Reinforcement Learning to Rank in E-Commerce Search Engine: Formalization, Analysis, and Application

2/28/2022

CS 885 Reinforcement Learning

Jared Feng
Department of Statistics and Actuarial Science

UNIVERSITY OF WATERLOO

INTRODUCTION

Why is it important for e-commerce

- To develop algorithms that optimize how to rank items shown in the users' search query
- Users get what they want using the search engine
- Boost the total amount of transactions

Learn to rank (LTR) paradigm

- **Supervised Learning**
- Pointwise: evaluate each item separately
- Pairwise: compare two items
- Listwise: directly output the entire order
- Labeled data has high costs

- **(Partial) RL**
- Bandits: assumes only one state
- Only considers one round of action-and-feedback

General workflow of a search session

1. User inputs a query
2. Search engine ranks all related items & displays the top K (e.g. K=10) items
3. User clicks on & buys a shown item, or abandons the search, or requests the next page
4. Search engine re-ranks the remaining items & displays the top K ones, whenever requested again

PROBLEM FORMULATION

Formalize some basic concepts

- D: the finite set of all items related to the query
- $\mathcal{L}_K(D, f)$: the **top K list** where the items in the items set D are ordered according to the pointwise ranking function f, and the top K ones are returned
- p_t: **item page**, the output of the top K list at step t
- D_t: the set of remaining items at step t, $D_0 = D$, $D_t = D_{t-1} \setminus p_t$
- h_t: **item page history** at step t, $h_0 = q$ (user's query), $h_t = h_{t-1} \cup \{p_t\}$

UNIVERSITY OF WATERLOO

Formalize some basic concepts

- $B(h_t)$: an event that user buys an item given h_t
- $L(h_t)$: an event that user leaves the session given h_t
- $C(h_t)$: an event that user continues to the next page given h_t
- $b(h_t)$: average probability that $B(h_t)$ occurs
- Likewise, we have $l(h_t), c(h_t)$
- $b(h_t) + l(h_t) + c(h_t) = 1$, and $c(h_0) = 1$ always true

Search Session MDP (SSMDP)

- $T = \left\lceil \frac{|D|}{K} \right\rceil$: maximal number of steps of a session
- $\mathcal{H} = \cup_{t=0}^{T} \mathcal{H}_t$: the set of all possible histories, where $\mathcal{H}_t$ is the set of all possible histories up to t $(0 < t \leq T)$
- $\mathcal{H}_C = \{C(h_t) | \forall h_t \in \mathcal{H}_t, 0 < t \leq T\}$, likewise for $\mathcal{H}_B, \mathcal{H}_L$
- $\mathcal{S} = \mathcal{H}_C \cup \mathcal{H}_B \cup \mathcal{H}_L$
- $\mathcal{A}$: the action space that contains all possible ranking functions
- $\mathcal{R}: \mathcal{H}_C \times \mathcal{A} \times \mathcal{S} \rightarrow \mathbb{R}$ is the reward function
- $\mathcal{P}: \mathcal{H}_C \times \mathcal{A} \times \mathcal{S} \rightarrow [0,1]$ is the state transition function

Modeling rewards & transitions

- $\mathcal{P}(C(h_t), a, s') = \begin{cases} b(h_{t+1}), & if\ s' = B(h_{t+1}) \\ l(h_{t+1}), & if\ s' = L(h_{t+1}) \\ c(h_{t+1}), & if\ s' = C(h_{t+1}) \end{cases}$
- $\mathcal{R}(C(h_t), a, s') = \begin{cases} m(h_{t+1}), & if\ s' = B(h_{t+1}) \\ 0, & otherwise \end{cases}$
- $m(h_t)$ is the expected deal price given h_t

ANALYSIS OF SSMDP

Markov property in SSMDP

$$
\begin{aligned}
&\Pr(s_t | s_0, a_0, s_1, a_1, ..., s_{t-1}, a_{t-1}) \\
=&\Pr(s_t | C(h_0), a_0, C(h_1), a_1, ..., C(h_{t-1}), a_{t-1}) \\
=&\Pr(s_t | h_1, h_2, ..., h_{t-1}, C(h_{t-1}), a_{t-1}) \\
=&\Pr(s_t | h_{t-1}, C(h_{t-1}), a_{t-1}) \\
=&\Pr(s_t | C(h_{t-1}), a_{t-1}) \\
=&\Pr(s_t | s_{t-1}, a_{t-1}).
\end{aligned}
$$

Rewards & states' value analysis

- Simplified notations: $C_t^\pi \leftarrow C(h_t^\pi)$, where h_t^π is the item page history generated under policy π at step t.
- $$V_\gamma^\pi(C_t^\pi) = \mathbb{E}^\pi\{r_{t+1} + \gamma r_{t+2} + \cdots + \gamma^{T-t-1} r_T | C_t^\pi\}$$
  $$= \sum_{k=1}^{T-t} \gamma^{k-1} \Pr(C_t^\pi \rightarrow h_{t+k}^\pi)\, b_{t+k}^\pi m_{t+k}^\pi$$
  $$= b_{t+1}^\pi m_{t+1}^\pi + \sum_{k=2}^{T-t} \gamma^{k-1} ((\prod_{j=1}^{k-1} c_{t+j}^\pi) b_{t+k}^\pi m_{t+k}^\pi)$$
- In real scenarios, often γ is set to 1

ALGORITHM

Gradient for the policy network

- Parameter of policy network: θ
- Objective $J(\theta)$: the expected sum of rewards with $\gamma = 1$

- Basic REINFORCE: $$\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim \rho_\theta} \left\{ \sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(s_t, a_t) R_t^T(\tau) \right\}$$

- Generalized REINFORCE: $$\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim \rho_\theta} \left\{ \sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(s_t, a_t) Q^{\pi_\theta}(s_t, a_t) \right\},$$

- Deterministic Policy Grad.: $$\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim \rho_\theta} \left\{ \sum_{t=0}^{T-1} \nabla_\theta \pi_\theta(s_t) \nabla_a Q^{\pi_\theta}(s_t, a) \Big|_{a=\pi_\theta(s_t)} \right\}.$$

Gradient for the value network

- Parameter of policy network: w
- Objective $MSE(w) = \sum_{s\in\mathcal{S}} \sum_{a\in\mathcal{A}} \left(Q^{w}(s,a) - Q^{\pi_\theta}(s,a)\right)^2$
- Gradient: $\nabla_w MSE(w) = \sum_{s\in\mathcal{S}} \sum_{a\in\mathcal{A}} \left(Q^{\pi_\theta}(s,a) - Q^{w}(s,a)\right)\nabla_{\mathrm{w}} \mathrm{Q}^{\mathrm{w}}(s,a)$
- Given (s, a, s') where $s' = C(h')$, estimate $Q^{\pi_\theta}(s,a) \approx b(h')m(h') + c(h') \max_{a'} Q^{w}(s', a')$

UNIVERSITY OF WATERLOO

Psudeocode

Deterministic Policy Gradient with Full Backup Estimation (DPG-FBE)

Input: Learning rate α_θ and α_w, pretrained conversion probability model b, continuing probability model c, and expected deal price model m of item page histories

Initialize the actor π_θ and the critic Q^w with parameter θ and w;

foreach *search session* **do**

- Use π_θ to sample a ranking action at each step with exploration;
- Get the trajectory τ of the session with its final step index t;
- $\Delta w \leftarrow 0$, $\Delta\theta \leftarrow 0$;
- **for** $k = 0, 1, 2, \ldots, t-1$ **do**
    - $(s_k, a_k, r_k, s_{k+1}) \leftarrow$ the sample tuple at step k;
    - $h_{k+1} \leftarrow$ the item page history of s_k;
    - **if** $s_{k+1} = B(h_{k+1})$ **then**
        - Update the models b, c, and m with the samples $(h_{k+1}, 1)$, $(h_{k+1}, 0)$, and (h_{k+1}, r_k), respectively;
    - **else**
        - Update the models b and c with the samples $(h_{k+1}, 0)$ and $(h_{k+1}, 1)$, respectively;
    - $s' \leftarrow C(h_{k+1})$, $a' \leftarrow \pi_\theta(s')$;
    - $p_{k+1} \leftarrow b(h_{k+1})m(h_{k+1})$;
    - $\delta_k \leftarrow p_{k+1} + c(h_{k+1})Q^w(s', a') - Q^w(s_k, a_k)$;
    - $\Delta w \leftarrow \Delta w + \alpha_w \delta_k \nabla_w Q^w(s_k, a_k)$;
    - $\Delta\theta \leftarrow \Delta\theta + \alpha_\theta \nabla_\theta \pi_\theta(s_k) \nabla_a Q^w(s_k, a_k)$;
- $w \leftarrow w + \Delta w / t$, $\theta \leftarrow \theta + \Delta\theta / t$;

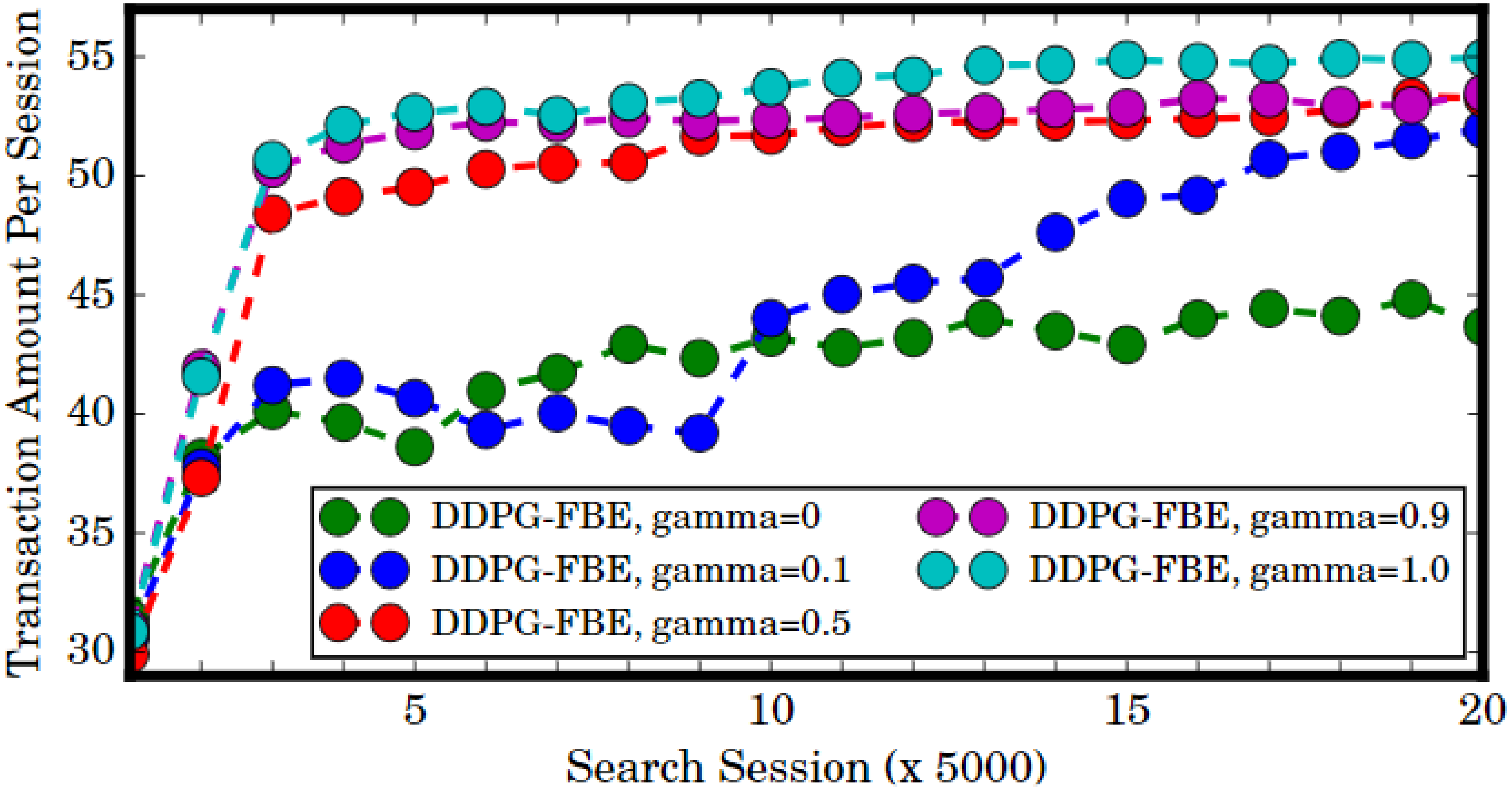
EXPERIMENTS

Simulation setup

- User behaviors of leaving, continuing, and buying simulated independently
- Info of each item converted to a vector of size 20 (20 most important features)
- Ranking action is also a vector of size 20 (to apply dot product)
- Feature extraction applied to item page histories & user behaviors
- 1000 items sampled from the dress category items on Taobao

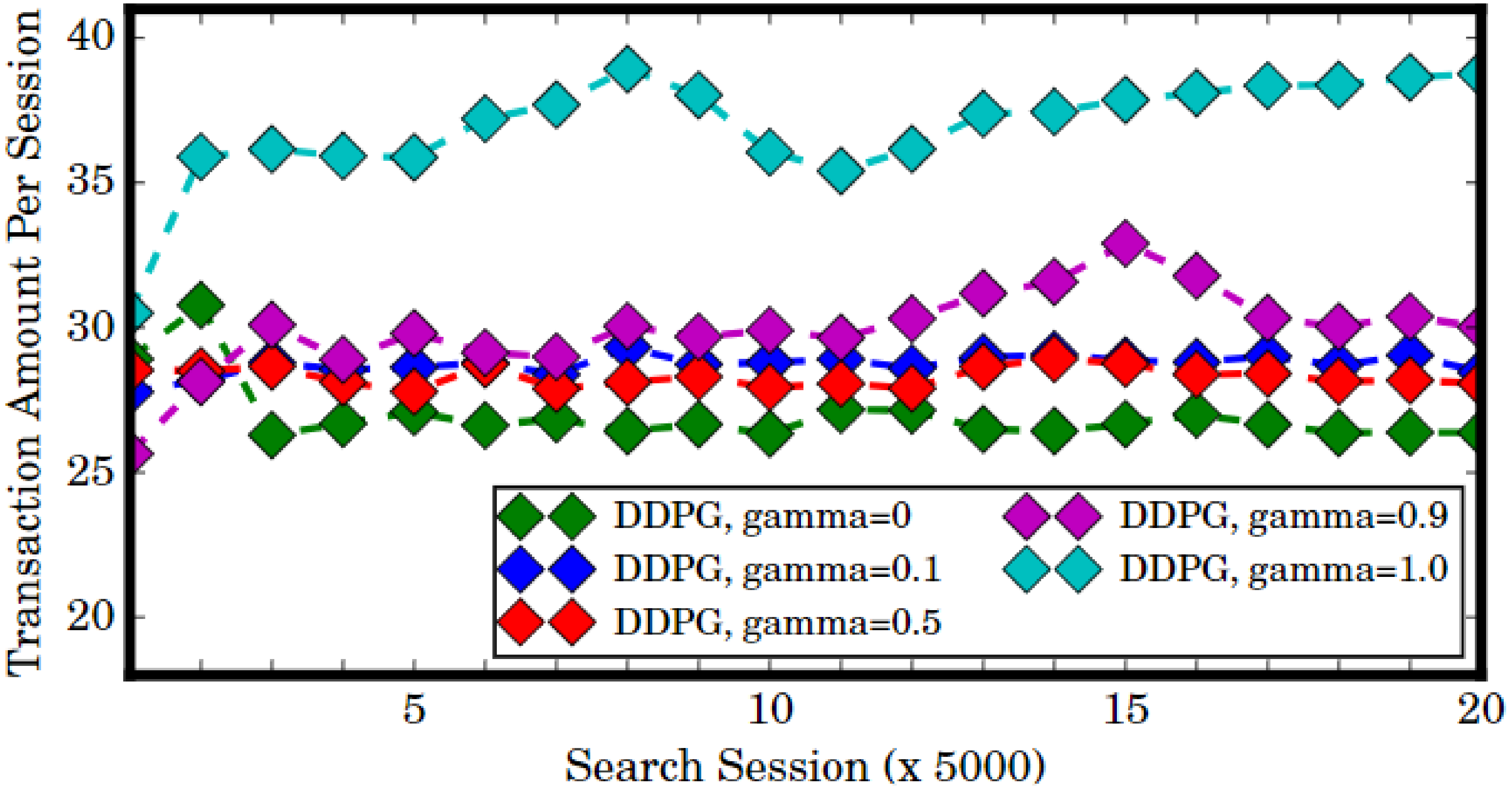
Simulation results

The learning performance of the DDPG-FBE algorithm in the simulation experiment



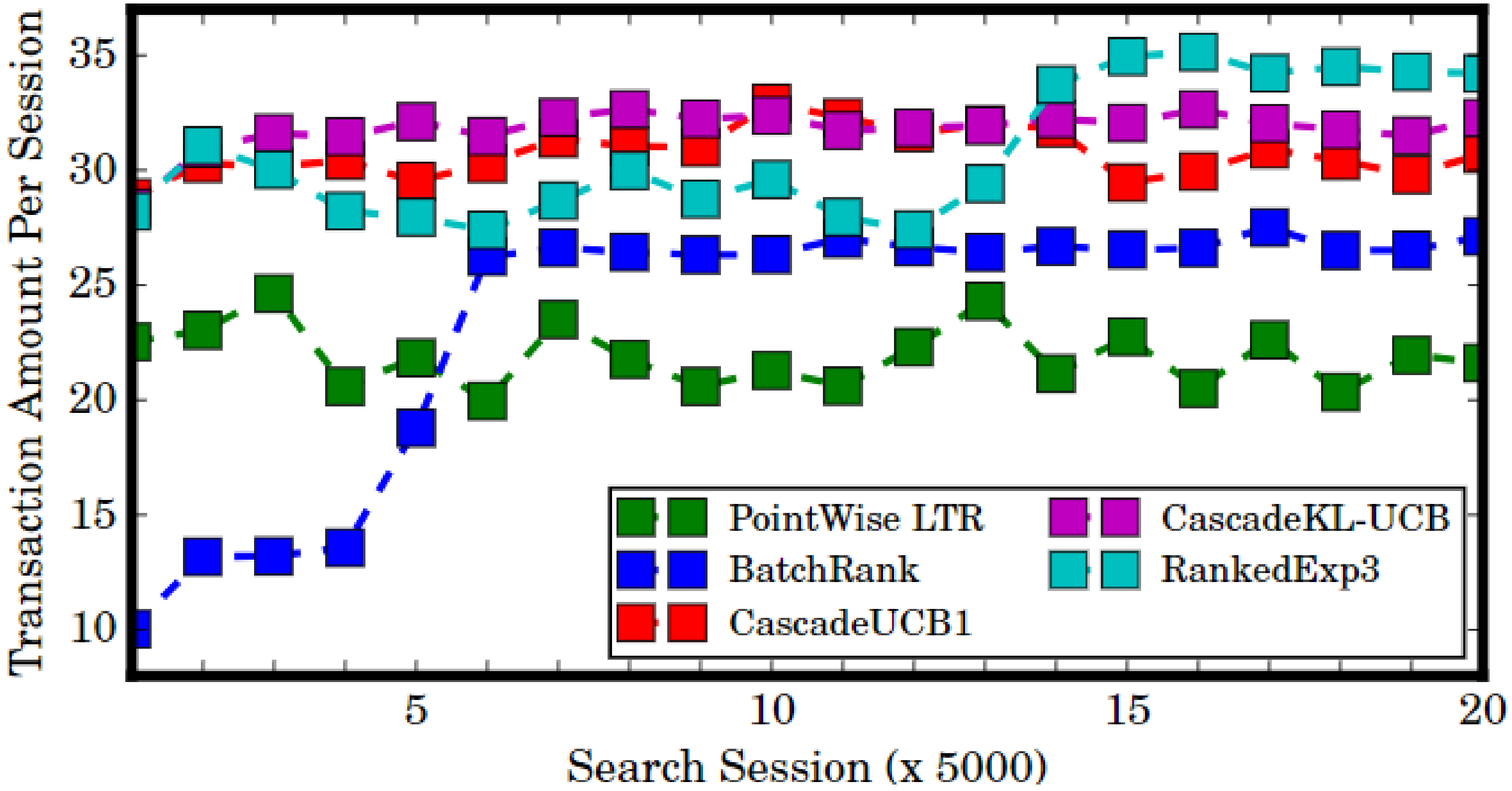
Simulation results

The learning performance of the DDPG algorithm in the simulation experiment



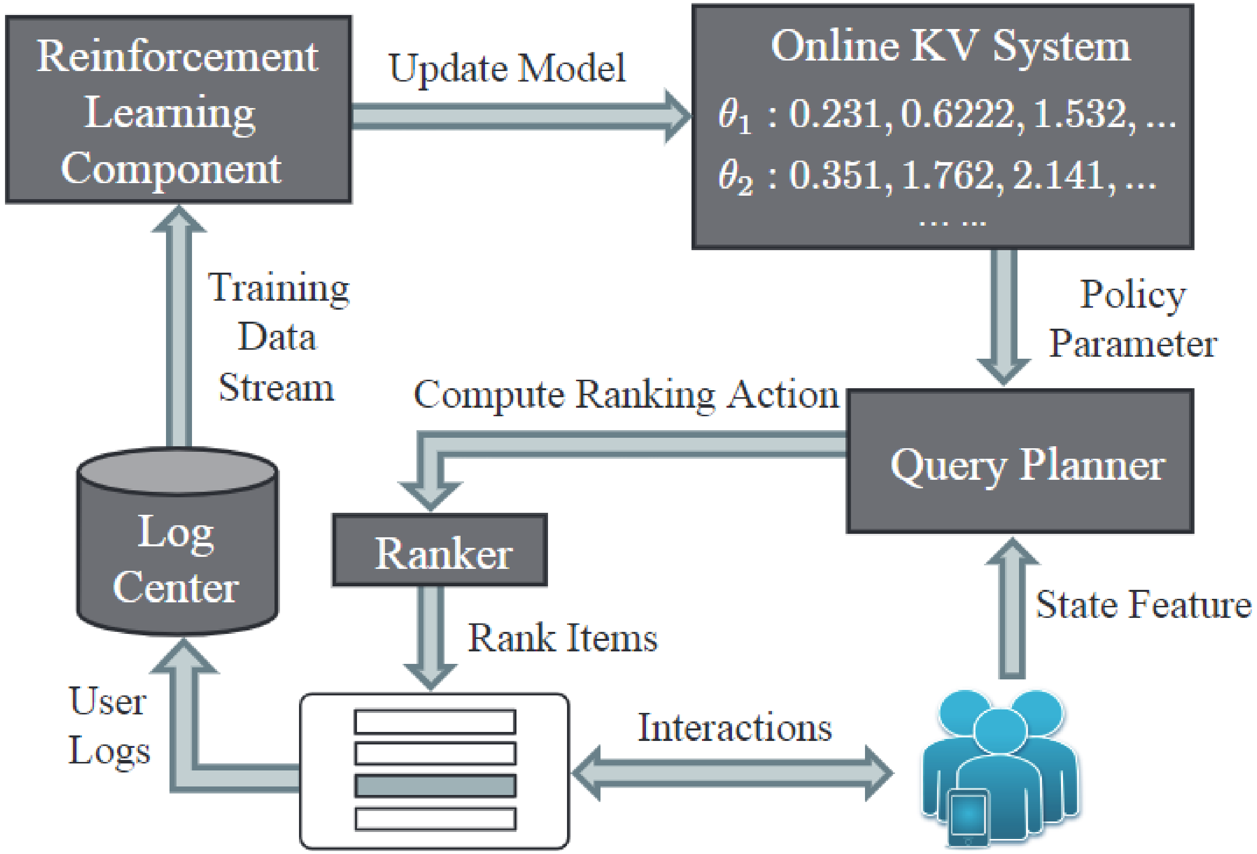
Simulation results

The learning performance of five online LTR algorithms in the simulation experiment



UNIVERSITY OF WATERLOO

Application: RL ranking system on Taobao



- Neural nets compressed to deal with constraints
- One-week A/B test between DDPG and DDPG-FBE: +2.7-4.3% daily amount out of DDPG-FBE
- 2016 Double 11 Shopping Festival: DDPG-FBE brought 30%+ growth in total transaction value

Conclusion

- Formalize the e-commerce item search problem into an extension of MDP, SSMDP
- Analyze the property of SSMDP
- Novel RL algorithm for learning a ranking policy under SSMDP

Thank you for listening!