RL Reading Group:
Top-K Off-Policy Correction for a REINFORCE Recommender System

Tingting Xu

10/21/2019

Boston University Division of System Engineering

BOSTON UNIVERSITY

Overview

- Objective and Challenges
- Techniques
  - Reinforce Recommender
  - Off-policy Correction
- Top-K Off-policy Correction
- Exploration

1. Objective and Challenges

- Objective
  - build a Recommender Systems to discover a small fraction of content that users would be interested in among huge corpuses of content
- Information
  - a sequence of user historical interactions with the system,
    - videos recommended
    - implicit user feedback, such as clicks and dwell-time
    - Context: page, device and time
- Why RL?
  - maximize each user's *long term* satisfaction with the system.
  - new perspectives on recommendation problems as well as opportunities to build on top of the recent RL advancement.

Challenges of RL in Recommender Systems

- Extremely large action and state spaces:
  - action spaces – many millions of items to recommend
  - state space - billions of users
- Sparse data:
  - most users exposed to a small fraction of items
  - explicit feedback provided to an even smaller fraction
- Continuously-evolving user states:
  - user preferences over these items are shifting all the time

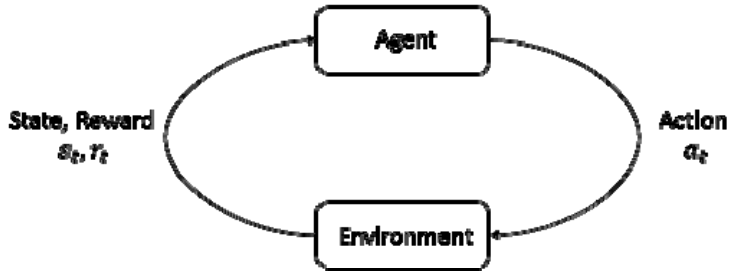
2. Which RL algorithm?

- value-based methods (e.g. Q Learning)
  - Pros
    - seamless off-policy learning
  - Cons
    - instability with function approximation
    - extensive hyper-parameter tuning is required to achieve stable behavior for these approaches
    - policy convergence not well-studied
- Policy-based approaches (e.g. REINFORCE)
  - stable w.r.t. function approximations given a sufficiently small learning rate
- This paper
  - adapt REINFORCE (on-policy method)
    - policy-gradient-based approach
  - provide reliable policy gradient estimates when training off-policy

Markov Decision Process

- A finite MDP is a 5-tuple $\mathcal{M} =< S, A, P, R, \rho_0, \gamma >$:
  - S: continuous state space
  - A: discrete action space
    - items available for recommendation
  - $R: S \times A \times S \to \mathbb{R}$ is the reward function

    The immediate reward is $r(s, a)$
  - $P: S \times A \times S \to [0,1]$ is the state transition probability
  - ρ_0: initial state distribution
  - γ: discount factor for future rewards



[1] Bertsekas, Dimitri P. Abstract dynamic programming. Athena Scientific, 2018.
[2] https://spinningup.openai.com/en/latest/spinningup/rl_intro.html

3. REINFORCE Recommender

- Objective

$$\max_{\pi} \mathbb{E}_{\tau\sim\pi}\left[R(\tau)\right], \text{ where } R(\tau) = \sum_{t=0}^{|\tau|} r(s_t, a_t)$$

  - expectation over trajectories $\tau = (s_0, a_0, s_1, \cdots)$
  - where $s_0 \sim \rho_0,\ a_t \sim \pi(\cdot\,|s_t),\ s_{t+1} \sim P(\cdot\,|s_t, a_t)$
- Assume policy π_θ parametrized by θ
- REINFORCE gradient

$$\mathbb{E}_{\tau\sim\pi_\theta}\left[R(\tau)\nabla_\theta \log \pi_\theta(\tau)\right]$$

- Estimator of the policy gradient based on samples

$$\sum_{\tau\sim\pi_\theta} R(\tau)\nabla_\theta \log \pi_\theta(\tau) \approx \sum_{\tau\sim\pi_\theta}\left[\sum_{t=0}^{|\tau|} R_t \nabla_\theta \log \pi_\theta(a_t|s_t)\right]$$

  - where R(τ) is replaced with $R_t = \sum_{t'=t}^{|\tau|} \gamma^{t'-t} r(s_{t'}, a_{t'})$

4. Off-Policy Correction

- Distribution mismatch
  - the gradient estimation requires sampling trajectories from the policy to update π_θ
  - while the trajectories we collected were drawn from a combination of historical policies β.
  - A naive policy gradient estimator is no longer unbiased
- Importance weighting => unbiased estimation

$$\sum_{\tau \sim \beta} \frac{\pi_\theta(\tau)}{\beta(\tau)} \left[\sum_{t=0}^{|\tau|} R_t \nabla_\theta \log \left(\pi_\theta(a_t|s_t) \right) \right]$$

  - importance weight $\frac{\pi_\theta(\tau)}{\beta(\tau)} = \frac{\rho(s_0) \prod_{t=0}^{|\tau|} \mathbf{P}(s_{t+1}|s_t, a_t) \pi(a_t|s_t)}{\rho(s_0) \prod_{t=0}^{|\tau|} \mathbf{P}(s_{t+1}|s_t, a_t) \beta(a_t|s_t)} = \prod_{t=0}^{|\tau|} \frac{\pi(a_t|s_t)}{\beta(a_t|s_t)}$
  - unbiased estimate if the trajectories are collected with actions sampled according to β
- Huge variance of the estimator
  - chained products leads to very low or high values of the importance weights

Reduce Variance of Off-Policy Gradient

- unbiased estimation w/ high variance $\sum_{\tau\sim\beta}\frac{\pi_\theta(\tau)}{\beta(\tau)}\left[\sum_{t=0}^{|\tau|}R_t\nabla_\theta\log\left(\pi_\theta(a_t|s_t)\right)\right]$
  - importance weight $\frac{\pi_\theta(\tau)}{\beta(\tau)}=\frac{\rho(s_0)\prod_{t=0}^{|\tau|}\mathbf{P}(s_{t+1}|s_t,a_t)\pi(a_t|s_t)}{\rho(s_0)\prod_{t=0}^{|\tau|}\mathbf{P}(s_{t+1}|s_t,a_t)\beta(a_t|s_t)}=\prod_{t=0}^{|\tau|}\frac{\pi(a_t|s_t)}{\beta(a_t|s_t)}$
  - unbiased estimate if the trajectories are collected with actions sampled according to β
- To reduce the variance of each gradient term corresponding to a time t, approximate the importance weights
  - first ignore the terms after time t in the chained products
  - further take a first-order approximation

$$\prod_{t'=0}^{|\tau|}\frac{\pi(a_{t'}|s_{t'})}{\beta(a_{t'}|s_{t'})}\approx\prod_{t'=0}^{t}\frac{\pi(a_{t'}|s_{t'})}{\beta(a_{t'}|s_{t'})}=\frac{P_{\pi_\theta}(s_t)}{P_\beta(s_t)}\frac{\pi(a_t|s_t)}{\beta(a_t|s_t)}\approx\frac{\pi(a_t|s_t)}{\beta(a_t|s_t)}.$$

- a biased estimator of the policy gradient with lower variance

$$\sum_{\tau\sim\beta}\left[\sum_{t=0}^{|\tau|}\frac{\pi_\theta(a_t|s_t)}{\beta(a_t|s_t)}R_t\nabla_\theta\log\pi_\theta(a_t|s_t)\right]$$

Reduce Variance of Off-Policy Gradient

- A biased estimator of the policy gradient with lower variance

$$\sum_{\tau\sim\beta}\left[\sum_{t=0}^{|\tau|}\frac{\pi_\theta(a_t|s_t)}{\beta(a_t|s_t)}R_t\nabla_\theta\log\pi_\theta(a_t|s_t)\right]$$

- Achiam et al. [1] prove that the impact of this first-order approximation on the total reward of the learned policy is bounded in magnitude

$$O\left(E_{s\sim d^\beta}\left[D_{TV}(\pi|\beta)[s]\right]\right)$$

  - where DTV is the total variation between π(·|s) and β(·|s)
  - d^β is the discounted future state distribution under β.

[1] Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. 2017. Constrained policy optimization. arXiv preprint arXiv:1705.10528

4.1 Parametrize the policy π_θ

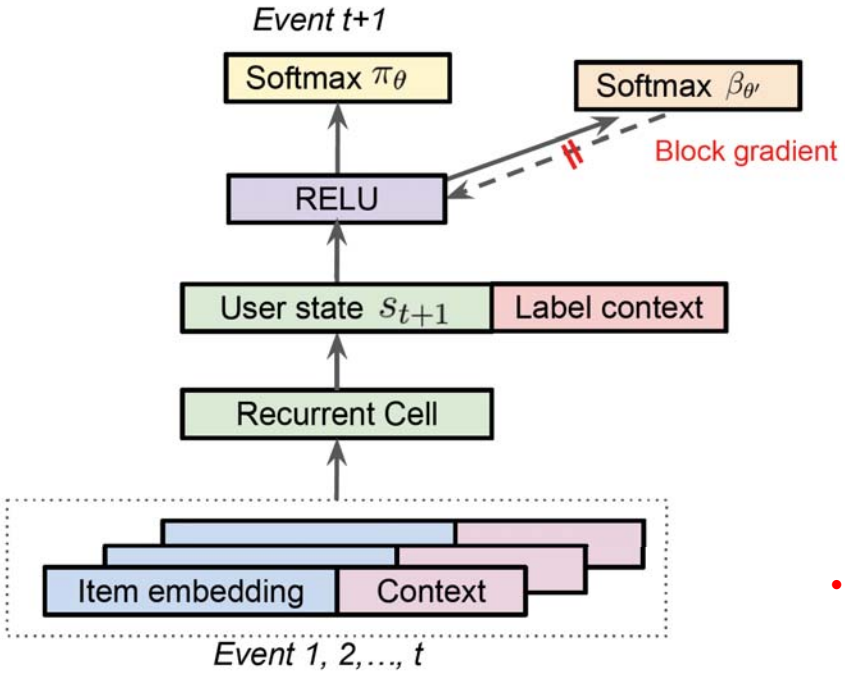


Figure 1: A diagram shows the parametrisation of the policy π_θ as well as the behavior policy $\beta_{\theta'}$.

The state transition is modeled with a recurrent neural network

$$\mathbf{s}_{t+1} = \mathbf{z}_t \odot \tanh(\mathbf{s}_t) + \mathbf{i}_t \odot \tanh(\mathbf{W}_a \mathbf{u}_{a_t})$$
$$\mathbf{z}_t = \sigma(\mathbf{U}_z \mathbf{s}_t + \mathbf{W}_z \mathbf{u}_{a_t} + \mathbf{b}_z)$$
$$\mathbf{i}_t = \sigma(\mathbf{U}_i \mathbf{s}_t + \mathbf{W}_i \mathbf{u}_{a_t} + \mathbf{b}_i)$$

where $\mathbf{z}_t, \mathbf{i}_t \in \mathbb{R}^n$ are the update and input gate respectively.

Conditioning on a user state $\mathbf{s}$, the policy $\pi_\theta(a|\mathbf{s})$ is then modeled with a simple softmax,

$$\pi_\theta(a|\mathbf{s}) = \frac{exp(\mathbf{s}^\top \mathbf{v}_a / T)}{\sum_{a' \in \mathcal{A}} exp(\mathbf{s}^\top \mathbf{v}_{a'} / T)} \quad (5)$$

- where $v_a \in R^n$ is another embedding for each action a in the action space A
- T is a temperature that is normally set to 1.

- 4.2 Estimating the behavior policy β
  - re-use the user states generated from the RNN model
    - model policy β with another softmax layer.
  - prevent the behavior head from intefering with the user state of the main policy
    - block its gradient from flowing back into the RNN

4.3 Top-K Off-Policy Correction

- Challenge
  - Policy $\Pi_\Theta(A|s)$: action A is to select a set of k items.
  - Recommend a page of k items to users at a time.
- Objective $\max_\Theta \mathbb{E}_{\tau\sim\Pi_\theta}\left[\sum_t r(s_t, A_t)\right]$
  - expectation over trajectories $\tau = (s_0, A_0, s_1, \cdots)$
  - where $s_0 \sim \rho_0$, $A_t \sim \Pi(\cdot|s_t)$, $s_{t+1} \sim P(\cdot|s_t, a_t)$
- Assume
  - the expected reward of a set of non-repetitive items equal to the sum of the expected reward of each item in the set.
  - generate the set action A by independently sampling each item a according to the softmax policy π_θ and then de-duplicate.

$$\Pi_\Theta(A'|s) = \prod_{a\in A'} \pi_\theta(a|s)$$

- Adapt the REINFORCE algorithm to the set recommendation

$$\sum_{\tau\sim\pi_\theta}\left[\sum_{t=0}^{|\tau|} R_t \nabla_\theta \log \alpha_\theta(a_t|s_t)\right]$$

where $\alpha_\theta(a|s) = 1-(1-\pi_\theta(a|s))^K$ is the probability that an item a appears in the final non-repetitive set A.

4.3 Top-K Off-Policy Correction

- a biased estimator of the policy gradient with lower variance

$$\sum_{\tau\sim\beta}\left[\sum_{t=0}^{|\tau|}\frac{\alpha_\theta(a_t|s_t)}{\beta(a_t|s_t)}R_t\nabla_\theta\log\alpha_\theta(a_t|s_t)\right] \tag{6}$$

$$=\sum_{\tau\sim\beta}\left[\sum_{t=0}^{|\tau|}\frac{\pi_\theta(a_t|s_t)}{\beta(a_t|s_t)}\frac{\partial\alpha(a_t|s_t)}{\partial\pi(a_t|s_t)}R_t\nabla_\theta\log\pi_\theta(a_t|s_t)\right].$$

where $\alpha_\theta(a|s) = 1-(1-\pi_\theta(a|s))^K$

- top-K policy adds an additional multiplier to the original off-policy correction factor

$$\lambda_K(s_t,a_t)=\frac{\partial\alpha(a_t|s_t)}{\partial\pi(a_t|s_t)}=K(1-\pi_\theta(a_t|s_t))^{K-1} \tag{7}$$

As $\pi_\theta(a|s)\to 0, \lambda_K(s,a)\to K.$ increases the policy update by a factor of K; (add likelihood for small-mass items)

As $\pi_\theta(a|s)\to 1, \lambda_K(s,a)\to 0.$ zeros out the policy update. (no longer increase likelihood if the desirable item already has reasonable mass, i.e., likely to appear in the top-K)

=> allow other items of interest to take up some mass in the softmax policy.

4.4 Variance Reduction Techniques

- Importance weight: $\omega(s,a) = \frac{\pi(a|s)}{\beta(a|s)}$
- Large importance weight due to:
  - large deviation of the new policy π from the behavior policy
    - the new policy explores regions that are less explored by the behavior policy.
    - π(a|s) ≫ β(a|s)
  - large variance in the β estimate

Weight Capping. The first approach we take is to simply cap the weight [8] as

$$\bar{\omega}_c(s,a) = \min\left(\frac{\pi(a|s)}{\beta(a|s)}, c\right). \quad (8)$$

Smaller value of c reduces variance in the gradient estimate, but introduces larger bias.

Normalized Importance Sampling (NIS). Second technique we employed is to introduce a ratio control variate, where we use classical weight normalization [32] defined by:

$$\bar{\omega}_n(s,a) = \frac{\omega(s,a)}{\sum_{(s',a')\sim\beta}\omega(s',a')}.$$

As $\mathbb{E}_\beta[\omega(s,a)] = 1$, the normalizing constant is equal to n, the batch size, in expectation. As n increases, the effect of NIS is equivalent to tuning down the learning rate.

- Trusted Region Policy Optimization (TRPO). TRPO [36]
  - prevents the new policy π from deviating from the behavior policy
  - by adding a regularization that penalizes the KL divergence of these two policies.
  - It achieves similar effect as the weight capping.

5 Exploration

- Boltzmann exploration
  - get the benefit of exploratory data without negatively impacting user experience
- consider using a stochastic policy where recommendations are sampled from π_θ rather than taking the K items with the highest probability.
  - efficient approximate nearest neighbor-based systems to look up the top M items in the softmax
  - feed the logits of these M items into a smaller softmax to normalize the probabilities and sample from this distribution.
  - $M \gg K$, we can still retrieve most of the probability mass, limit the risk of bad recommendations, and enable computationally efficient sampling.
- In practice
  - we further balance exploration and exploitation by returning the top K' most probable items and sample $K - K'$ items from the remaining $M - K'$ items.

Contributions

- REINFORCE Recommender:
  - We scale a REINFORCE policy gradient-based approach to learn a neural recommendation policy in a extremely large action space.
- Off-Policy Candidate Generation:
  - We apply off-policy correction to learn from logged feedback, collected from an ensemble of prior model policies.
  - We incorporate a learned neural model of the behavior policies to correct data biases.
- Top-K Off-Policy Correction:
  - We offer a novel top-K off policy correction to account for the fact that our recommender outputs multiple items at a time.
- Benefits in Live Experiments:
  - We demonstrate in live experiments, which was rarely done in existing RL literature, the value of these approaches to improve user long term satisfaction.