Model-based Reinforcement Learning and Its Potential Use in IR

Weinan Zhang

Shanghai Jiao Tong University

http://wnzhang.net

Jul. 15, 2021

DRL4IR Workshop, SIGIR’21

Content

1. A brief of model-based reinforcement learning
2. User click model and item ranking in recommendation
3. Future research on this direction

A Perspective of

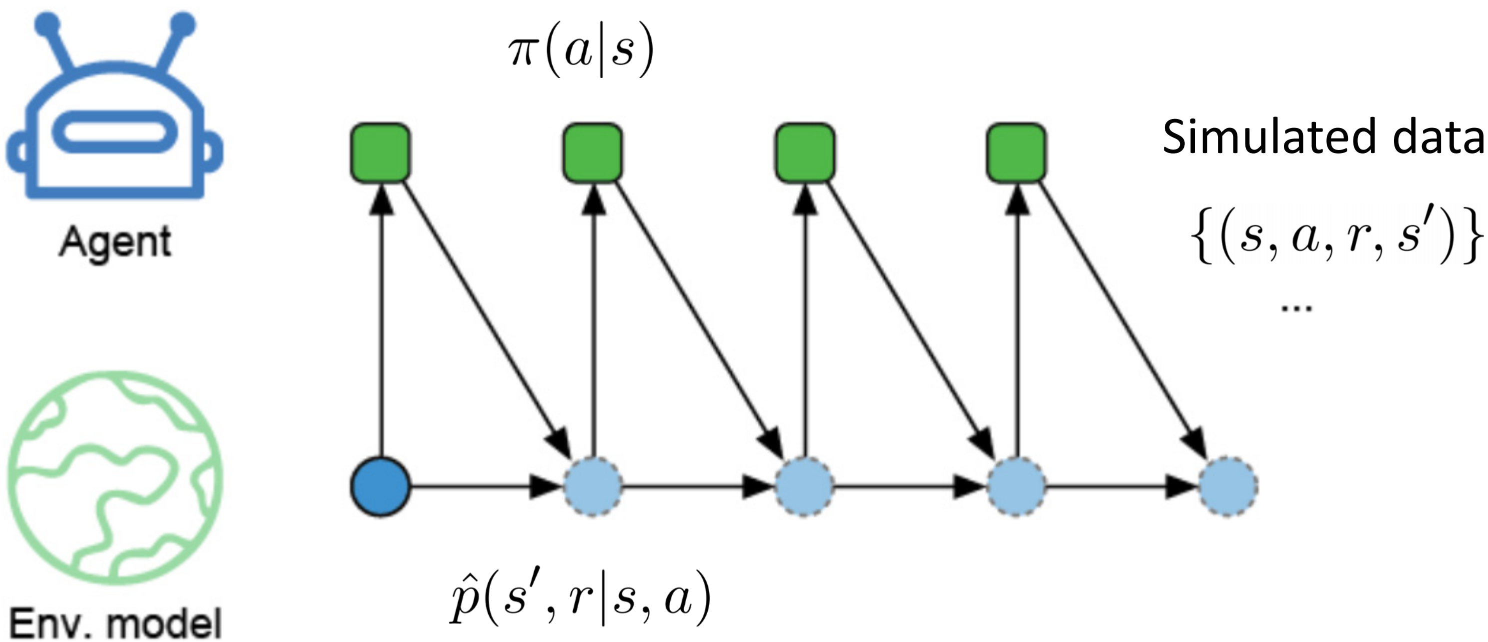
Overall Pathway of DRL

- Deep reinforcement learning gets appealing success
  - Atari, AlphaGo, DOTA 2, AlphaStar

- But DRL has very low data efficiency
  - Trial-and-error learning for deep networks

- A recent popular direction is model-based RL
  - Build a model $p(s', r|s, a)$
  - Based on the model to train the policy
  - So that the data efficiency could be improved

Interaction between Agent and Env. Model



- Real environment
  - State dynamics $p(s'|s,a)$
  - Reward function $r(s,a)$
- Environment model
  - State dynamics $\hat{p}(s'|s,a)$
  - Reward function $\hat{r}(s,a)$

Q-Planning

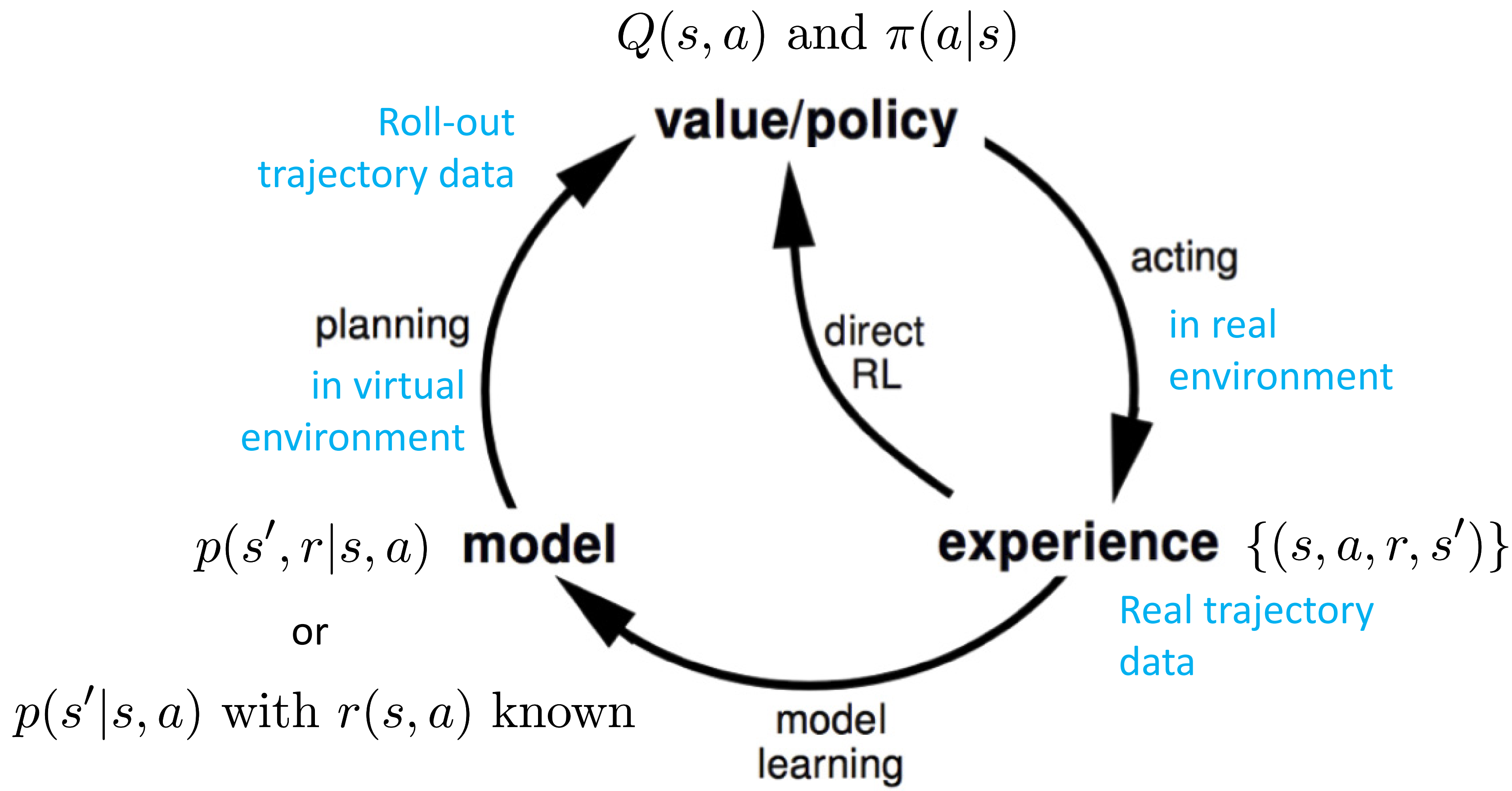
- Random-sample one-step tabular Q-planning
  - First, learn a model *p*(*s'*,*r*|*s*,*a*) from experience data
  - Then perform one-step sampling by the model to learn the Q function

Do forever:

1. Select a state, $S \in \mathcal{S}$, and an action, $A \in \mathcal{A}(s)$, at random
2. Send S, A to a sample model, and obtain
   a sample next reward, R, and a sample next state, S'
3. Apply one-step tabular Q-learning to S, A, R, S':
   $Q(S,A) \leftarrow Q(S,A) + \alpha\big[R + \gamma \max_a Q(S',a) - Q(S,A)\big]$

- Here model learning and reinforcement learning are separate

Dyna



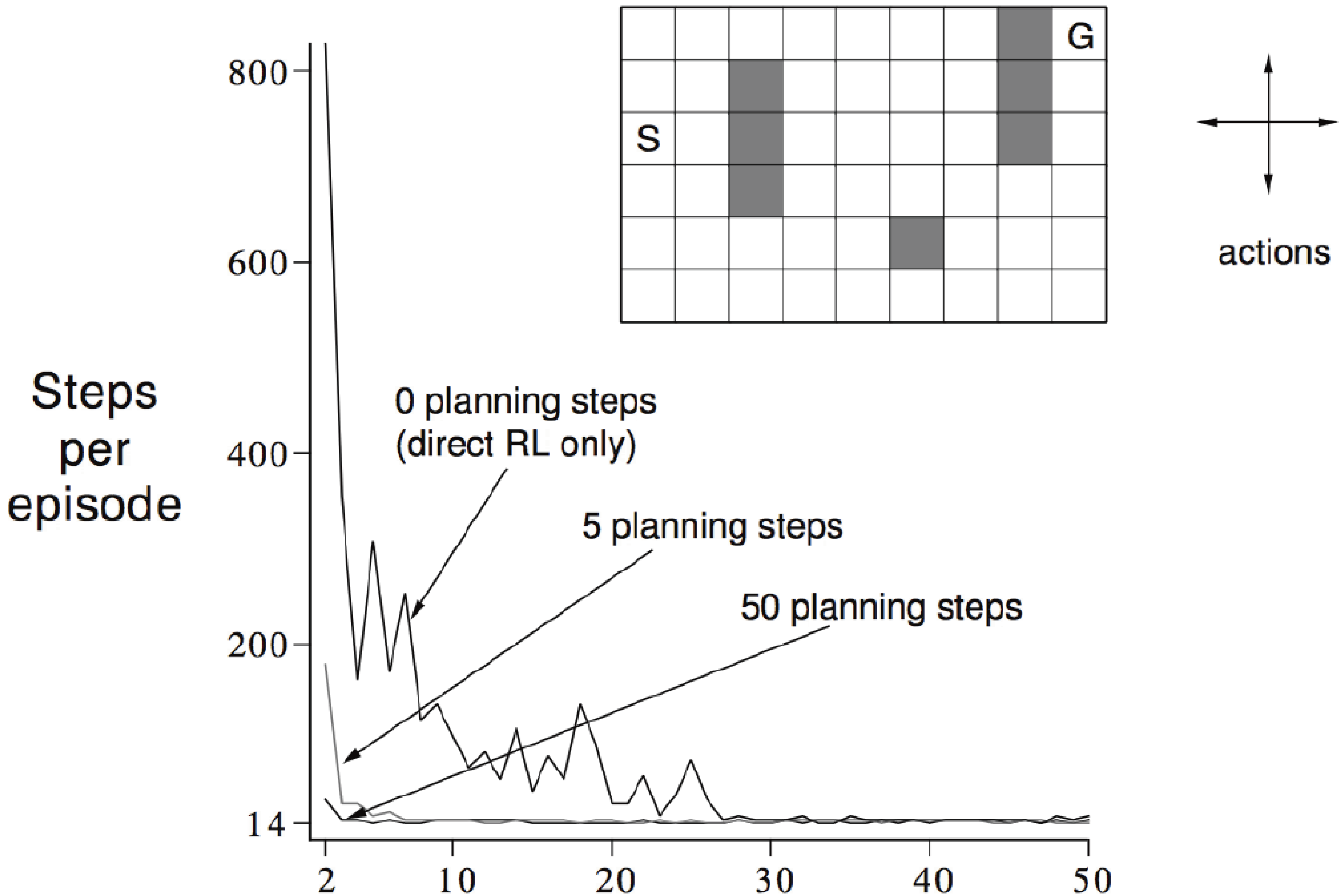
Annotations based on Rich Sutton's figure

Dyna-Q

Initialize $Q(s,a)$ and $Model(s,a)$ for all $s \in \mathcal{S}$ and $a \in \mathcal{A}(s)$
Do forever:
  (a) $S \leftarrow$ current (nonterminal) state
  (b) $A \leftarrow \varepsilon$-greedy(S, Q)
  (c) Execute action A; observe resultant reward, R, and state, S'
  (d) $Q(S,A) \leftarrow Q(S,A) + \alpha\left[R + \gamma \max_a Q(S',a) - Q(S,A)\right]$
  (e) $Model(S,A) \leftarrow R, S'$ (assuming deterministic environment)
  (f) Repeat n times:
      $S \leftarrow$ random previously observed state
      $A \leftarrow$ random action previously taken in S
      $R, S' \leftarrow Model(S,A)$
      $Q(S,A) \leftarrow Q(S,A) + \alpha\left[R + \gamma \max_a Q(S',a) - Q(S,A)\right]$

Sutton, Richard S. "Integrated architectures for learning, planning, and reacting based on approximating dynamic programming." *Machine Learning Proceedings 1990*. Morgan Kaufmann, 1990. 216-224.

Dyna-Q on a Simple Maze



Key Questions of Model-based RL

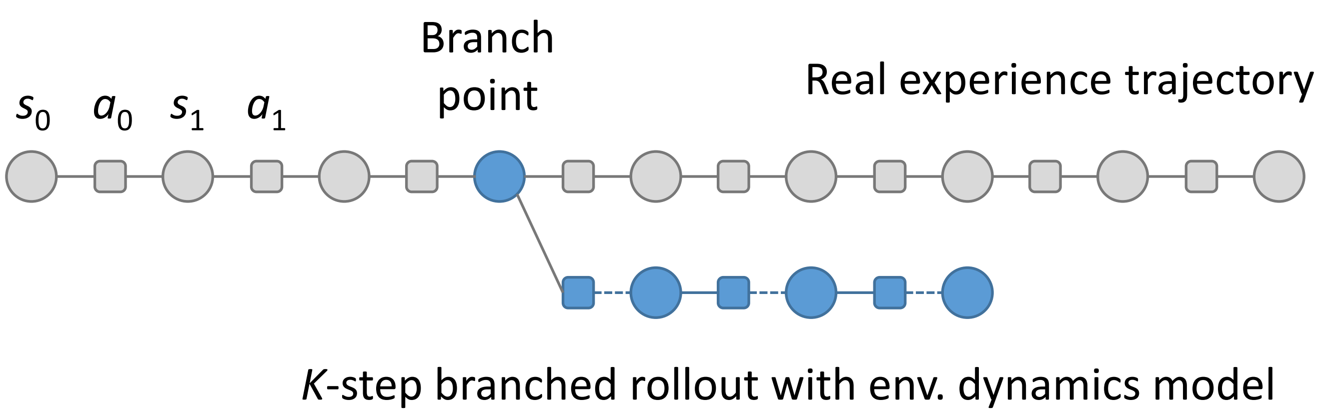
- Does the model really help improve the data efficiency?

- Inevitably, the model is to-some-extent inaccurate. When to trust the model?

- How to properly leverage the model to better train our policy?

Bound based on Model & Policy Error

- Branched rollout scheme
  - Begin a rollout from a state under the previous policy's state distribution $d_{\pi_D}(s)$ and run *k* steps according to π under the learned model p_θ
- Dyna can be viewed as a special case of *k* = 1 branched rollout



Janner, Michael, et al. "When to Trust Your Model: Model-Based Policy Optimization." NIPS 2019.

Bound based on Model & Policy Error

- Quantify model error and policy shift as

$$\epsilon_{m'} = \max_t \mathbb{E}_{s\sim\pi_t}[D_{TV}(p(s',r|s,a)\|p_\theta(s',r|s,a))]$$

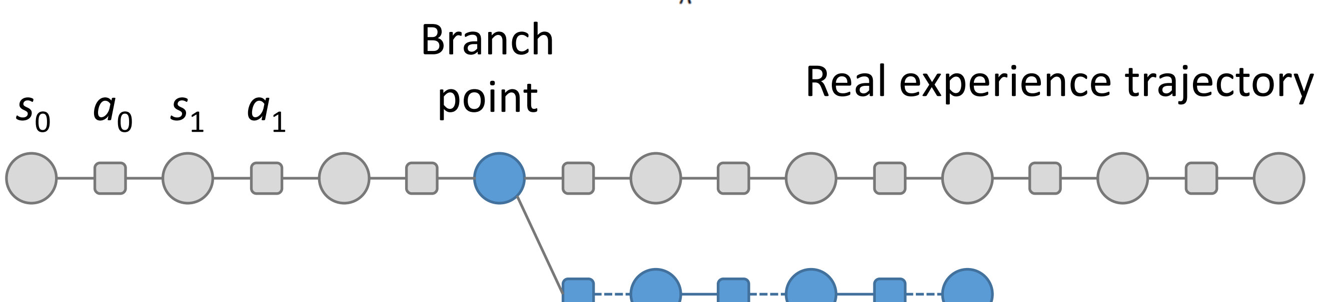
$$\epsilon_\pi = \max_s D_{TV}(\pi\|\pi_D)$$

- The policy value discrepancy bound is written as

$$\eta[\pi] \geq \eta^{\text{branch}}[\pi] - 2r_{\max}\left[\frac{\gamma^{k+1}\epsilon_\pi}{(1-\gamma)^2} + \frac{\gamma^k\epsilon_\pi}{(1-\gamma)} + \frac{k}{1-\gamma}(\epsilon_{m'})\right]$$

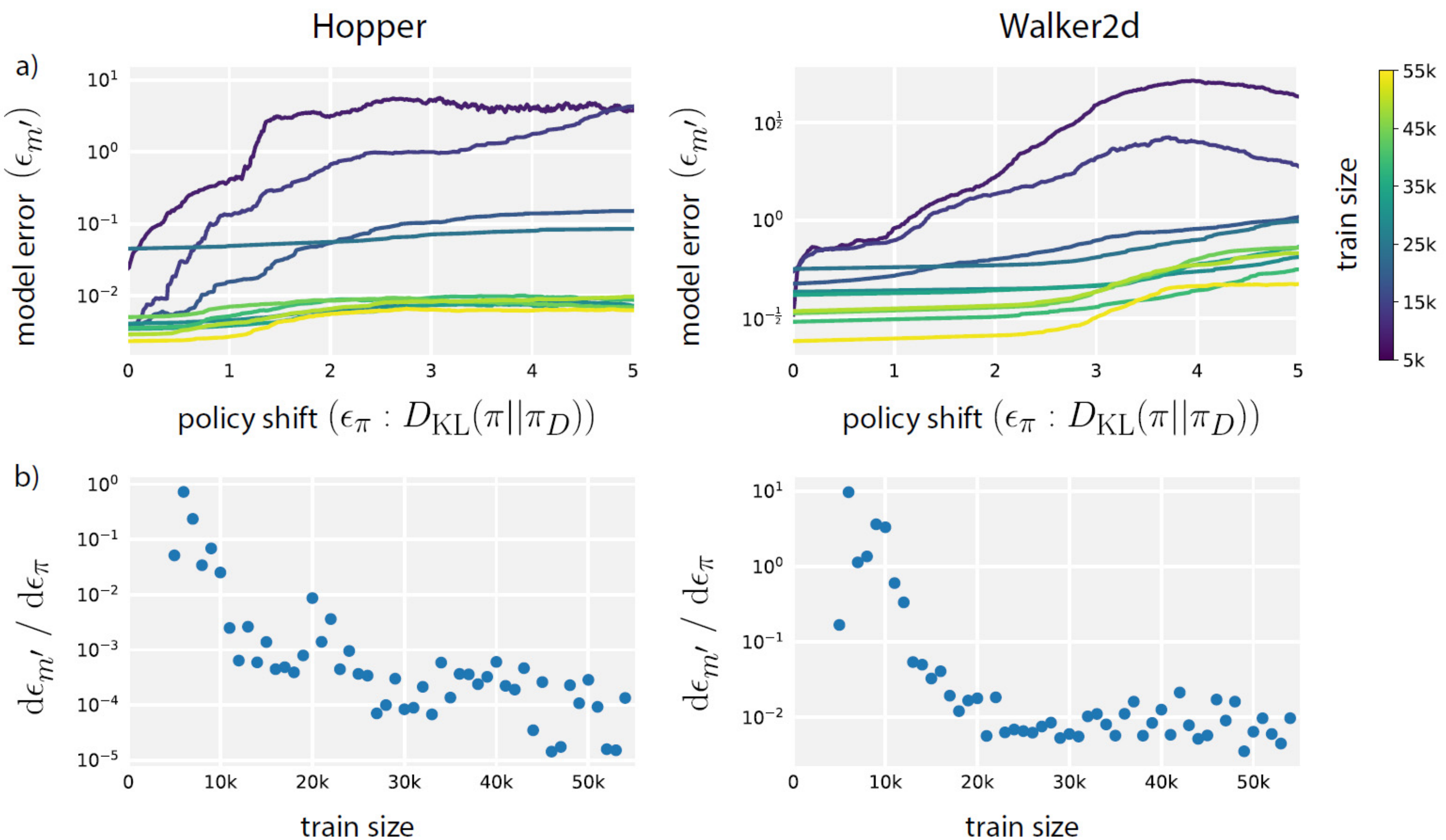
True value   Value in model

where the optimal $k > 0$ if $\frac{\mathrm{d}\epsilon_{m'}}{\mathrm{d}\epsilon_\pi}$ is sufficiently small



K-step branched rollout with env. dynamics model

Empirical Analysis of $\frac{d\epsilon_{m'}}{d\epsilon_\pi}$



$$\eta[\pi] \geq \eta^{\text{branch}}[\pi] - 2r_{\max}\left[\frac{\gamma^{k+1}\epsilon_\pi}{(1-\gamma)^2} + \frac{\gamma^k \epsilon_\pi}{(1-\gamma)} + \frac{k}{1-\gamma}(\epsilon_{m'})\right]$$

where the optimal k > 0 if $\frac{d\epsilon_{m'}}{d\epsilon_\pi}$ is sufficiently small

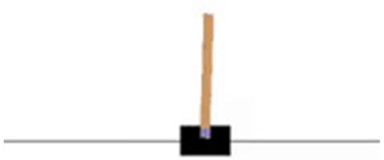
MBPO Algorithm

Algorithm 2 Model-Based Policy Optimization with Deep Reinforcement Learning

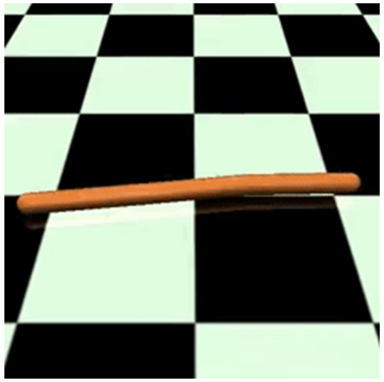
Initialize policy π_ϕ, predictive model p_θ, environment dataset $\mathcal{D}_{\text{env}}$, model dataset $\mathcal{D}_{\text{model}}$
for N epochs **do**
    Train model p_θ on $\mathcal{D}_{\text{env}}$ via maximum likelihood
    for E steps **do**
        Take action in environment according to π_ϕ; add to $\mathcal{D}_{\text{env}}$
        for M model rollouts **do**
            Sample s_t uniformly from $\mathcal{D}_{\text{env}}$
            Perform k-step model rollout starting from s_t using policy π_ϕ; add to $\mathcal{D}_{\text{model}}$
        for G gradient updates **do**
            Update policy parameters on model data: $\phi \leftarrow \phi - \lambda_\pi \hat{\nabla}_\phi J_\pi(\phi, \mathcal{D}_{\text{model}})$

- Remarks
    - Branch out from the real trajectories (instead from s_0)
    - Branch rollout k steps depends on model & policy
    - Soft AC to update policy

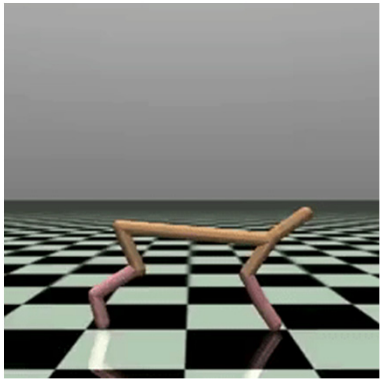
Experiment Environments



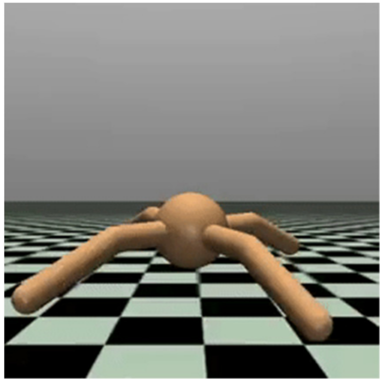
Cartpole



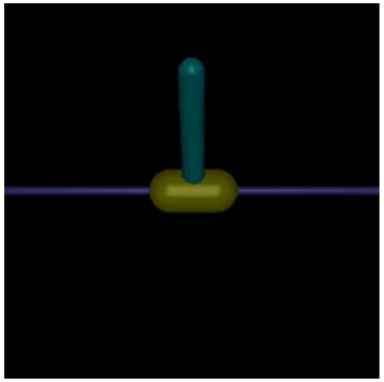
Swimmer



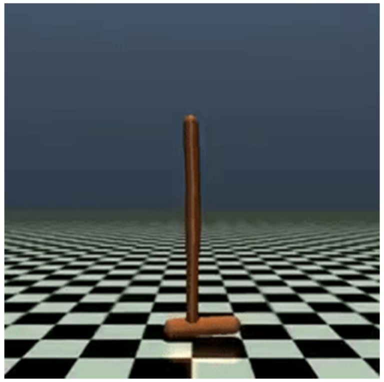
HalfCheetah



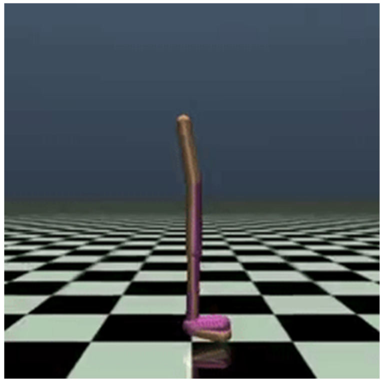
Ant



InvertedPendulum



Hopper

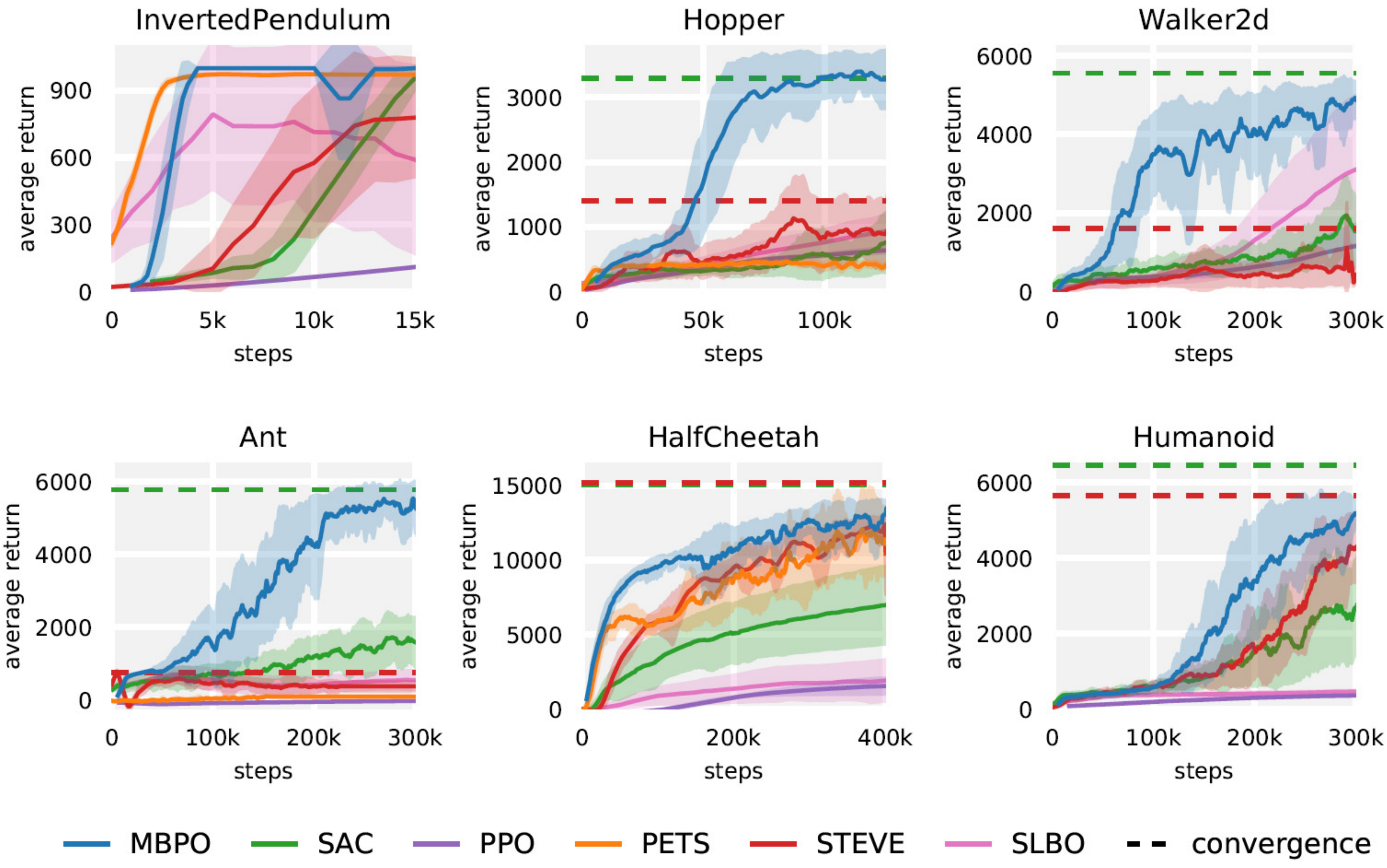


Walker2d

Humanoid

MBPO Experiments

1000-step horizon

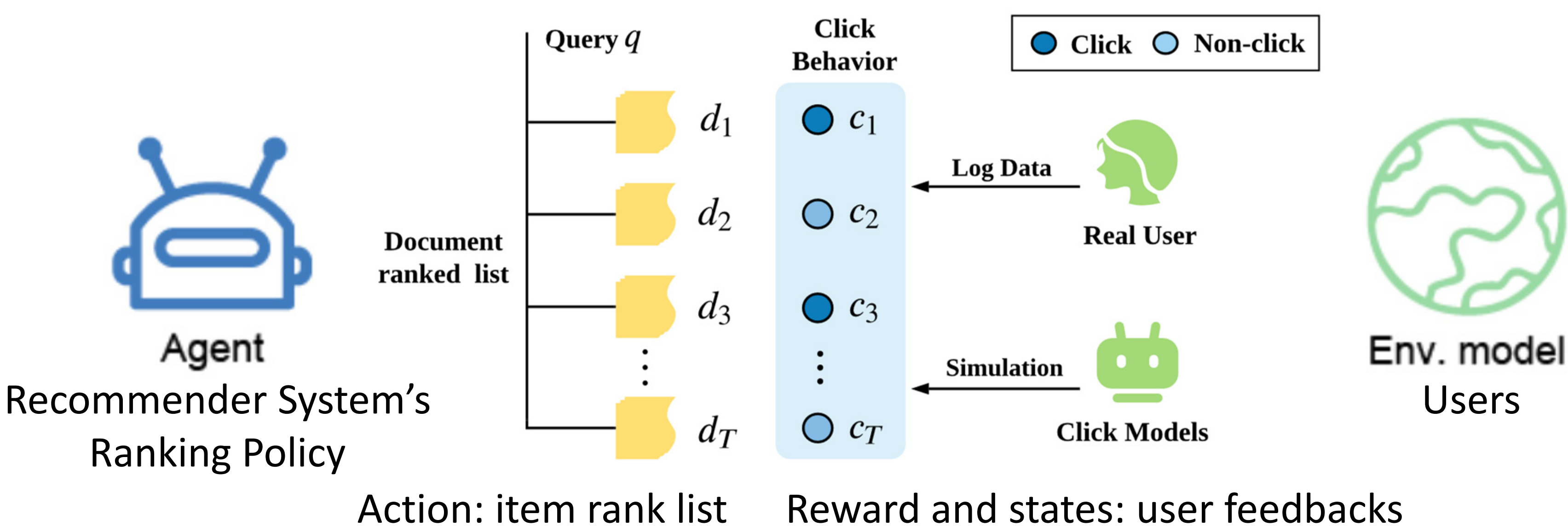


Content

1. A brief of model-based reinforcement learning
2. User click model and item ranking in recommendation
3. Future research on this direction

Click Model and Ranking Policy

- Click models (CMs) characterize how users interact with a ranked list of items.
- Given click logs, CMs are trained to predict a sequence of user clicks, and return a set of model parameters that reflect users' underlying behaviors.
- CMs provide useful evidence for ranking policies in both training and testing.



Overview of Click Models

- Probabilistic graphical model (PGM) based CMs
  - User behaviors are presented as a sequence of observable and hidden states.
  - Require manually designed dependencies.

- Neural network (NN) based CMs
  - User behaviors are encoded as vector representations.
  - Automatically model flexible dependencies.
  - Larger model capacity, leading to better performance.

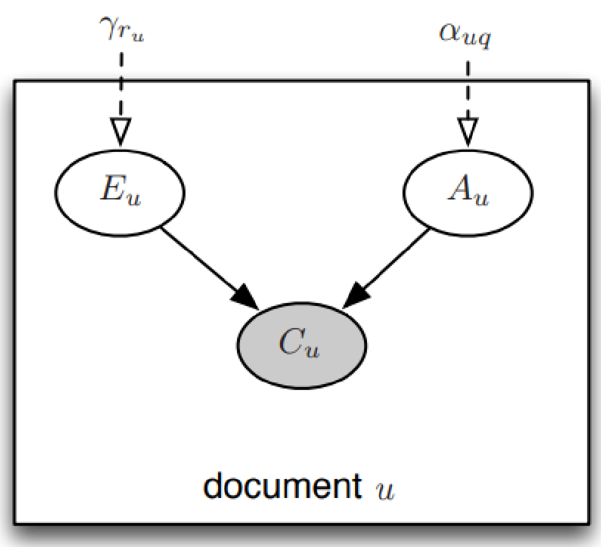
PBM: Position Based Model

- Examination hypothesis
  - A user clicks a document if and only if he/she examines the document and is attracted by the document.

$$C_u = 1 \Leftrightarrow E_u = 1 \text{ and } A_u = 1$$

  - Model parameters can be learned via MLE or EM.

$$P(C_u = 1) = P(E_u = 1) \cdot P(A_u = 1)$$
$$P(A_u = 1) = \alpha_{uq}$$
$$P(E_u = 1) = \gamma_{r_u}.$$



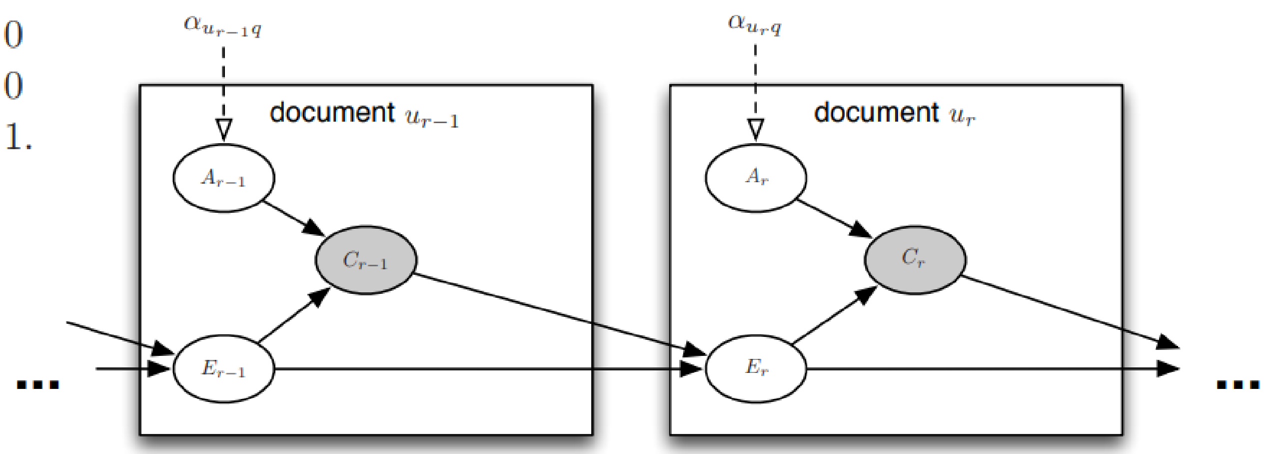
Graph dependencies of the position-based model (PBM)

Thorsten Joachims, Laura Granka, Bing Pan, Helene Hembrooke, and Geri Gay. Accurately interpreting clickthrough data as implicit feedback. In SIGIR, 2005.

CM: Cascade Model

- User browsing assumption
    - A user scans documents on a search page from top to bottom until he/she finds a relevant document.

$$
\begin{aligned}
C_r = 1 &\Leftrightarrow E_r = 1 \text{ and } A_r = 1 \\
P(A_r = 1) &= \alpha_{u_r q} \\
P(E_1 = 1) &= 1 \\
P(E_r = 1 \mid E_{r-1} = 0) &= 0 \\
P(E_r = 1 \mid C_{r-1} = 1) &= 0 \\
P(E_r = 1 \mid E_{r-1} = 1, C_{r-1} = 0) &= 1.
\end{aligned}
$$

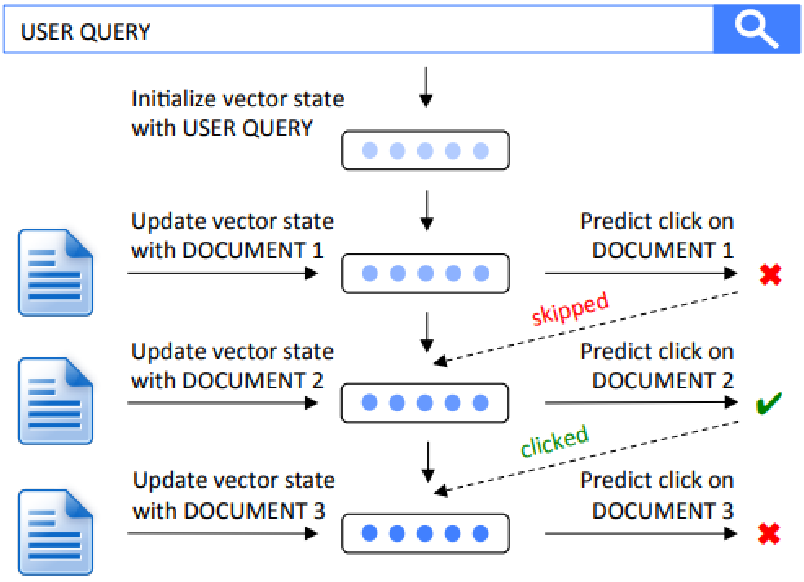


Graph dependencies of the cascade model (CM)

Nick Craswell, Onno Zoeter, Michael Taylor, and Bill Ramsey. 2008. An experimental comparison of click position-bias models. In Proceedings of the 2008 international conference on web search and data mining. 87–94

NCM: Neural Click Model

- The first work to introduce neural networks click models.
  - Vector representations for user behaviors or query/document features.
  - Apply RNN/LSTM to encode sequential information within a document list of the query.



Alexey Borisov, Ilya Markov, Maarten De Rijke, and Pavel Serdyukov. 2016. A neural click model for web search. In Proceedings of the 25th International Conference on World Wide Web. 531–541

AICM: Adversarial Imitation Click Model

- Motivations
  - Click models learn a click behavior policy from log data, which is an imitation of real users.
  - Existing click models suffer from exposure bias.
    - **During training,** predict next click based on the 'right' clicks
    - **During testing,** predict next click based on previous predictions
    - dynamic nature of user behavior **V.S.** static user modeling

Xinyi Dai, Jianghao Lin, Weinan Zhang, Shuai Li, Weiwen Liu, Ruiming Tang, Xiuqiang He, Jianye Hao, Jun Wang, Yong Yu. An Adversarial Imitation Click Model for Information Retrieval. In WWW 2021.

AICM: Adversarial Imitation Click Model

- Dynamic modeling
  - Base user's current state on previous predictions
  - Optimize a long term objective instead of a short-sighted one-step loss
  - Alleviates the exposure bias
- Adversarial training
  - Minimize JS divergence instead of KL divergence
  - Generalize well on different ranked list distributions
- Modeling users' intrinsic utility explicitly
  - Use a reward function to guide the learning of a click policy that reproduce users' behavior
  - Provide important insights and useful guidance for ranking

Xinyi Dai, Jianghao Lin, Weinan Zhang, Shuai Li, Weiwen Liu, Ruiming Tang, Xiuqiang He, Jianye Hao, Jun Wang, Yong Yu.
An Adversarial Imitation Click Model for Information Retrieval. In WWW 2021.

AICM: Adversarial Imitation Click Model

Generator

$$\mathbf{x}_t = \begin{cases} \mathbf{v}_q \oplus \mathbf{0}_d \oplus \mathbf{0}_v \oplus \mathbf{0}_c & t = 0 \\ \mathbf{v}_q \oplus \mathbf{v}_d \oplus \mathbf{v}_v \oplus \mathbf{0}_c & t = 1 \\ \mathbf{v}_q \oplus \mathbf{v}_d \oplus \mathbf{v}_v \oplus \mathbf{v}_c & t = 2, \ldots, T \end{cases}$$

$$\mathbf{h}_t = \mathrm{GRU}_g\left(\mathbf{h}_{t-1}, \mathbf{x}_t\right),$$

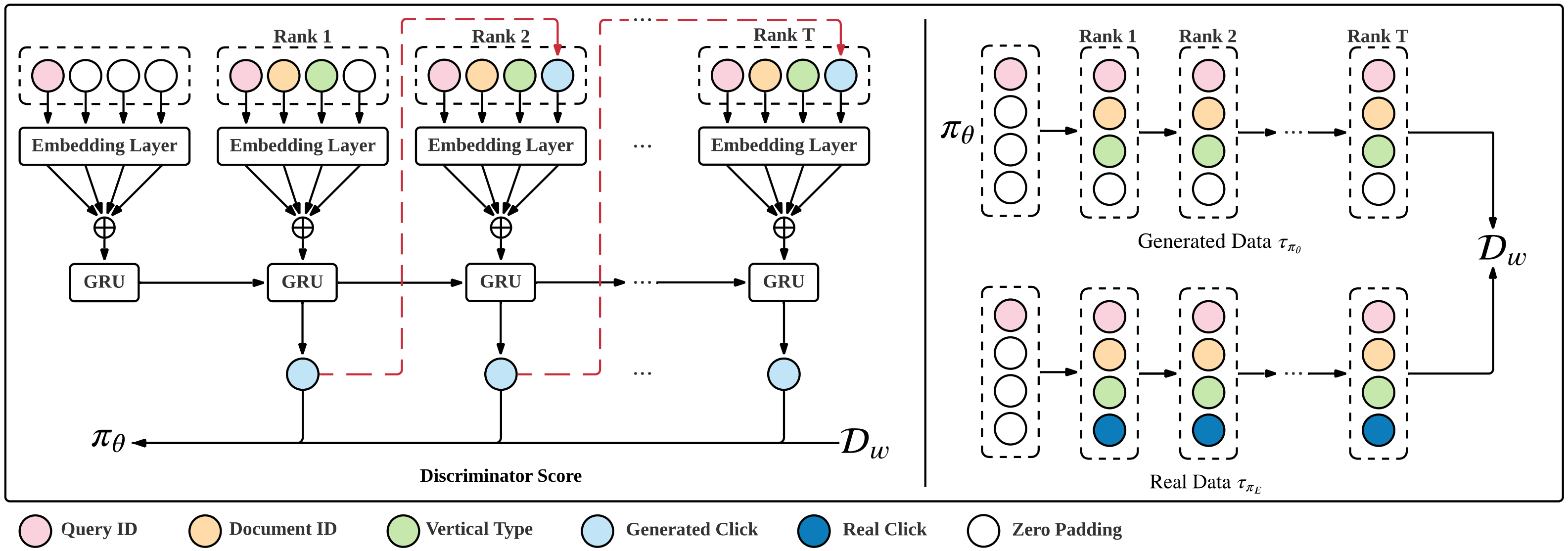
$$\pi_\theta\left(a_t | s_t\right) = \mathrm{Softmax}\left(\mathrm{Linear}(\mathbf{h}_t)\right),$$

Discriminator

$$\mathbf{x}'_t = \begin{cases} \mathbf{v}_q \oplus \mathbf{0}_d \oplus \mathbf{0}_v \oplus \mathbf{0}_c & t = 0 \\ \mathbf{v}_q \oplus \mathbf{v}_d \oplus \mathbf{v}_v \oplus \mathbf{v}_c & t = 1, \ldots, T \end{cases}$$

$$\mathbf{h}'_t = \mathrm{GRU}_d\left(\mathbf{h}'_{t-1}, x'_t\right),$$

$$D_w\left(s_t, a_t\right) = \mathrm{Sigmoid}\left(\mathrm{Linear}\left(\mathbf{h}'_t\right)\right).$$



Xinyi Dai, Jianghao Lin, Weinan Zhang, Shuai Li, Weiwen Liu, Ruiming Tang, Xiuqiang He, Jianye Hao, Jun Wang, Yong Yu.
An Adversarial Imitation Click Model for Information Retrieval. In WWW 2021.

AICM Experiments

- Performance on traditional metrics

Group	Model	Click Prediction		Relevance Estimation			
		LL	PPL	NDCG@1	NDCG@3	NDCG@5	NDCG@10
PGM based	CCM	-0.2224	1.2034	0.6702	0.6941	0.7229	0.8477
	DCM	-0.2302	1.1994	0.6807	0.6824	0.7161	0.8452
	DBN	-0.2218	1.2103	0.6711	0.6958	0.7241	0.8471
	SDBN	-0.2328	1.2116	0.6868	0.6846	0.7177	0.8455
	PBM	-0.1483	1.1894	0.6481	0.6419	0.6726	0.8235
	UBM	-0.1494	1.1896	0.6435	0.6381	0.6681	0.8223
NN based	NCM	-0.1443	1.1855	0.7003	0.7041	0.7351	0.8608
	CACM	-0.1426	1.1832	0.7347	0.7153	0.7403	0.8662
	AICM	**-0.1385****	**1.1747****	**0.7348**	**0.7167***	**0.7439***	**0.8667***

$$LL = \frac{1}{MN}\sum_{i=1}^{N}\sum_{t=1}^{M} C_{i,t}\log \mathcal{P}_{i,t} + (1 - C_{i,t})\log(1 - \mathcal{P}_{i,t}),$$

$$PPL@t = 2^{-\frac{1}{N}\sum_{i=1}^{N} C_{i,t}\log \mathcal{P}_{i,t} + (1-C_{i,t})\log(1-\mathcal{P}_{i,t})},$$

Xinyi Dai, Jianghao Lin, Weinan Zhang, Shuai Li, Weiwen Liu, Ruiming Tang, Xiuqiang He, Jianye Hao, Jun Wang, Yong Yu.
An Adversarial Imitation Click Model for Information Retrieval. In WWW 2021.

AICM Experiments

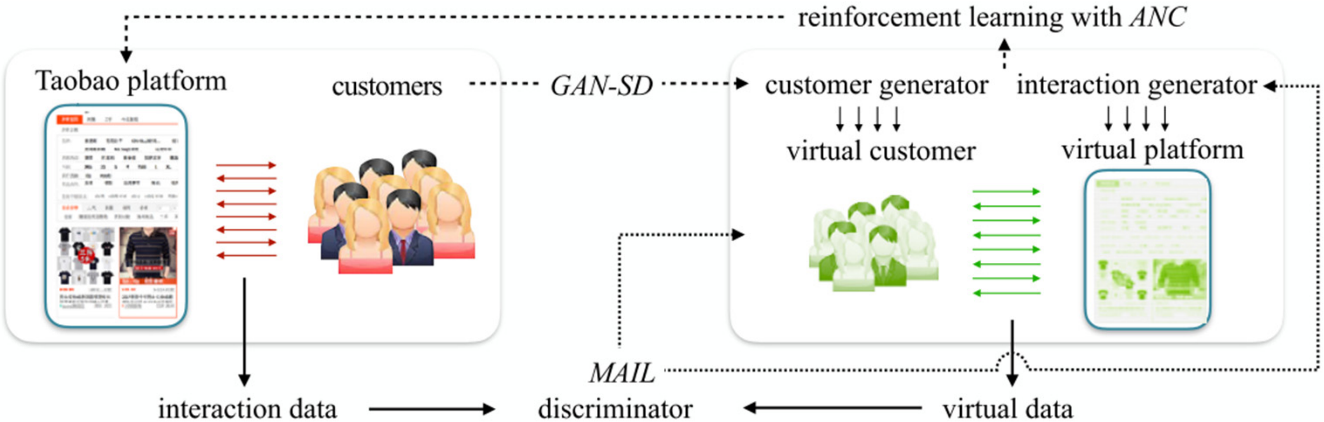
- Distributional Coverage
  - **Forward PPL:** the PPL of a surrogate model that is trained on held-out real data and evaluated on generated samples.
  - **Reverse PPL:** the PPL of a surrogate model that is trained on generated samples and evaluated on held-out real data.

	Surrogate UBM		Surrogate NCM	
Data	Reverse PPL	Forward PPL	Reverse PPL	Forward PPL
Real data	1.1412	1.1412	1.1453	1.1453
UBM samples	1.4249	3.3833	1.4231	2.9435
NCM samples	1.1831	1.2072	1.1848	1.2021
CACM samples	1.1854	1.2615	1.1812	1.2565
AICM samples	**1.1747**	**1.1383**	**1.1745**	**1.1324**

Xinyi Dai, Jianghao Lin, Weinan Zhang, Shuai Li, Weiwen Liu, Ruiming Tang, Xiuqiang He, Jianye Hao, Jun Wang, Yong Yu.
An Adversarial Imitation Click Model for Information Retrieval. In WWW 2021.

User Models for Ranking Policy Training

- Virtual-Taobao
  - Build a simulator using GAN-SD and MAIL, then train policies on this simulator rather than real environment.

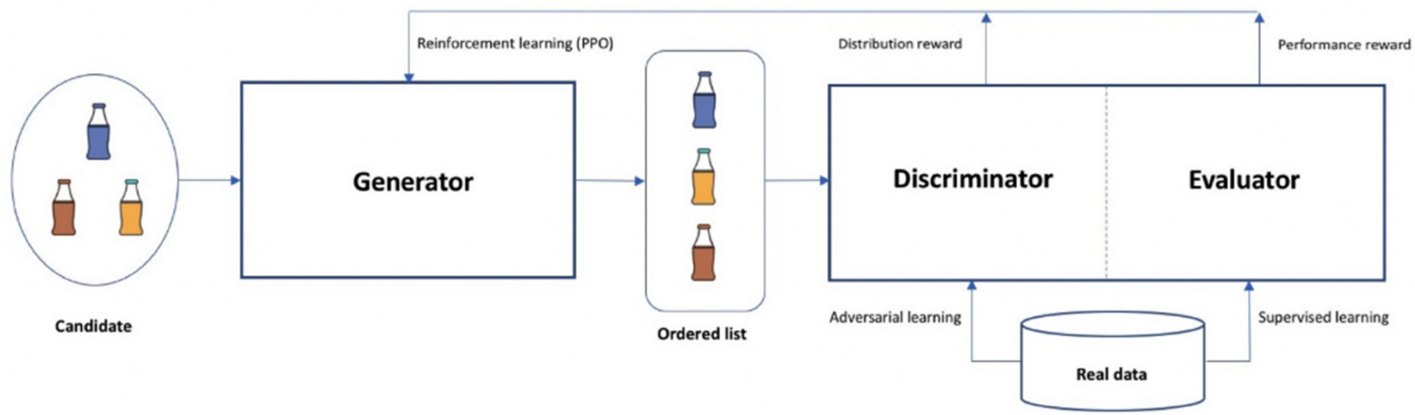


  - GAN-SD: Simulate customers including their request.
  - MAIL: Generate interactions by distinguish the simulated interactions from the real interactions.
  - ANC: Avoid algorithm over fit to virtual environment.

Shi J C, Yu Y, Da Q, et al. Virtual-taobao: Virtualizing real-world online retail environment for reinforcement learning[C] Proceedings of the AAAI Conference on Artificial Intelligence. 2019, 33(01): 4902-4909. https://arxiv.org/abs/1805.10000

User Models for Ranking Policy Training

- EG-Rerank+
  - Address offline-online inconsistency problem, avoiding the pitfalls of online interaction-based evaluation.

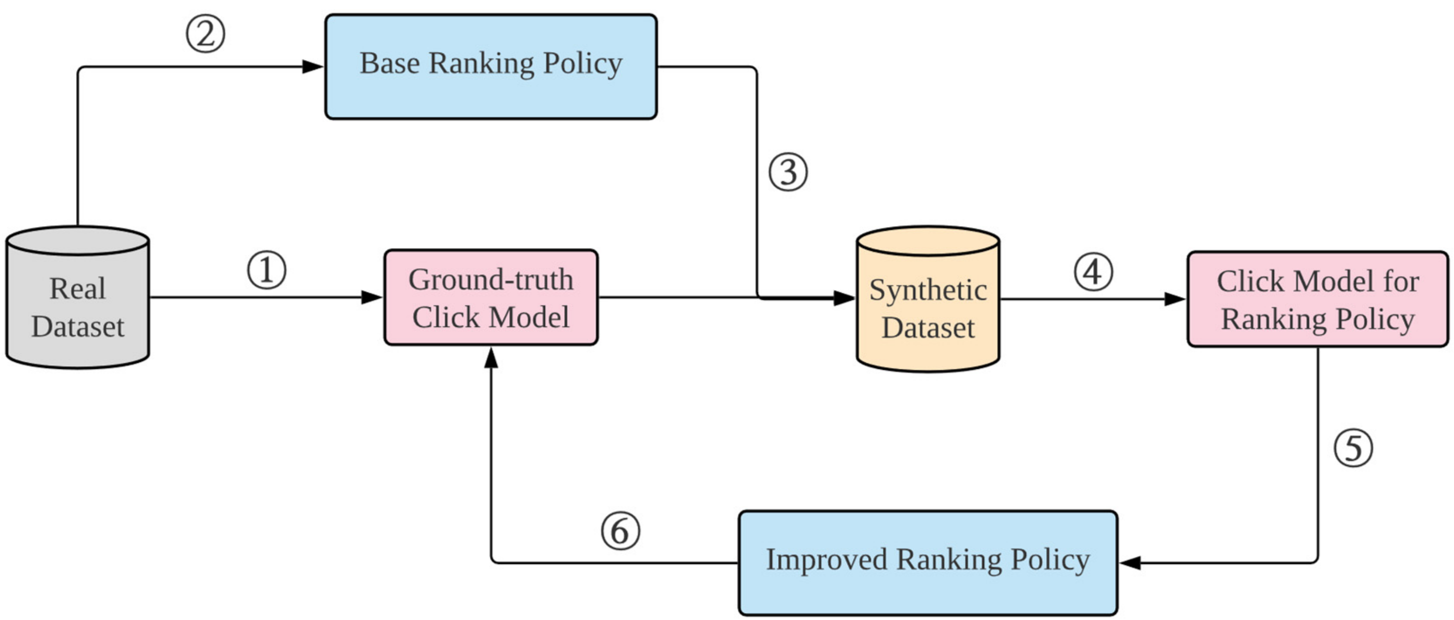


  - Generator: Produces orders with high scores using RL.
  - Evaluator: Predicts the performance of given lists.
  - Discriminator: Measures how much evaluator results can be trusted (adversarial learning).

Huzhang G, Pang Z J, Gao Y, et al. AliExpress Learning-To-Rank: Maximizing Online Model Performance without Going Online[J]. arXiv preprint arXiv:2003.11941, 2020. https://arxiv.org/abs/2003.11941v5

Click Models for Ranking Policy Training

- Use well trained click models (UBM, AICM, etc.) for training and evaluating ranking policy.
- Overall Offline Experiment Procedure:



Content

1. A brief of model-based reinforcement learning
2. User click model and item ranking in recommendation
3. Future research on this direction

Future Research

- Key questions to answer
  - As the user click model is always inaccurate, to-what-extent can it improve sample efficiency of the training of ranking policy?
  - How large should be the training data when it does not need model-based RL?
  - How to properly leverage the click model to improve the performance of ranking policy?
  - Can learning to rank, as a solver of ranking policy, yield higher sample efficiency than reinforcement learning when training using click model?

Thank You!
Questions?

Weinan Zhang

Shanghai Jiao Tong University

http://wnzhang.net