Deep Reinforcement Learning for Page-wise Recommendations

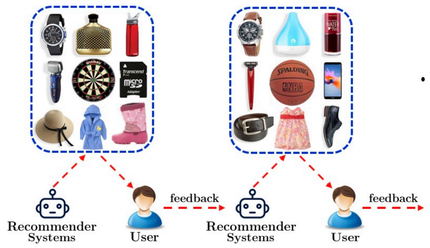
Xiangyu Zhao[1], Long Xia[2], Liang Zhang[2], Zhuoye Ding[2], Dawei Yin[2] and Jiliang Tang[1]
1: Data Science and Engineering Lab, Michigan State University 2: Data Science Lab, JD.com

Data Science and Engineering Lab

My homepage

Motivation

Recommender systems can mitigate the information overload problem by suggesting users' personalized items

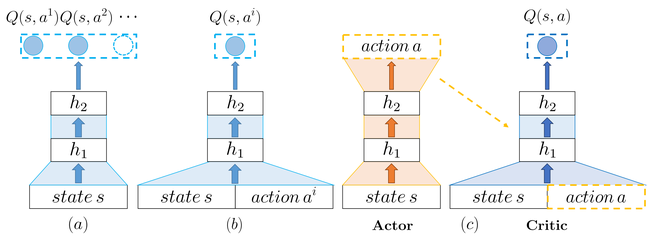


- **Interactions between recommender systems and users**
  - Each time the system recommends a page of items to the user
  - Next the user browses these items and provides real-time feedback
  - Then the system recommends a new page of items...
- **Two key problems**
  - How to efficiently capture user's **dynamic preference** and update recommending strategy according to user's **real-time feedback**
  - How to generate **a page of items with proper display** based on user's preferences

Framework Overview

We model the recommendation task as a MDP and leverage **Reinforcement Learning** to automatically learn the optimal recommendation strategies

1. Framework Architecture Selection



- ✓ **Actor-Critic Framework**
  - ✓ Actor: current state -> a deterministic action (recommend a deterministic page of items)
  - ✓ Critic: this state-action pair -> Q-value $Q(s,a)=\mathbb{E}_{s'}\left[r+\gamma Q(s',a')|s,a\right]$
  - ✗ (a) cannot handle large and dynamic action space scenario
  - ✗ (b) high computational cost $Q^*(s,a)=\mathbb{E}_{s'}\left[r+\gamma\max_{a'}Q^*(s',a')|s,a\right]$

2. Markov Decision Process

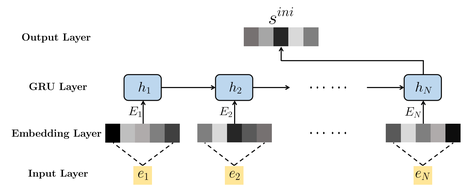
- State space $\mathcal{S}$: A state $s\in\mathcal{S}$ is defined as user's current preference, which is generated based on user's browsing history, i.e., the items that a user browsed and her corresponding feedback.
- Action space $\mathcal{A}$: An action $a=\{a^1,\cdots,a^M\}\in\mathcal{A}$ is to recommend a page of M items to a user based on current state s.
- Reward $\mathcal{R}$: After the RA takes an action a at the state s, i.e., recommending a page of items to a user, the user browses these items and provides her feedback. She can skip (not click), click, or order these items, and the agent receives immediate reward $r(s,a)$ according to the user's feedback.
- Transition $\mathcal{P}$: Transition $p(s'|s,a)$ defines the state transition from s to s' when RA takes action a.
- Discount factor γ: $\gamma\in[0,1]$ defines the discount factor when we measure the present value of future reward. In particular, when $\gamma=0$, RA only considers the immediate reward. In other words, when $\gamma=1$, all future rewards can be counted fully into that of the current action.

DeepPage Framework

1. Architecture of Actor Framework

- The Actor is designed to **generate a page of recommendations** according to user's preference, which needs to tackle **three challenges**
  - Setting an initial preference at the beginning of a new recommendation session
  - Learning the real-time preference in the current session
  - Jointly generating a set of recommendations and displaying them in a 2-D page

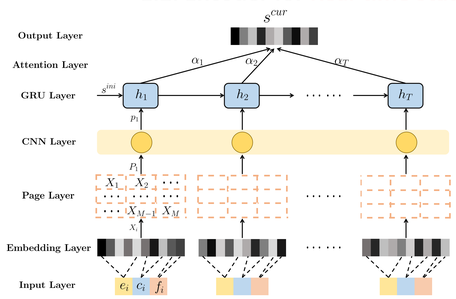
1.1. Encoder for Initial State Generation Process



$$z_t=\sigma(W_zE_t+U_zh_{t-1})$$
$$r_t=\sigma(W_rE_t+U_rh_{t-1})$$
$$\hat{h}_t=\tanh[WE_t+U(r_t\cdot h_{t-1})]$$
$$h_t=(1-z_t)h_{t-1}+z_t\hat{h}_t$$
$$s^{ini}=h_t$$

- Setting an **initial preference** at the beginning of a new session
  - **Inputs**: user's last clicked/ordered items before the current session
  - **Output**: the representation of users' initial preference by a vector

1.2. Encoder for Real-time State Generation Process



$$s^{cur}=\sum_{t=1}^{T}\alpha_th_t \qquad \alpha_t=\frac{exp(W_ah_t+b_a)}{\sum_j exp(W_ah_j+b_a)}$$

$$p_t=conv2d(P_t)$$

Convolution Neural Network

$$X_i=concat(E_i,C_i,F_i)$$
$$=\tanh\left(concat(W_Ee_i+b_E,W_Cc_i+b_C,W_Ff_i+b_F)\right)$$

$e_i: item\ representation$
$c_i: item's\ category$
$f_i: user's\ feedback$

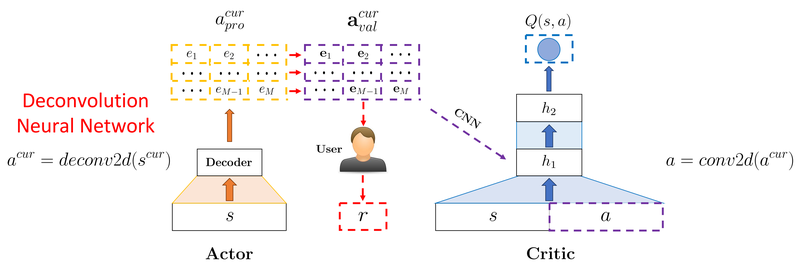
- Learning the **real-time preference** in the current session
  - **Inputs**: item representation, item's category (to generate complementary recommendations) and user's feedback (to capture user's interests in current page)
  - **Output**: the representation of users' real-time preference by a vector

1.3. Decoder for Action Generation Process

- Jointly generating a set of recommendations and displaying them in a page
  - **Inputs**: users' real-time preference
  - **Output**: the representation of a page of recommendations (items) with proper display

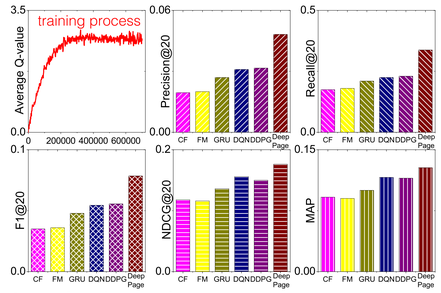
2. The Architecture of Critic Framework

- Critic aims to learn an action-value function *Q(s,a)*, which is a judgment of whether the action "*a*" matches the current state "*s*"
  - **Inputs**: users' real-time preference "*s*" and a page of recommendations (items) "*a*"
  - **Output**: Q-value, i.e., *Q(s,a)*



Experiments

1. How DeepPage performs compared to representative baselines?



- Dataset from JD.com
- Baselines
  - Collaborative filtering
  - Factorization Machines
  - GRU for Rec
  - DQN
  - DDPG

- It can be observed:
  - CF and FM perform worse than other baselines. These two baselines ignore the temporal sequence of the users' browsing history
  - DQN and DDPG outperform GRU. We design GRU to maximize the immediate reward for recommendations, while DQN and DDPG are designed to achieve the trade-off between short-term and long-term rewards
  - DeepPage performs better than conventional DDPG. Compared to DDPG, DeepPage jointly optimizes a page of items and uses GRU to learn user's real-time preference

2. How the components in Actor and Critic contribute to the performance?

Methods	Precision @20	Recall @20	F1score @20	NDCG @20	MAP
DeepPage-1	0.0479	0.3351	0.0779	0.1753	0.1276
DeepPage-2	0.0475	0.3308	0.0772	0.1737	0.1265
DeepPage-3	0.0351	0.2627	0.0578	0.1393	0.1071
DeepPage-4	0.0452	0.3136	0.0729	0.1679	0.1216
DeepPage-5	0.0476	0.3342	0.0775	0.1716	0.1243
DeepPage-6	0.0318	0.2433	0.0528	0.1316	0.1039
DeepPage-7	0.0459	0.3179	0.0736	0.1698	0.1233
DeepPage	**0.0491**	**0.3576**	**0.0805**	**0.1872**	**0.1378**

- 1: no embedding-layers
- 2: no category and feedback
- 3: no GRU for initial state
- 4: no CNNs
- 5: no attention layers
- 6: no GRU for real-time state
- 7: no DeCNN

- We make following observations:
  - DeepPage-1 and DeepPage-2 validate that incorporating category/feedback information and the embedding layer can boost the quality of recommendation
  - DeepPage-3 and DeepPage-6 suggest that setting user's initial preference, and capturing user's real-time preference is helpful for accurate recommendations
  - DeepPage outperforms DeepPage-4 and DeepPage-7, which indicates that item display strategy can influence the decision making process of users

Conclusion and Future Work

- **Conclusion**
  - We propose a novel Actor-Critic framework, which can be applied in scenarios with large and dynamic item space and can reduce computational cost significantly
  - DeepPage framework is able to capture the real-time preference and optimize a page of items jointly, which can boost recommendation performance
- **Future Work**
  - We would like to handle multiple tasks such as search, bidding, advertisement and recommendation collaboratively in one reinforcement learning framework
  - We would like to validate with more agent-user interaction patterns, e.g., adding items into shopping cart
- **Acknowledgements**
  - This research is supported by National Science Foundation (IIS-1714741, IIS-1715940) and Criteo Faculty Research Award