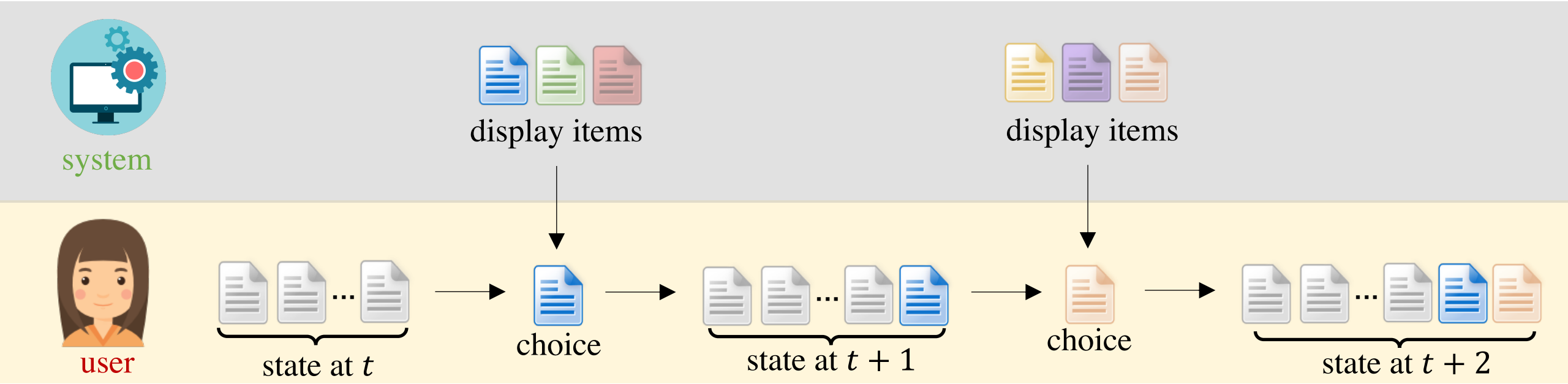
Generative Adversarial User Model for Reinforcement Learning Based Recommendation System

Xinshi Chen[1], Shuang Li[1], Hui Li[2], Shaohua Jiang[2], Yuan Qi[2], Le Song[1,2]

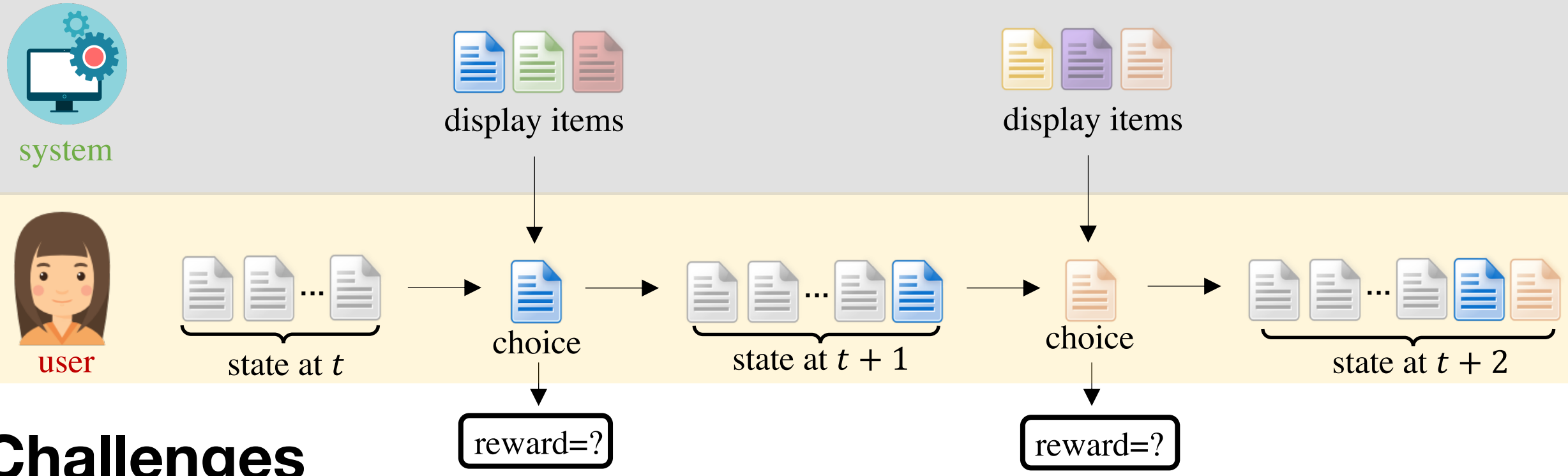
[1]Georgia Tech, [2]Ant Financial

ICML 2019

RL for Recommendation System



- A user's interest evolves over time based on what she observes.
- Recommender's action can significantly influence such evolution.
- A RL based recommender can consider user's long term interest.



Challenges

1. User is the **environment** → Training of **RL** policy requires lots of interactions with users

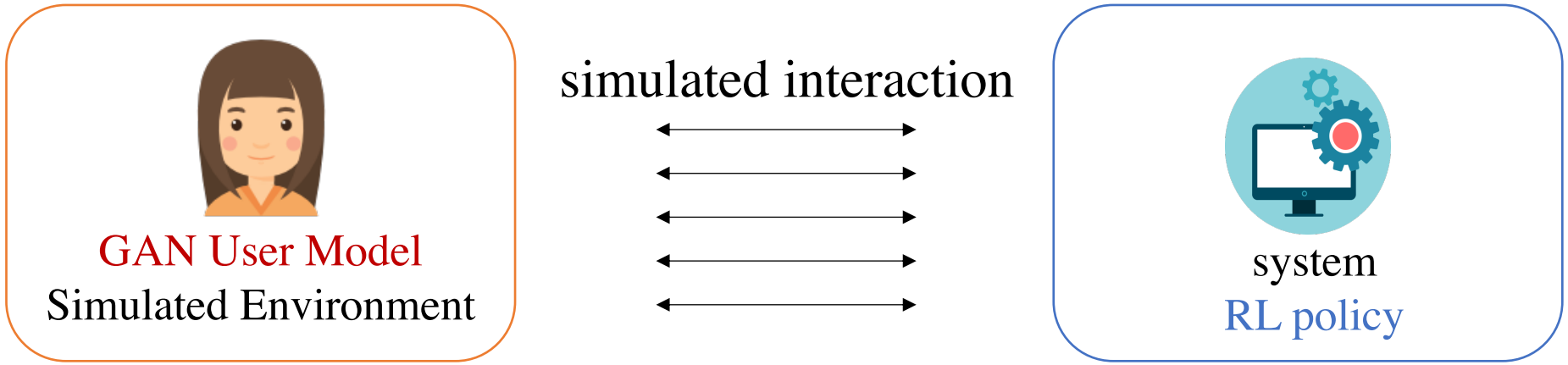
e.g.
(1) For **AlphaGo Zero**, **4.9 million** games of self-play were generated for training.
(2) RL for **Atari** game takes more than **50 hours** on GPU for training.

2. The **reward** function (a user's interest) is unknown

Our solution

We propose

- A **G**enerative **A**dversarial **User Model**
  - to model user's ***action***
  - to recover user's ***reward***
- Use GAN User Model as a ***simulator*** to pre-train the **RL** policy offline



Generative Adversarial User Model

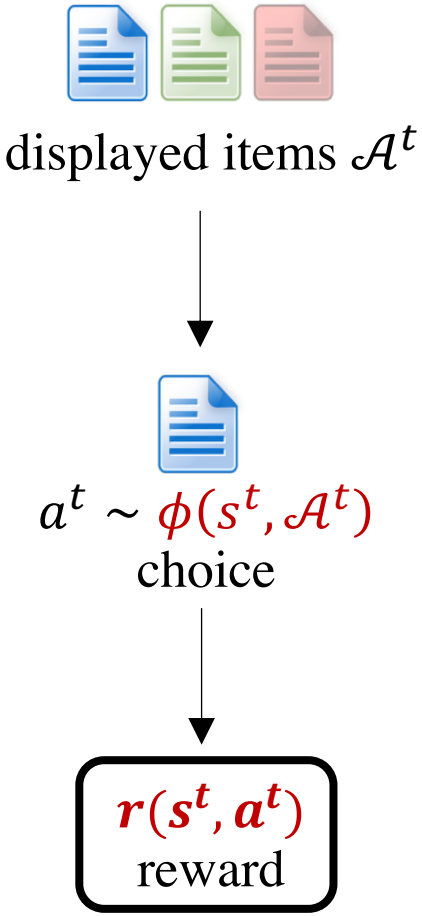
2 components:

User's ***reward*** $\boldsymbol{r(s^t, a^t)}$

- a^t is clicked item.
- s^t is user's experience (state).

User's ***behavior*** $\boldsymbol{\phi(s^t, \mathcal{A}^t)}$

- $\mathcal{A}^t$ contains items displayed by the system.
- act $a^t \sim \phi$ to maximize her expected reward.
- $\phi^*(s^t, \mathcal{A}^t) = \arg\max_{\phi} \mathbb{E}_{\phi}[r(s^t, a^t)] - R(\phi)/\eta$



Generative Adversarial Training

In analogy to GAN:

- $\boldsymbol{\phi}$ (behavior) acts as a generator
- $\boldsymbol{r}$ (reward) acts as a discriminator

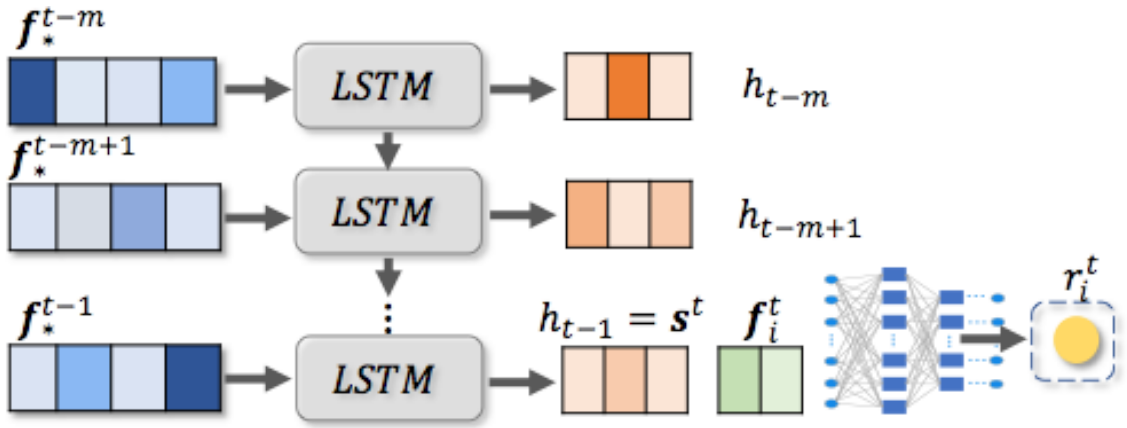
Jointly learned via a ***mini-max formulation***:

$$\min_{r} \max_{\phi} \mathbb{E}_{\phi} \left[\sum_{t=1}^{T} r(s^t, a^t) \right] - R(\phi)/\eta - \sum_{t=1}^{T} r(s^t_{true}, a^t_{true})$$

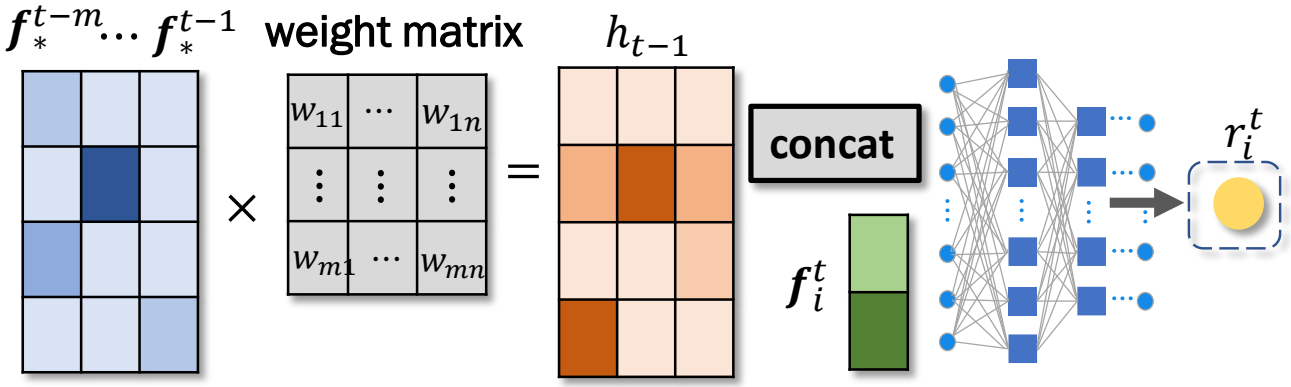
Model Parameterization

2 architectures for aggregating historical information (i.e. state s^t)

(1) LSTM



(2) Position Weight



Set Recommendation RL policy

all available K items

set recommendation

a_1^* a_2^* a_k^*

display k items

$$a_1^*, a_2^*, \ldots a_k^* = \arg\max_{a_1,\ldots,a_k} Q(s^t, a_1, a_2, \ldots, a_k)$$

combinatorial action space $\binom{K}{k}$ → Intractable computation!

Set Recommendation RL policy

We design a **cascading Q** network to compute the optimal action with ***linear*** complexity:

$$a_1^*, a_2^*, \dots a_k^* = \arg\max_{a_1,\dots,a_k} Q(s^t, a_1, a_2, \dots, a_k)$$

decompose

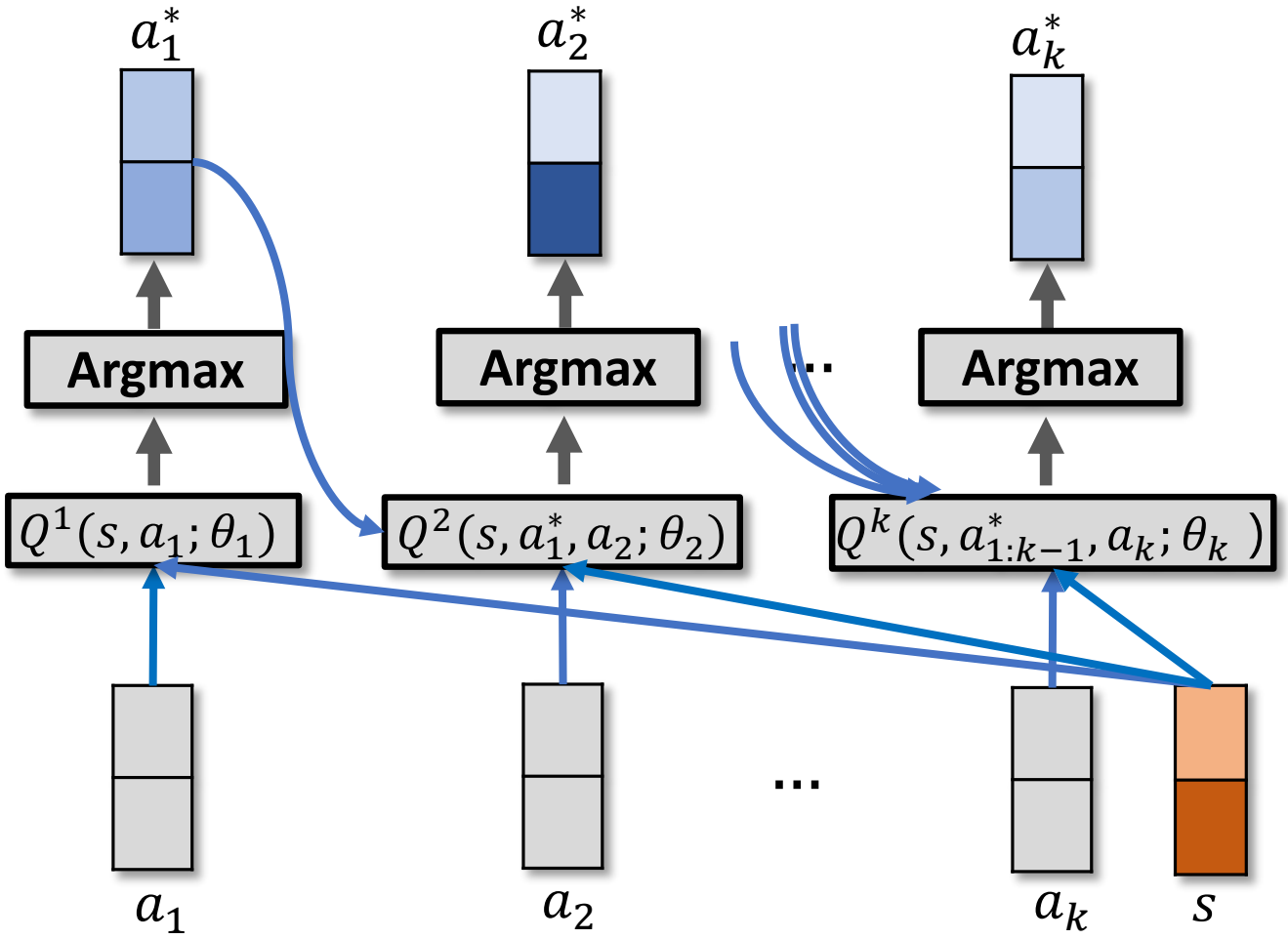
$$a_1^* = \arg\max_{a_1} \boldsymbol{Q^{1*}}(s^t, a_1) \qquad \boldsymbol{Q^{1*}}(s^t, a_1) := \max_{a_{2:k}} Q(s^t, a_1, a_{2:k})$$

$$a_2^* = \arg\max_{a_2} \boldsymbol{Q^{2*}}(s^t, a_1^*, a_2) \qquad \boldsymbol{Q^{2*}}(s^t, a_1, a_2) := \max_{a_{3:k}} Q(s^t, a_1, a_2, a_{3:k})$$

... ...

$$a_k^* = \arg\max_{a_k} \boldsymbol{Q^{k*}}(s^t, a_1^*, a_2^*, \dots, a_k)$$

Set Recommendation RL policy: Cascading DQN



Experiments

Predictive Performance of User Model

	(1) MovieLens		(2) LastFM		(6) Ant Financial	
Model	prec(%)@1	prec(%)@2	prec(%)@1	prec(%)@2	prec(%)@1	prec(%)@2
IKNN	38.8(±1.9)	40.3(±1.9)	20.4(±0.6)	32.5(±1.4)	20.6(±0.2)	32.1(±0.2)
S-RNN	39.3(±2.7)	42.9(±3.6)	9.4(±1.6)	17.4(±0.9)	32.2(±0.9)	40.3(±0.6)
SCKNNC	49.4(±1.9)	51.8(±2.3)	21.4(±0.5)	26.1(±1.0)	34.6(±0.7)	43.2(±0.8)
XGBOOST	66.7(±1.1)	76.0(±0.9)	10.2(±2.6)	19.2(±3.1)	41.9(±0.1)	65.4(±0.2)
DFM	63.3(±0.4)	75.9(±0.3)	10.5(±0.4)	20.4(±0.1)	41.7(±0.1)	64.2(±0.2)
W&D-LR	61.5(±0.7)	73.8(±1.2)	7.6(±2.9)	16.6(±3.3)	37.5(±0.2)	60.9(±0.1)
W&D-CCF	65.7(±0.8)	75.2(±1.1)	15.4(±2.4)	25.7(±2.6)	37.7(±0.1)	61.1(±0.1)
GAN-PW	66.6(±0.7)	75.4(±1.3)	**24.1**(±0.8)	**34.9**(±0.7)	41.9(±0.1)	65.8(±0.1)
GAN-LSTM	**67.4**(±0.5)	**76.3**(±1.2)	24.0(±0.9)	34.9(±0.8)	**42.1**(±0.2)	**65.9**(±0.2)

Recommendation Policy Based On User Model

	$k = 3$		$k = 5$	
model	reward	ctr	reward	ctr
W&D-LR	14.46(±0.42)	0.46(±0.01)	15.18(±0.38)	0.48(±0.01)
W&D-CCF	19.93(±1.09)	0.62(±0.03)	20.94(±1.03)	0.65(±0.03)
GAN-Greedy	21.37(±1.24)	0.67(±0.04)	22.97(±1.22)	0.71(±0.03)
GAN-RWD1	22.17(±1.07)	0.68(±0.03)	25.15(±1.04)	**0.78**(±0.03)
GAN-GDQN	23.60(±1.06)	0.72(±0.03)	23.19(±1.17)	0.70(±0.03)
GAN-CDQN	**24.05**(±0.98)	**0.74**(±0.03)	**25.36**(±1.10)	0.77(±0.03)
DQN-Off	20.31(±0.14)	0.63(±0.01)	21.82(±0.08)	0.67(±0.01)

Experiments

Cascading-DQN policy ***pre-trained*** over a **GAN User Model** can quickly achieve a high CTR even when it is applied to a new set of users.

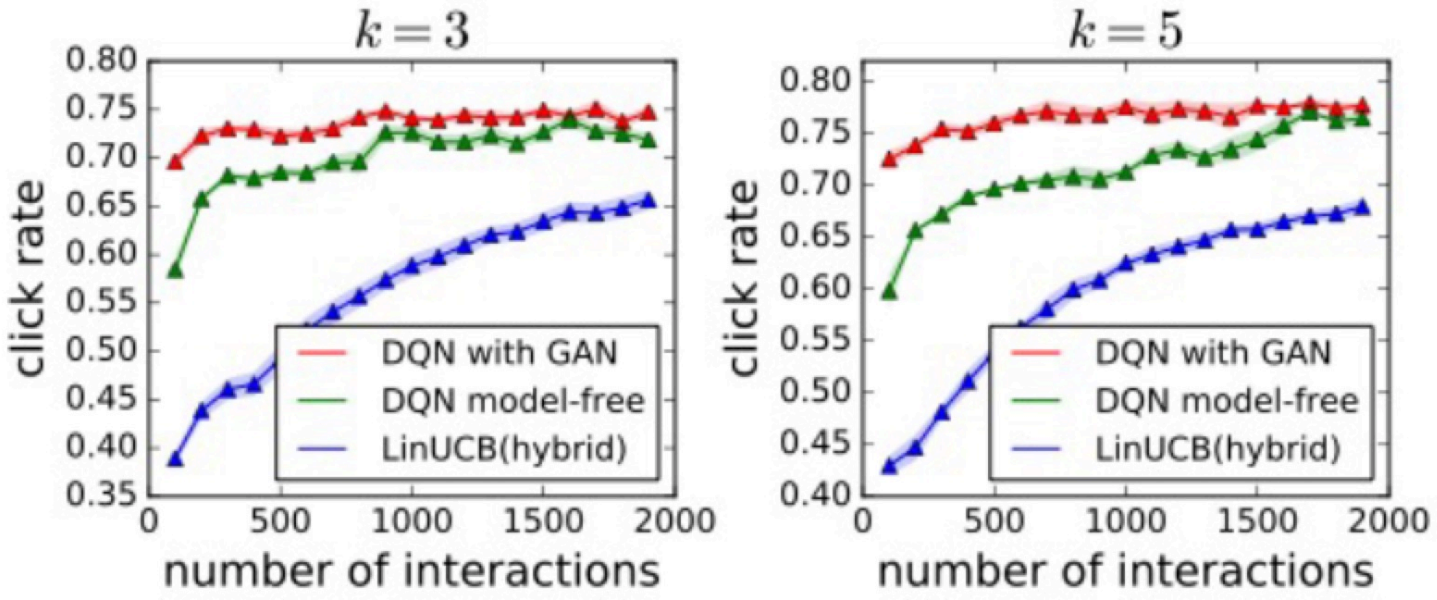


Figure: Comparison of the averaged click rate averaged over 1,000 users under different recommendation policies. X-axis represents how many times the recommender interacts with online users. Y-axis is the click rate. Each point (x, y) means the click rate y is achieved after x times of user interactions.

Thanks!

Poster: Pacific Ballroom #252, Tue, 06:30 PM

Contact: xinshi.chen@gatech.edu