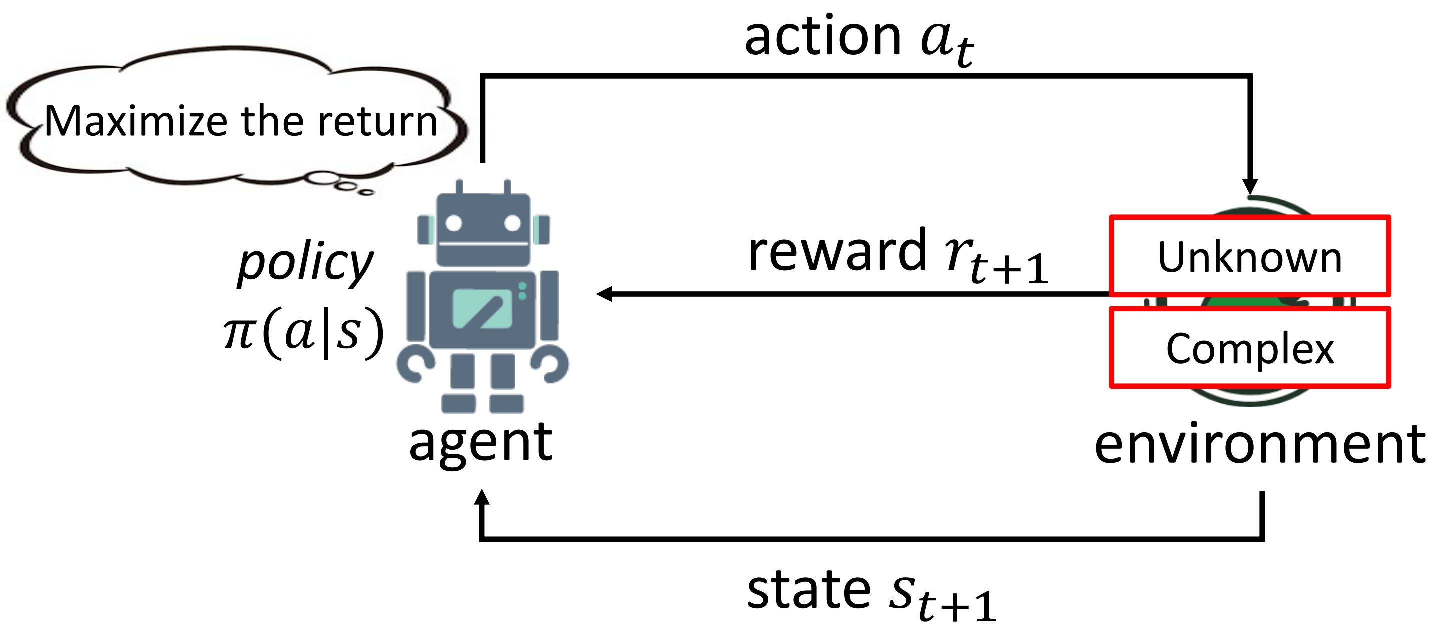
2 Reinforcement Learning for Short Video Recommender Systems

- Introduction, Formulations and Challenges
- Basic Version of Reinforcement Learning for Kuaishou RS
- Advanced Version of Reinforcement Learning for Kuaishou RS
    - Constrained RL
    - Exploration

3 Future Research Directions

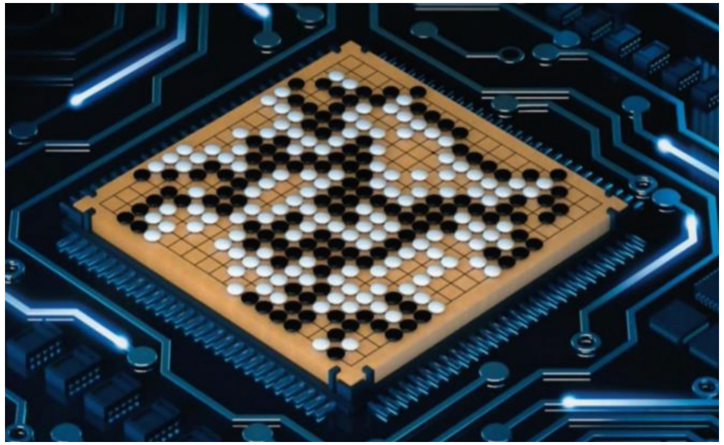
Reinforcement Learning

Reinforcement Learning



Deep Reinforcement Learning

Atari



Go

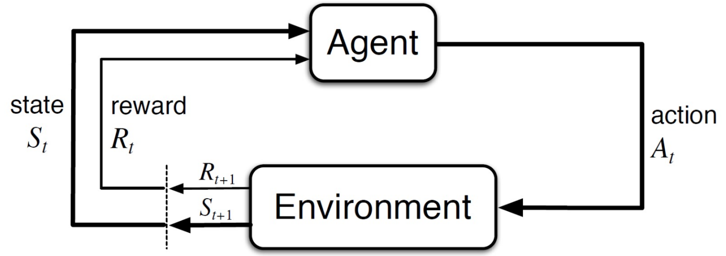
StarCraft II

Robotics

Transportation

Introduction of Reinforcement Learning

- Agent maximizes rewards by interaction with environments



- Markov Decision Process (MDP)：
- Markov Property： $P(s_{t+1} | s_t, \ldots, s_1) = P(s_{t+1} | s_t)$
- Tuple:(S, A, P,R,γ)
- Objective：Find the policy that maximizes the discounted sum of rewards

$$G_t = R_{t+1} + \gamma R_{t+2} + \ldots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

- Bellman Equation
  - Value function

$$v_\pi(s) = \sum_{a \in \mathcal{A}} \pi(a|s) \left(\mathcal{R}_s^a + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^a v_\pi(s') \right)$$

  - Q function

$$q_\pi(s,a) = \mathcal{R}_s^a + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^a \sum_{a' \in \mathcal{A}} \pi(a'|s') q_\pi(s',a')$$

Introduction of Reinforcement Learning

RL algorithms

Reinforce

Trajectory $\tau = \{s_1, a_1, s_2, a_2, \cdots, s_T, a_T\}$

$$p_\theta(\tau) = p(s_1)p_\theta(a_1|s_1)p(s_2|s_1,a_1)p_\theta(a_2|s_2)p(s_3|s_2,a_2)\cdots$$

$$= p(s_1)\prod_{t=1}^{T} p_\theta(a_t|s_t)p(s_{t+1}|s_t,a_t)$$

$$R(\tau) = \sum_{t=1}^{T} r_t$$

$R(\tau)$ do not have to be differentiable

Expected Reward:

$$\bar{R}_\theta = \sum_\tau R(\tau)p_\theta(\tau) = E_{\tau\sim p_\theta(\tau)}[R(\tau)]$$

$$\nabla f(x) = f(x)\nabla log f(x)$$

$\nabla \bar{R}_\theta = ?$

$$\nabla \bar{R}_\theta = \sum_\tau R(\tau)\nabla p_\theta(\tau) = \sum_\tau R(\tau)p_\theta(\tau)\frac{\nabla p_\theta(\tau)}{p_\theta(\tau)}$$

$$= \boxed{\sum_\tau} R(\tau)\boxed{p_\theta(\tau)}\nabla log p_\theta(\tau)$$

$$= E_{\tau\sim p_\theta(\tau)}[R(\tau)\nabla log p_\theta(\tau)] \approx \frac{1}{N}\sum_{n=1}^{N} R(\tau^n)\nabla log p_\theta(\tau^n)$$

$$= \frac{1}{N}\sum_{n=1}^{N}\sum_{t=1}^{T_n} R(\tau^n)\nabla log p_\theta(a_t^n|s_t^n)$$

Deep Q Network

$$Q(s,a) = Q(s,a) + \alpha[r + \gamma \max_{a'} Q(s',a') - Q(s,a)]$$

损失函数：MSE

$$L = r + \gamma \max_{a'} Q(s',a') - Q(s,a)$$

训练技巧:

- Experience-Replay
- Double Q-Network

Actor-Critic Algorithm

Critic 优化:

$$Loss = min\left(r_t + \gamma * V(s_{t+1}) - V(s_t)\right)^2$$

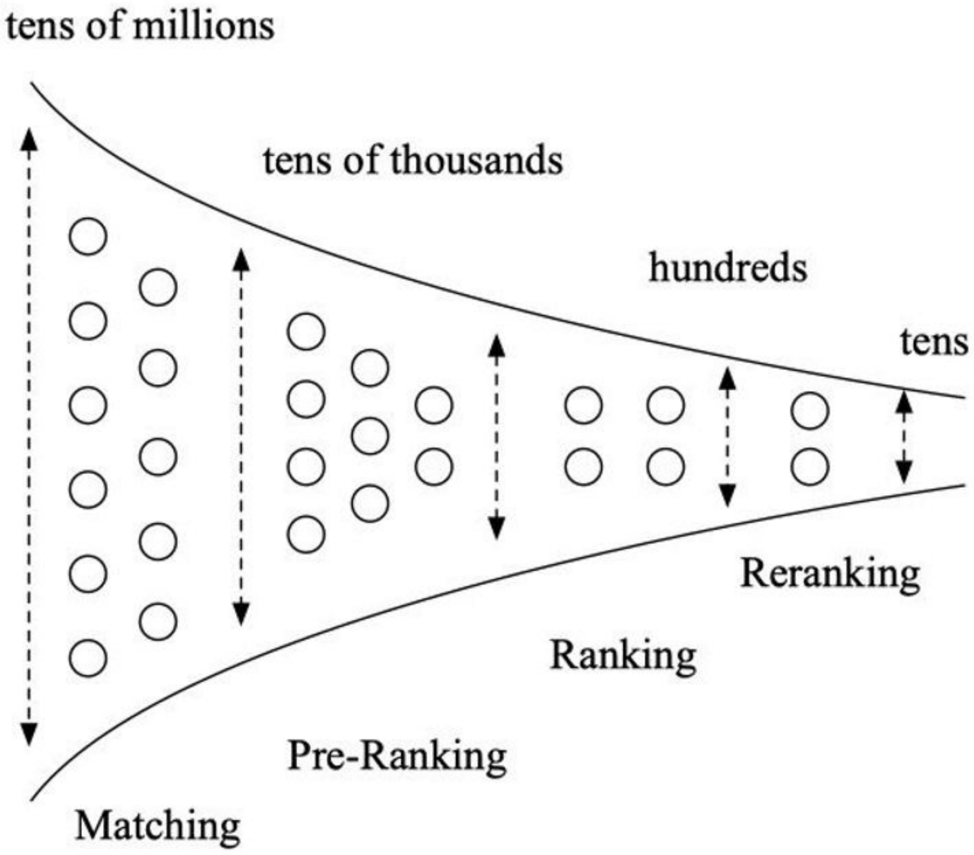
Actor 优化：

$$advantage_t = Q(s_t,a_t) - V(s_t) = r_t + \gamma * V(s_{t+1}) - V(s_t)$$

$$Loss = min(-log(\pi) * advantage_t)$$

Reinforcement Learning for Short Video Recommender System: Introduction, Formulations and Challenges

Infrastructures of Short Video Recommender Systems



Why RL for RS?

- Most recommender systems deploy supervised learning methods
  - Predict the value of the candidate items or the lists
  - Problem
    - Lack of exploration ability
    - Unable to optimize the long-term signals
      - Short-term signals: the reward of each item and list
      - Long-term signals: the total rewards of each session, or user returning time

- RL is a perfect paradigm to tackle these problems
  - Interact with the environment by exploration and exploitation
  - Aim to maximize the long-term rewards

MDP Definition of RL for RS

- State
  - For each user request, we have a state:(user information, user history)
  - User history: actions and rewards of previous steps
- Action
  - Two common choices
    - The ID of the item to be recommended to the user
    - The hyper-parameter of ranking functions
- Immediate Reward
  - The user signals at each request
- Episode
  - The lists of user requests in a session or a day
- Objective
  - Aim to maximize the total positive signals of all users

Challenge of RL for Real RS

- Unstable Environment
  - Each user is a environment, rather than fixed game
  - System fluctuates between days and hours
- Large action space
  - The number of candidate items is over 100M
- Multi-objectives
  - Different reward signals in short-videos: dwell time, like, follow, forward, comment, visiting depth
- Safe and efficient exploration
  - Random exploration hurts user experience
- Delayed feedback and credit assignment
  - The long-term engagement signal is delayed and noisy
  - It is hard to allocate credits to immediate actions

Basic Version of Reinforcement Learning for Kuaishou RS

Motivation of RL in Kuaishou RS

- Many hyper-parameters Exist in Kuaishou RS
- How to learn optimal parameters to maximize different objectives?
  - Objectives: watch time, interactions, session depth
  - Non-gradient methods CEM/Bayes are used in Kuaishou
    - Unable to optimize long-term metric
    - Lack of personalization

- RL
  - Personalization
  - Aim to maximize the long-term performance

Request-based MDP for RS

- MDP
  - State:(user information, user history)
    - User information:
    - User history: states, actions, and rewards of previous steps
  - Action
    - Parameters of several ranking functions
    - A continuous vector
  - Reward
    - $r_t = watch\ time + like\ count\ *\ w_{like} + follow\ count\ *\ w_{follow} + forward\ count\ *\ w_{forward}$
  - Episode
    - Requests from opening the app to leaving the app

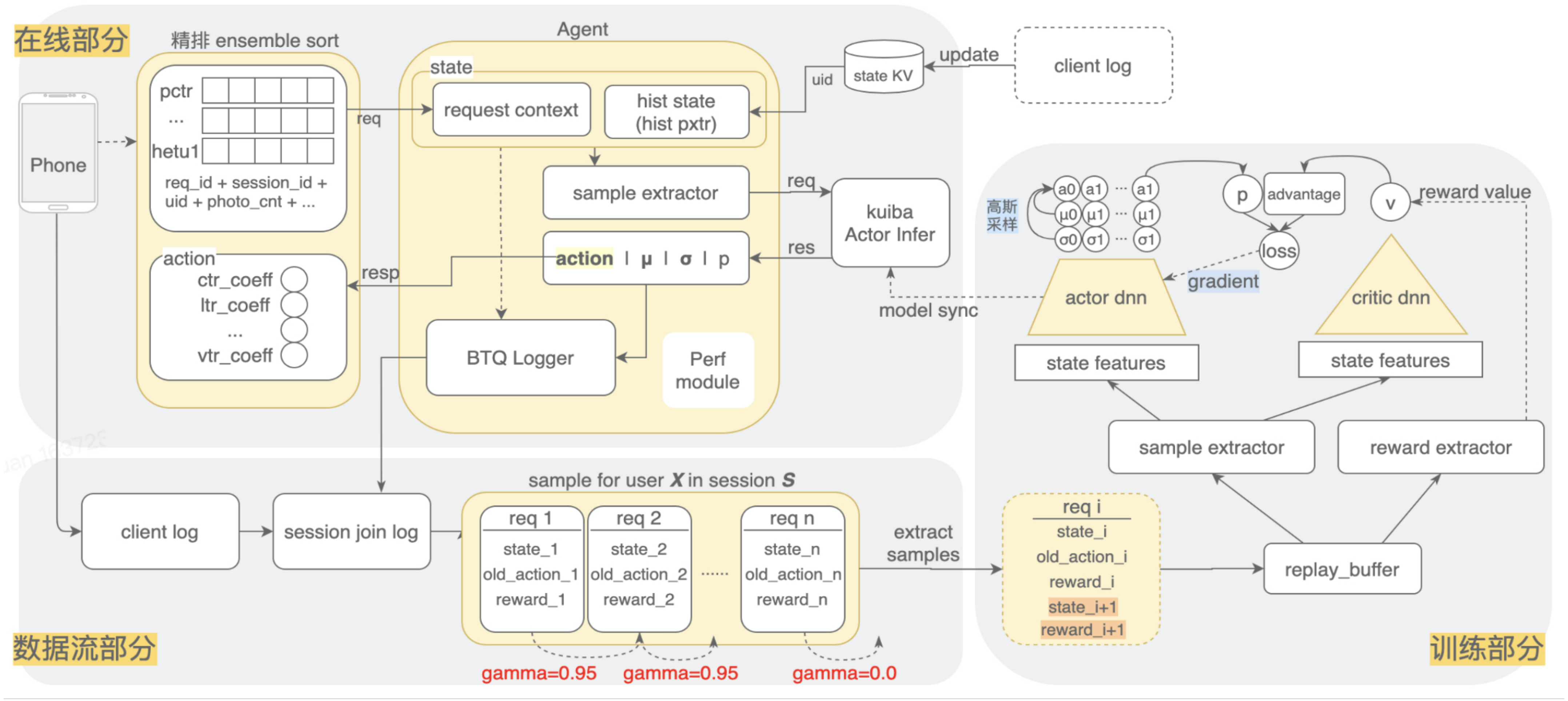
Request-based MDP for RS

- Objective

$$\max \sum_{t=0}^{T} \gamma^t (time_t + w1 * like_t + w2 * follow_t + w3 * forward_t + w4 * comment_t + w5 * 0.1)$$

- Policy
  - DNN
  - Input state , output mu and sigma
  - Sample action from Gaussian distribution
- Algorithm Selection
  - Reinforce (Google)
    - Slow convergence, only works for single objective
  - PPO
    - On-policy, does not work for off-policy setting of KS
  - A3C
    - Faster convergence , sensitive to different reward coefficient

Request-based MDP for RS



Request-based MDP for RS

- Loss functions
  - Actor loss $-log\pi(a|s)(r + \gamma * V(s') - V(s))$
  - Critic loss $(r + \gamma * V(s') - V(s))^2$
- Live Experiments
  - Baseline: CEM
  - Avg app time +0.15% Watch time +0.33% follow -1.08% VV -0.49%
  - Fully launched
- Comparison with Contextual Bandits
  - Gamma=0: contextual bandits
  - Gamma=0.95 compares with gamma=0
    - App time +0.089%, VV +0.37%, follow flat
    - RL performs better than Bandits!

Constrained Reinforcement Learning for Kuaishou RS

Constrained RL for Short-video Recommendation

- Several signals in short-video recommendations
  - Watch time of multiple videos
    - Main objective of the algorithm
    - Dense responses
      - Can be effectively optimized by RL
  - Share, Download, Comment
    - Sparse responses
    - Serve as constraints
- The optimization program

$$
\begin{aligned}
\max_{\pi} \quad & U_1(\pi) \\
\text{s.t.} \quad & U_i(\pi) \geq C_i, \quad i = 2, \ldots, m,
\end{aligned}
$$

Problem of Basic Version of RL

- The method is equal to learning a policy to optimize its Lagrangian

$$\mathcal{L}(\pi, \lambda) = U_1(\pi) + \sum_{i=2}^{m} \lambda_i (U_i(\pi) - C_i), \quad \text{where} \quad \lambda_i \geq 0.$$

- The method exists following questions:
  - 1.The estimation of the policy is not accurate for sparse signals
    - The dense signal, such as watch time dominates the estimation
  - 2.The discount factor of each signal is the same
    - The discount factor for sparse signal should be small
  - 3.It is hard for a single policy to balance both dense responses and sparse responses
    - Learning to optimize the sparse signal is difficult

Multi-Critic Policy Optimization

- Multi-Critic
  - Each critic estimated the value of one objective (Challenge 1)
    - Compare V_s and (V_w, V_i)
      - V_s learns watch time+interaction
      - V_w learns watch time, V_i learns interaction
      - Use MAE error to estimate two learning method
      - Separate learning outperforms joint learning by 0.191% and 0.143% in terms of both watch time and interaction
  - Different critic has different discounted factor (Challenge 2)
    - For Watch time, we can set factor to be 0.95
    - For Interactions, we can set small factors

Multi-Critic Policy Optimization

- Actor Optimizes the Weighted Advantages

$$\max_{\theta} log\pi_\theta(a|s)(Adv_{time}(s,a) + \lambda_1 Adv_{follow}(s,a) + \lambda_2 Adv_{forward}(s,a) + \lambda_3 Adv_{like}(s,a) + \lambda_4 Adv_{comment}(s,a))$$
$$Adv_x(s,a) = Q(s,a) - V(s)$$

- Live Experiments
  - Baseline: Basic RL Version
  - Avg app time +0.130% Watch time +0.387% follow -1.97% VV +0.15%
  - Fully launched
  - More flexible with weights of different objectives

Two-stage Constrained Actor-Critic(In submission)

- Challenge 3 still exists
  - It is hard for single policy to optimize both dense and sparse metrics.
- Stage One
  - For each auxiliary response, learn a policy to optimize its cumulative reward

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

We update the actor to maximize the advantage:

$$\theta_i^{(k+1)} \leftarrow \arg\max_{\theta} \mathbb{E}_{\pi_{\theta_i^{(k)}}} \left[A_i^{(k)} \log \left(\pi_\theta(a|s) \right) \right]$$
$$\text{where} \quad A_i^{(k)} = r_i(s,a) + \gamma_i V_{\phi_i^{(k)}}(s') - V_{\phi_i^{(k)}}(s).$$

- Stage Two
  - For the main response, learn a policy to optimize its cumulative reward
  - Softly regularize the policy to be close to other auxiliary policies

far from other policies. The optimization is formalized below:

$$\begin{aligned} \max_{\pi} \quad & \mathbb{E}_\pi [A_1^{(k)}] \\ \text{s.t.} \quad & D_{KL}(\pi || \pi_{\theta_i}) \le \epsilon_i, \quad i = 2, \dots, m, \\ \text{where} \quad & A_1^{(k)} = r_1(s,a) + \gamma_1 V_{\phi_1^{(k)}}(s') - V_{\phi_1^{(k)}}(s). \end{aligned} \tag{8}$$

Equation (8) has the closed form solution

$$\pi^*(a|s) \propto \prod_{i=2}^{m} \left(\pi_{\theta_i}(a|s) \right)^{\frac{\lambda_i}{\sum_{j=2}^m \lambda_j}} \exp\left(\frac{A_1^{(k)}}{\sum_{j=2}^m \lambda_j} \right), \tag{9}$$

where λ_i with $i = 2, \dots, m$ are Lagrangian multipliers for constraints in (8). Following [28], we specify the value of λ_i to control the degree of constraint.

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^*:

$$\begin{aligned} \theta_1^{(k+1)} & \leftarrow \arg\min_{\theta} \mathbb{E}_{\pi_{\theta_1^{(k)}}} [D_{KL}(\pi^*(\cdot|s) || \pi_\theta(\cdot|s))] \\ & = \arg\max_{\theta} \mathbb{E}_{\pi_{\theta_1^{(k)}}} \left[\prod_{i=2}^{m} \left(\frac{\pi_{\theta_i}(a|s)}{\pi_{\theta_1^{(k)}}(a|s)} \right)^{\frac{\lambda_i}{\sum_{j=2}^m \lambda_j}} \exp\left(\frac{A_1^{(k)}}{\sum_{j=2}^m \lambda_j} \right) \log \pi_\theta(a|s) \right]. \end{aligned} \tag{10}$$

Two-stage Constrained Actor-Critic

Algorithm 1: Two-Stage Constrained Actor Critic

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:

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

$$\theta_i^{(k+1)} \leftarrow \arg\max_{\theta} \mathbb{E}_{\pi_{\theta_i^{(k)}}} \left[A_i^{(k)} \log \left(\pi_\theta(a|s) \right) \right].$$

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:

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

$$\theta_1^{(k+1)} \leftarrow \arg\max_{\theta} \mathbb{E}_{\pi_{\theta_1^{(k)}}} \left[\prod_{i=2}^{m} \left(\frac{\pi_{\theta_i}(a|s)}{\pi_{\theta_1^{(k)}}(a|s)} \right)^{\frac{\lambda_i}{\sum_{j=2}^m \lambda_j}} \right.$$

$$\left. \times \exp\left(\frac{A_1^{(k)}}{\sum_{j=2}^m \lambda_j} \right) \log \pi_\theta(a|s) \right].$$

Output: the constrained policy π_1.

Live Experiments

- Baselines and Our Algorithms
  - A3C(basic RL version)
    - A RL algorithm to optimize the weighted reward, with $\gamma = 0.95$
  - RCPO-A3C(Multi-critic Policy Optimization)
    - Learning two critic to evaluate the time and the interactions
    - Optimize the actor by the weighted sum of advantages of two objectives
  - Two-Stage constrained A3C (Ours)
    - Stage 1: we learn a A3C policy to optimize the interactions
    - Stage 2: we learning a policy following Eq(10).
  - CEM(Baseline)

Live Experiments

Algorithm	WatchTime	Share	Download	Comment
A3C	+0.309%	−0.707%	0.153%	−1.313%
RCPO-A3C	+0.283%	−1.075%	−0.519%	−0.773%
Interaction-A3C	+0.117%	+5.008%	+1.952%	−0.101%
Constrained-A3C	+0.336%	+3.324%	+1.785%	−0.618%

Table 2: Performance of different algorithms relative to a supervised LTR baseline in a live experiment.

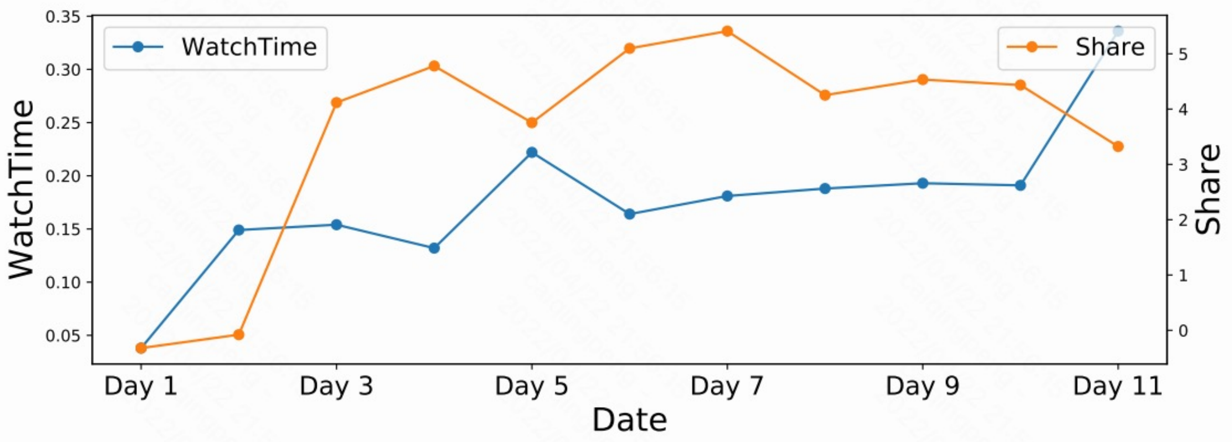


Figure 3: Learning curve of Contrained-Time-A3C. The blue and orange lines show its live performance on WatchTime and Share as compared to a supervised LTR baseline.

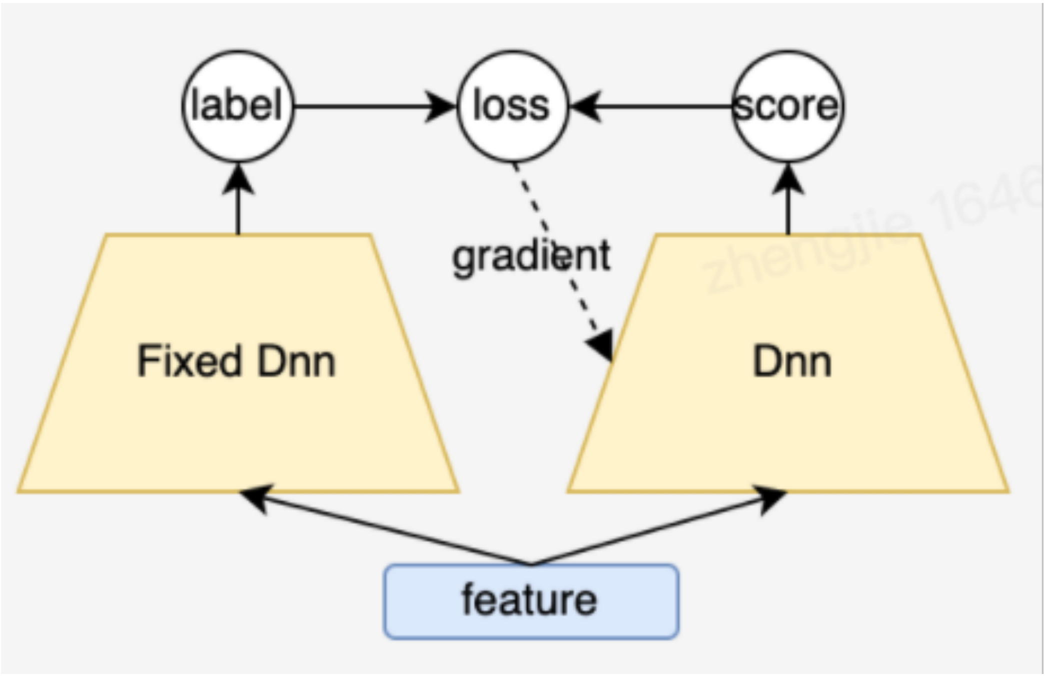
Exploration for Kuaishou RS

RND as Exploration

- Exploration
  - Crucial for RL
  - Simply improving entropy of policy does not improve performance
  - Random exploration hurts user experience
- RND(Random Network Distillation)
  - Estimate the novelty of each state
    - Higher frequency, lower novelty
  - Two same networks
    - One random initialized
    - The other one learns to fit one
  - Loss and Exploration reward

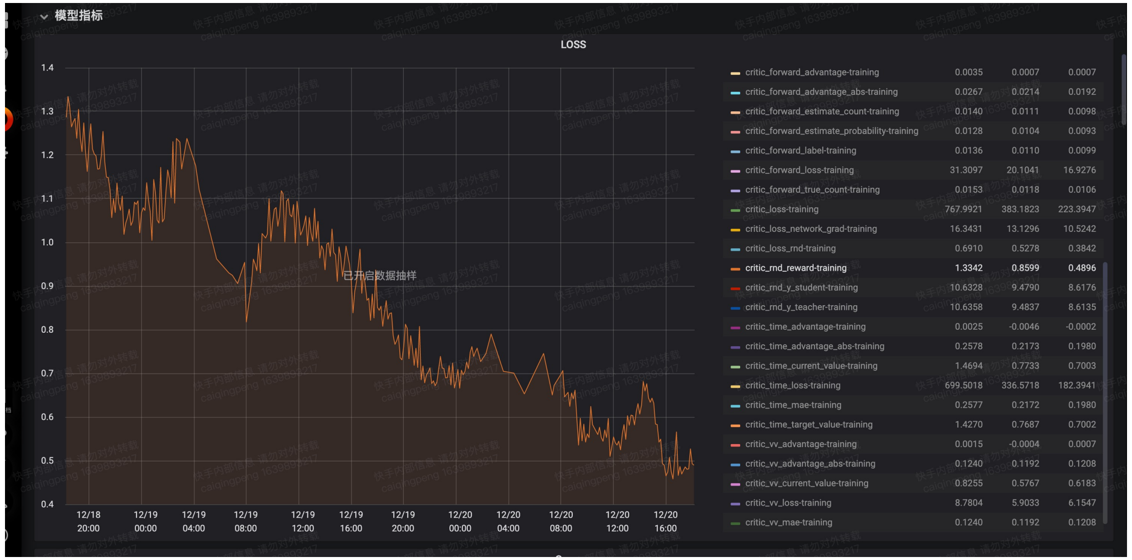
$$\min_{\theta} ||f_{\theta}(s) - f_{\theta*}(s)||_2^2$$

$$reward = r_e + ||f_{\theta}(s) - f_{\theta*}(s)||_2^2$$



RND as Exploration

- Training
  - Loss decreases with training



- Live Experiments
  - Baseline: Basic RL Version
  - Avg app time +0.231% Watch time +0.476% follow -1.96% VV +0.07%
  - Fully launched

Future Directions

Future Directions

- Exploration in large-scale action spaces
  - How to ensure safe exploration?
  - Efficient explorations
- Multi-agent Reinforcement Learning in RS
  - Different agent maximizes different signals
  - Different agent works in different phases of the recommend systems
- Counterfactual Reinforcement Learning in RS
  - Unbiased evaluation of a RL policy in RS
  - Credit assignment of a long-term delayed signal to each immediate steps

Thank you!