Modern Reinforcement Learning Algorithms in Operations Management

Yao Wang

Center for Intelligent Decision-making and Machine Learning

Xi'an Jiaotong University

Aug. 23, 2025

Outline

- **Introduction**
- Generalized Tensor Contextual Bandits
- Deep RL for Online Assortment Customization
- Distributed Q-Learning for DTRs

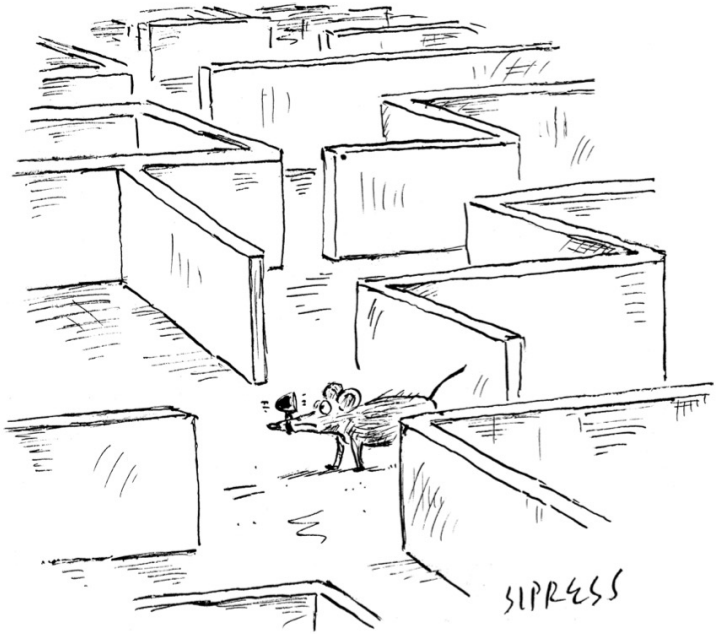
Recap: Supervised Learning

Given i.i.d training data, the goal is to make prediction on unseen data:

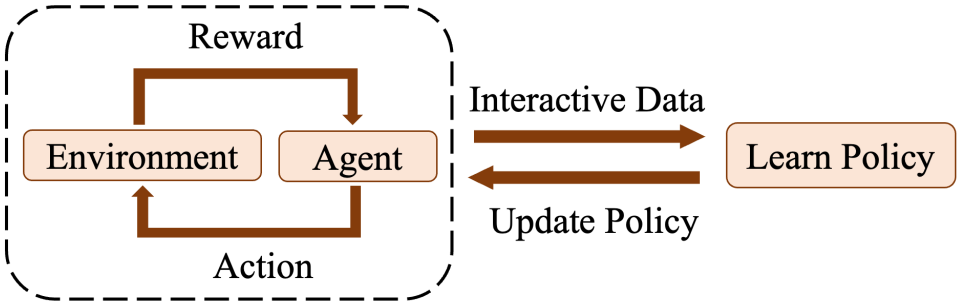
Reinforcement Learning (RL)

In RL, an agent learns by interacting with an environment.

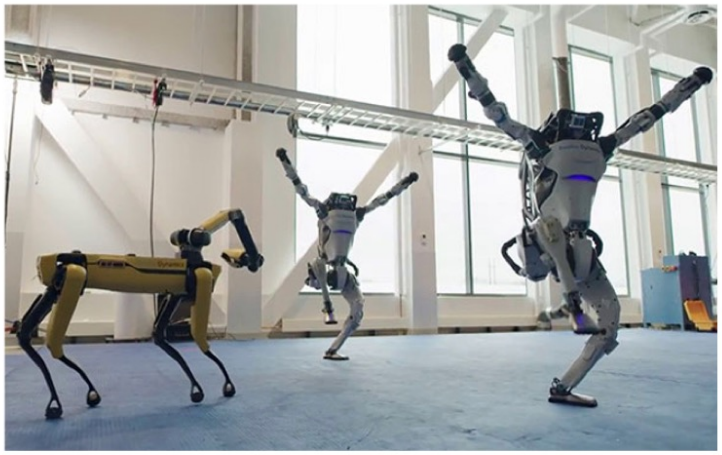
- no training data
- maximize total rewards
- trial-and-error
- sequential and online



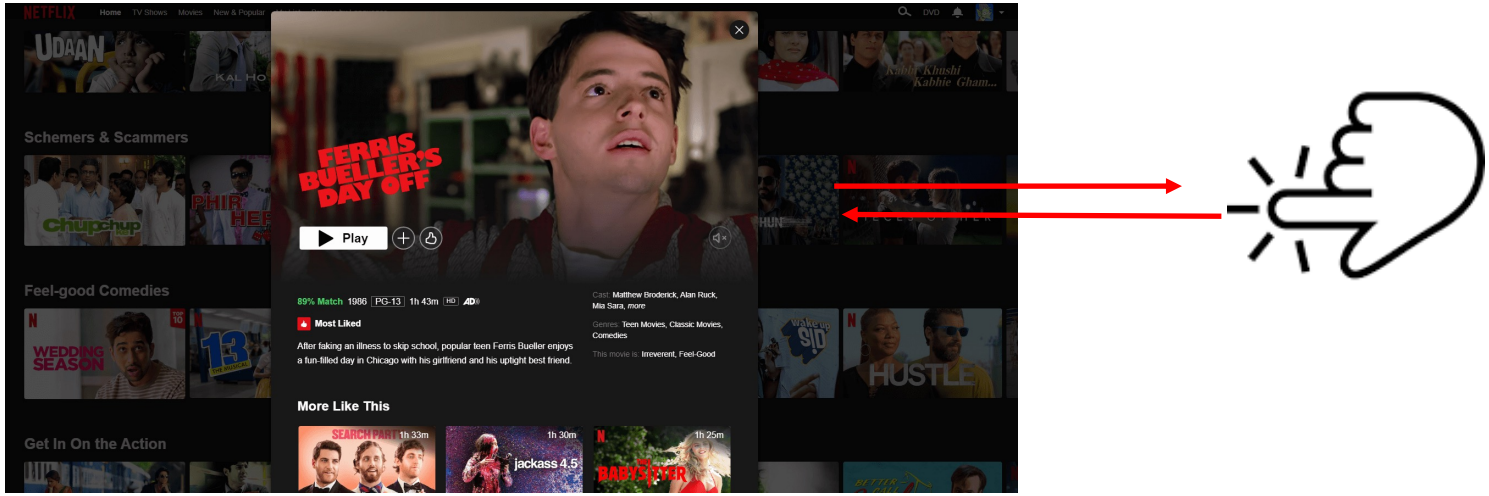
"Recalculating ... recalculating ..."



Successes of RL



Motivating Example: Movie Recommendation

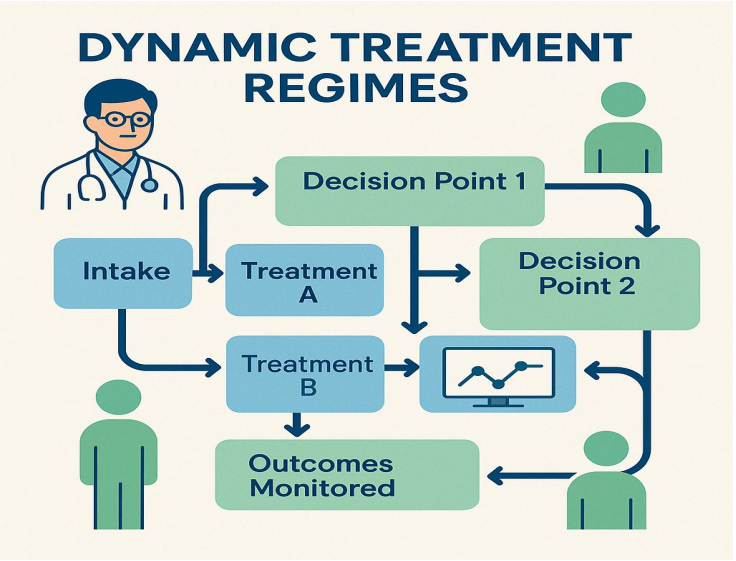


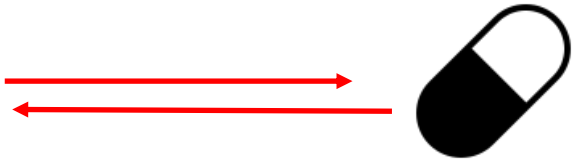
- Goal: Recommending movies in real-time based on user preferences and behavior
- Sequential decision making: Refine recommendations by learning from dynamic user interactions and feedback

Motivating Example: Assortment Optimization

- Goal: Optimizing product assortments to maximize revenue and customer satisfaction
- Sequential decision making: Dynamically adjust assortments based on customer choices and demand trends

Motivating Example: Precision Medicine





- Goal: Tailor medical treatments based on individual patient data and health history
- Sequential decision making: Refine treatment plans by learning from evolving patient responses and feedback

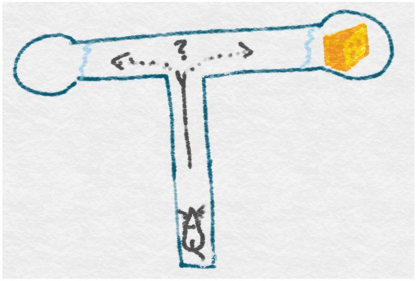
Outline

- Introduction
- **Generalized Tensor Contextual Bandits**
- Deep RL for Online Assortment Customization
- Distributed Q-Learning for DTRs

Yi et al., IEEE TKDE, 2024

Bandits

- A classical method of online reinforcement learning
- What's in a name? A tiny bit of history
  - First bandit algorithm proposed by Thompson (1933)
  - Bush and Mosteller (1953) were interested in how mice behaved in a T-maze



  - "Bandit'" because they steal your money

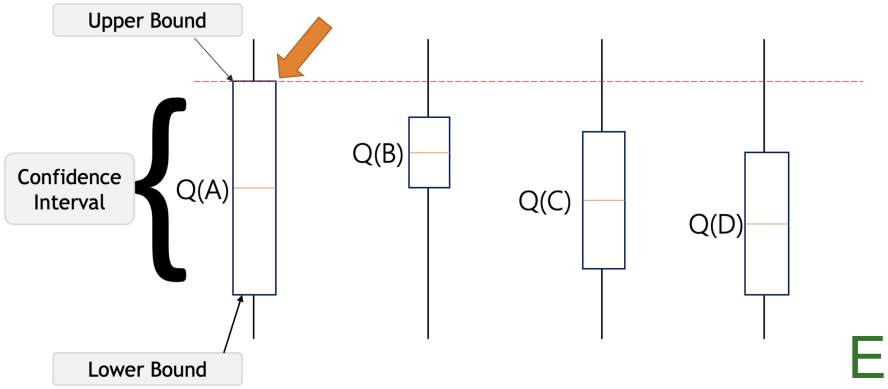
Round	1	2	3	4	5	6	7	8	9	10
LEFT	0		10	0		0				10
RIGHT		10			0		0	0	0	

What is your strategy? (Exploitation-Exploration dilemma)

The Upper Confidence Bound (UCB) Algorithm

Optimism in Face of Uncertainty

"Whenever the value of an action is uncertain, consider its largest plausible value, and then select the best action."



Upper **C**onfidence **B**ound (**UCB**)

$$x_t = \underset{x}{\operatorname{argmax}}\, Q_t(x) + C_t(x)$$

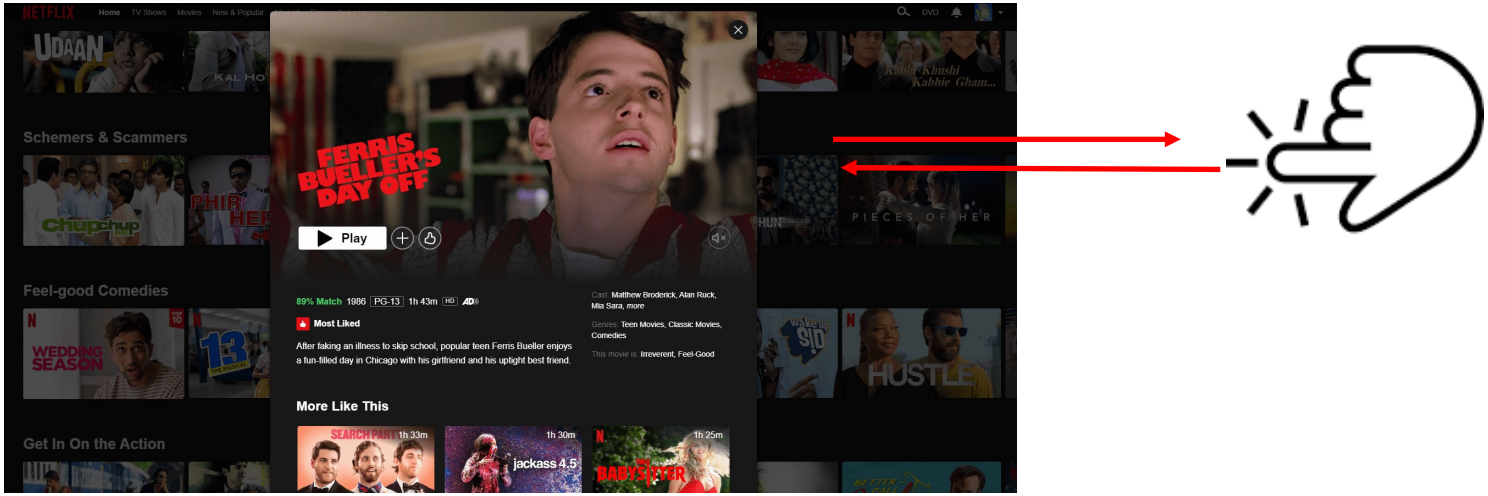
Estimated reward (Exploitation)

Confidence interval (Exploration)

UCB can efficiently solve the Exploitation-Exploration dilemma!

Auer et al., Machine Learning, 2002

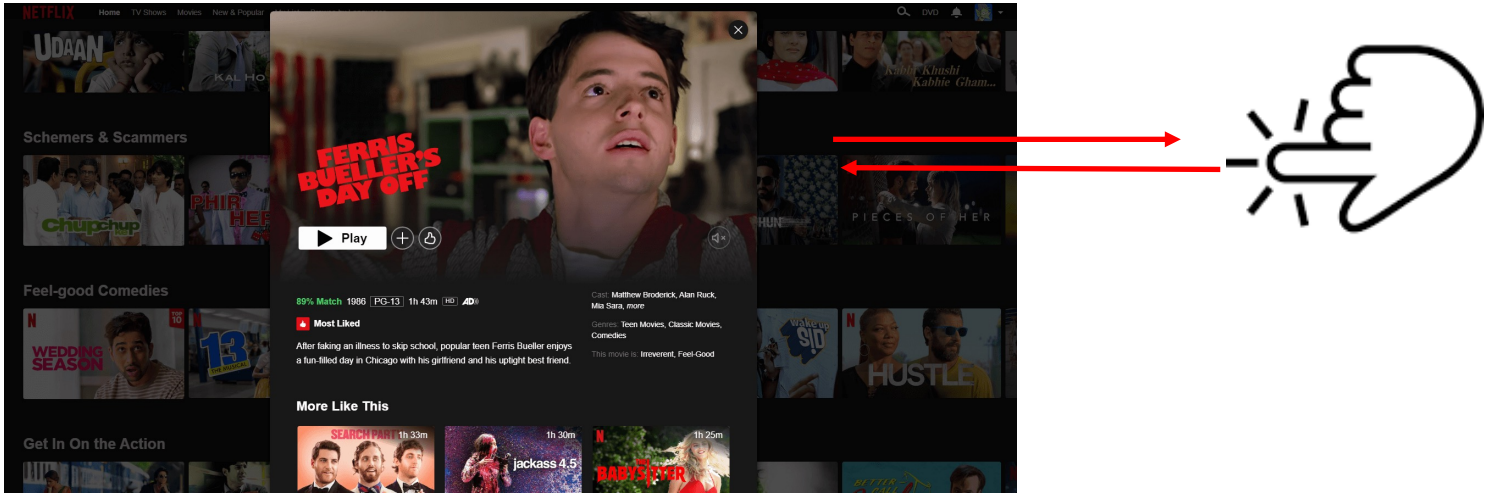
Back to Movie Recommendation



- Multi-armed Bandit:
  - Player → Recommendation system
  - Arms → Movies
  - Reward → User Click
- Contextual Bandit:

  The features (or side information) are available!

Back to Movie Recommendation



- Reward (User Click):

  Generalized linear model

  - Click or not click → Binary
  - Number of clicks → Count
  - ⋮

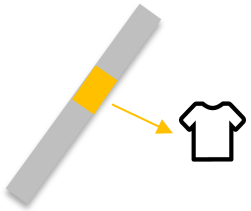
  → Logistic model; Poisson model; ⋮

We focus on the Generalized Linear Contextual Bandits!

Generalized Linear Contextual Bandit

Reward = μ(< Features, parameter θ>) + Noise

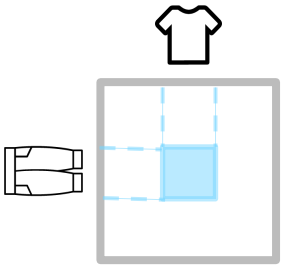
- When the features are in vector form:



$(d \times 1)$

  - Generalized linear contextual bandit [Filippi et al. 2010]
  - What if $\boldsymbol{\theta}$ is sparse? ⇒ Lasso bandit [Oh et al. 2021]

- When the features are in matrix form:



$(d \times d)$

  - What if $\text{rank}(\boldsymbol{\theta}) \ll \boldsymbol{d}$?

$$\boldsymbol{\theta} = \sum_{k=1}^{r} \boldsymbol{\sigma_i u_i v_i^T}$$ ⇒ $\sim dr$ unknowns

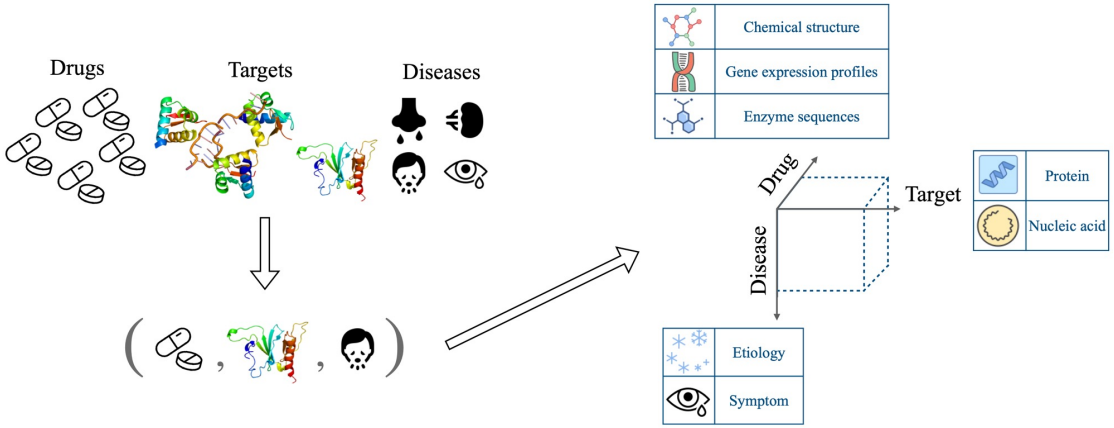
Low-rank bandit [Kang et al. 2022, Lu et al. 2021]

- What about higher order data form?

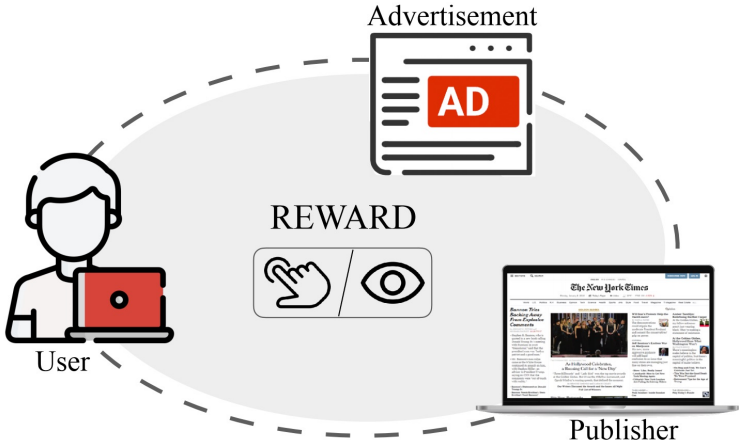
⟹ Generalized low rank tensor bandits! (Our research)

Motivating Examples

Precision medicine: continuous reward (IC50)

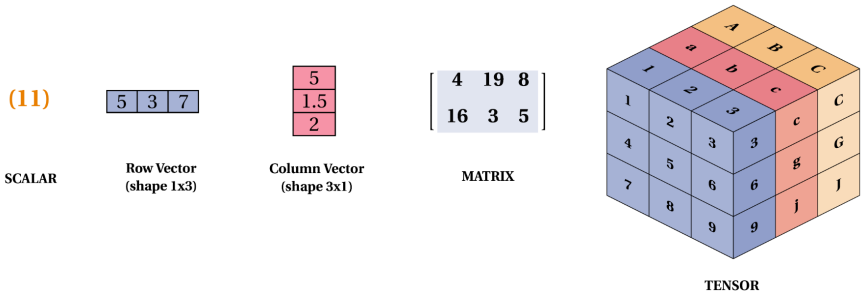


Advertising placement: binary reward (click, not click)



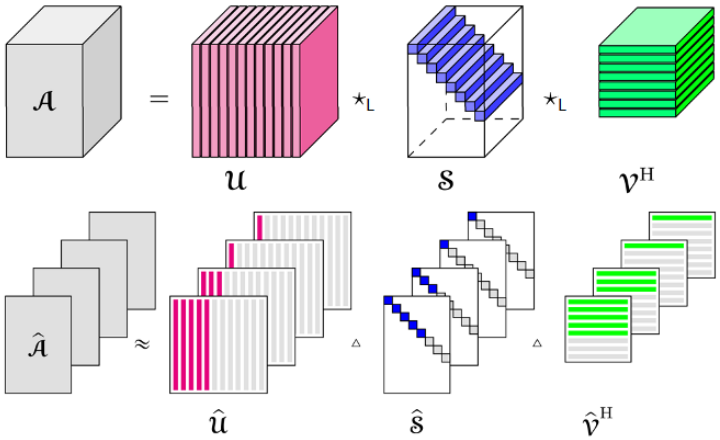
High-order (Tensor) Representation

- High-order data representation: Tensor



- A popular tensor decomposition method: Tensor SVD

  efficient computations, solid mathematical foundations



- Characterization of low rankness: Tensor nuclear norm

ME Kilmer, PNAS, 2021

Generalized Low Rank Tensor bandit

- Problem formulation：

$$y_t = \mu\langle \mathcal{X}_t, \mathcal{W}^* \rangle + \eta_t$$

where $\langle\cdot\rangle$ denotes the inner product, η_t is random noise, and $\mathcal{W}^* \in \mathbb{R}^{d_1 \times d_2 \times d_3}$ is an unknown parameter with the tubal rank $r \ll \min\{d_1, d_2\}$

Generalized linear model	Inverse link $\mu(x)$	Types of reward
Linear	$\mu(x) = x$	continuous
Binary logistic	$\mu(x) = \frac{e^x}{1+e^x}$	binary
Poisson	$\mu(x) = e^x$	count

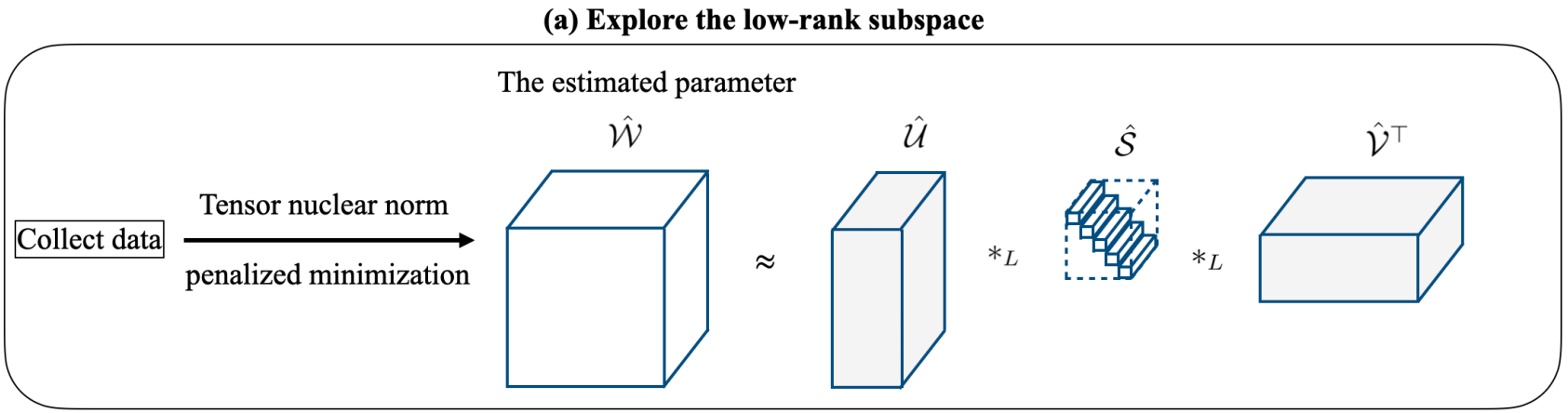
- Goal: Minimize the cumulative regret

$$R_T = \sum_{t=1}^{T} (\mu\langle \mathcal{X}^*, \mathcal{W}^* \rangle - \mu\langle \mathcal{X}_t, \mathcal{W}^* \rangle)$$

where $\mathcal{X}^* = \arg\max_{\mathcal{X} \in \mathbb{X}} \mu\langle \mathcal{X}, \mathcal{W}^* \rangle$

Measurement

Algorithm

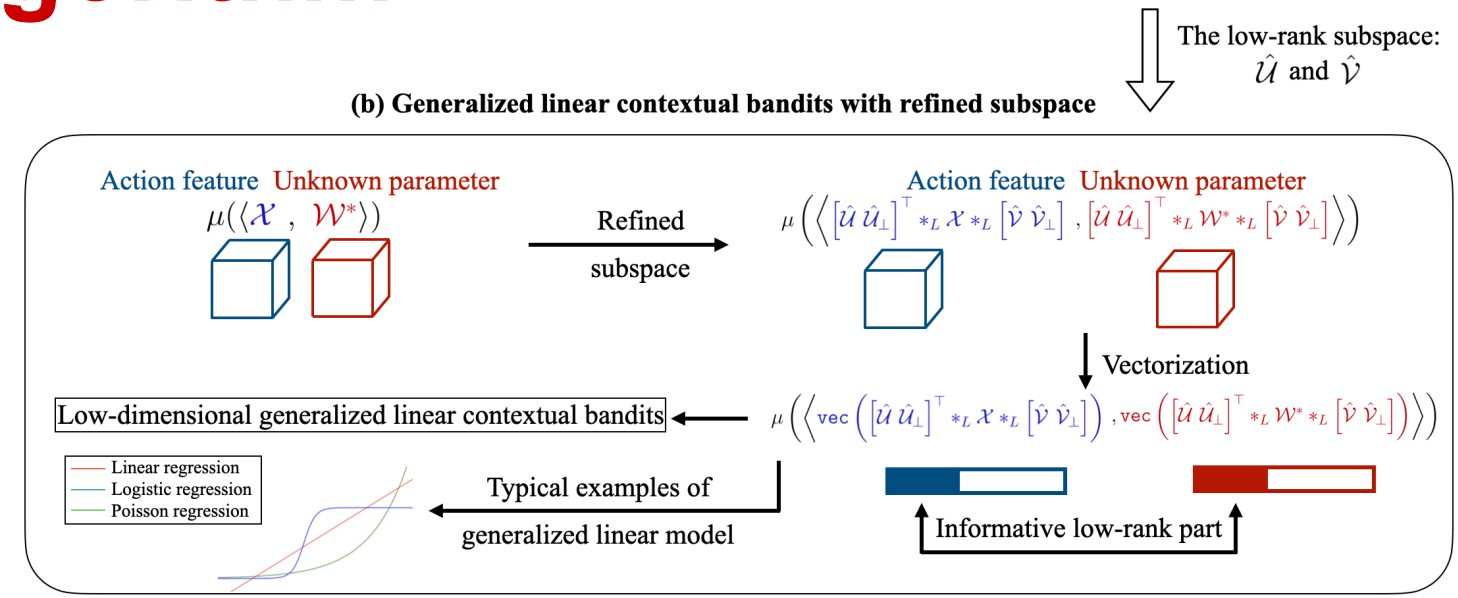


- Explore the low-rank subspace:

  (1) Randomly collect raw data (action features and reward),

  (2) Estimate $\mathcal{W}^*$ by solving a tensor nuclear norm penalized minimization

Algorithm



- Generalized linear contextual bandits with refined subspace:

(1) Using the above estimated low-rank subspace to make transformation

(2) Solve a low-dimensional generalized linear bandits (UCB-based algorithm)

i.e., choose the optimal arm according to

argmax Estimated reward (Exploitation) + Confidence interval/uncertainty (Exploration)

Theoretical Results

Comparison algorithms:

- Generalized linear bandits algorithm
- Generalized low-rank matrix bandits algorithm

Table 1 **Performance comparison with other generalized linear bandit algorithms in rounds T. The dimension of action feature is $d \times d \times d$, the unknown parameter in the generalized linear model is defined as $\mathcal{W}^*$, and ℓ is the coefficient of the invertible linear transform, we can take a transform with $\ell = 1$ in practice**

Algorithm	Regret bound	Order of d	Comment
GLM-UCB (Filippi et al. 2010)	$\widetilde{O}\left(d^3\sqrt{T}\right)$	3	
LowESTR (Lu et al. 2021a)	$\widetilde{O}\left(d^3\sqrt{\text{rank}(\mathcal{W}^*_{(1)})T}\right)$	3	
G-ESTT(Kang et al. 2022)	$\widetilde{O}\left(Md^2\,\text{rank}(\mathcal{W}^*_{(1)})\sqrt{T}\right)$	?	Under an impractical assumption; M is related to d in some cases
G-LowTESTR (this paper)	$\widetilde{O}\left(d^{5/2}\sqrt{\text{rank}_t(\mathcal{W}^*)\ell T}\right)$	2.5	

Synthetic Data Experiments

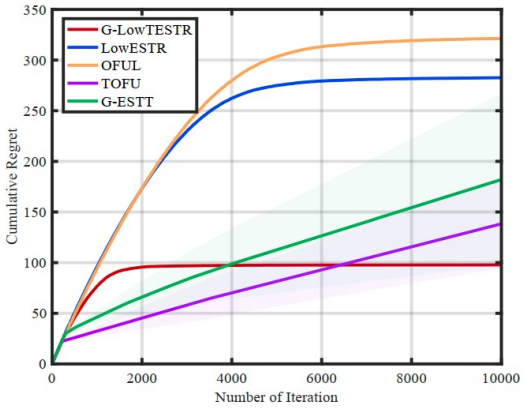
The process of generating rewards:

(1) Linear case: $y_t = \langle \mathcal{X}_t, \mathcal{W}^* \rangle + \eta_t$, where $\eta_t \sim N(0, 0.01^2)$.

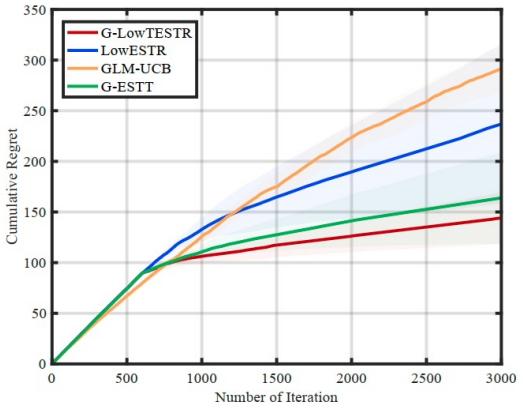
(2) Binary logistic case: $y_t \sim \text{Logistic}(p_t)$ with $p_t = \frac{1}{1+e^{-\langle \mathcal{X}_t, \mathcal{W}^* \rangle}}$.

(3) Poisson case: $y_t \sim \text{Poisson}(\nu_t)$ with $\nu_t = e^{\langle \mathcal{X}_t, \mathcal{W}^* \rangle}$.

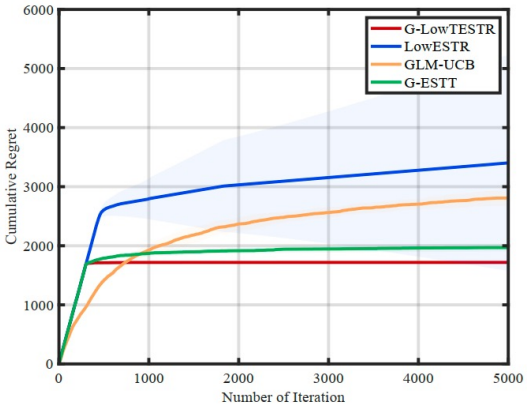
Comparison analysis with other popular algorithms:



(a) Linear case



(b) Binary logistic case



(c) Poisson case

Operations Management Applications

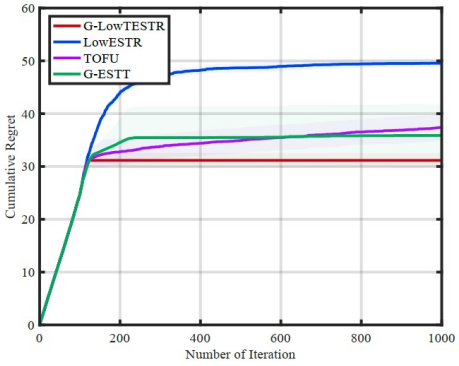
Applications:

(1) Linear case: Precision medicine (CCLE dataset). Choose the best treatment for patients with cancer.

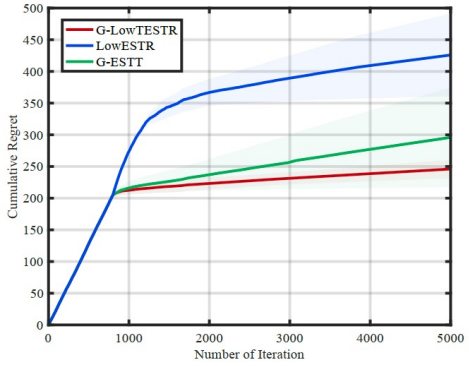
(2) Binary logistic case: Movie recommendation (IMDB dataset). The system recommends [movie, director, actor] for each user.

(3) Poisson case: Topic modeling on multiway publications (Duke publications dataset). Study the optimal the [author, word, venue] triplet.

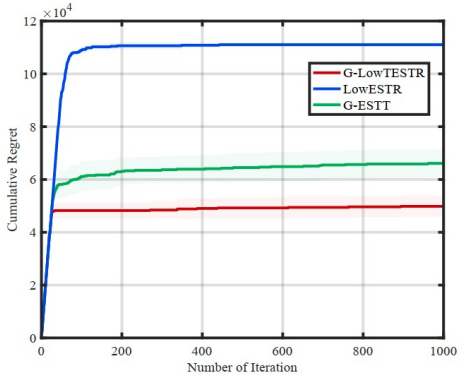
Comparison analysis with other algorithms:



(a) Precision medicine



(b) Movie recommendation



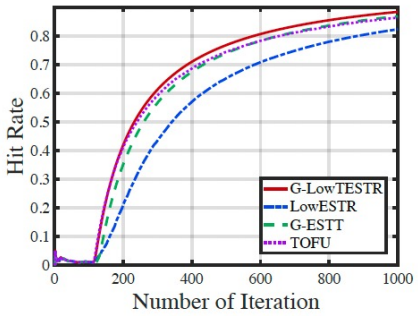
(c) Topic modeling

Operations Management Applications

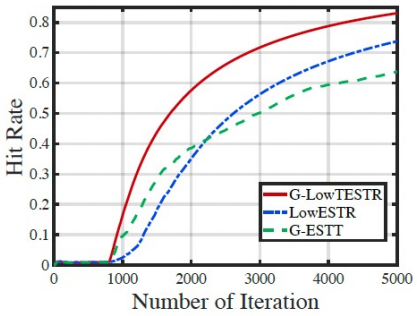
- Hit Rate: measures the frequency of selecting the optimal arms

$$\text{Hit Rate } (t) := \frac{1}{t}\sum_{i=1}^{t}\mathbb{I}(k_t \in K)$$

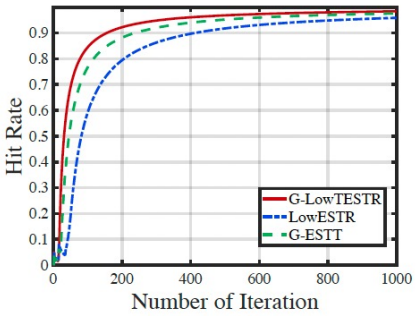
- A higher Hit Rate reflects a more precise selection of optimal arms



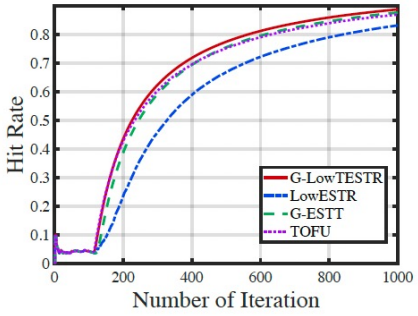
(a) Precision medicine (top 1%)



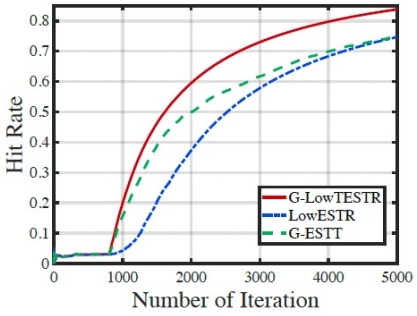
(b) Movie recommendation (top 1%)



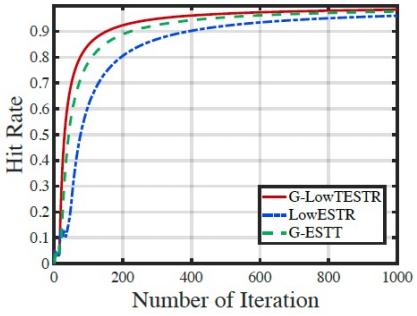
(c) Topic modeling (top 1%)



(d) Precision medicine (top 3%)



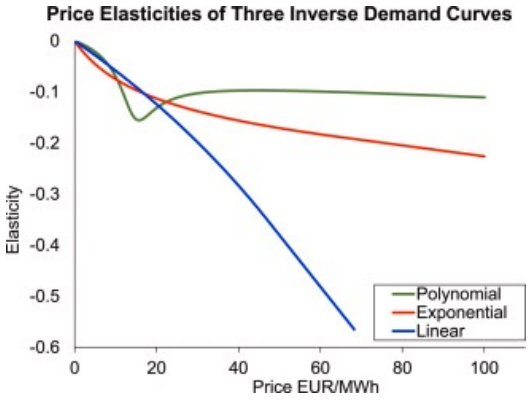
(e) Movie recommendation (top 3%)



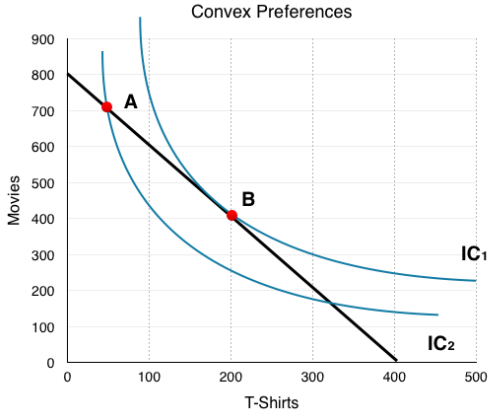
(f) Topic modeling (top 3%)

Neural Tensor Bandits

- Goal: To solve the optimization problem of nonlinear reward function in more complex tensor tasks
- Neural Tensor Bandits: Combined with deep learning and bandits, non-linear rewards can be better learned



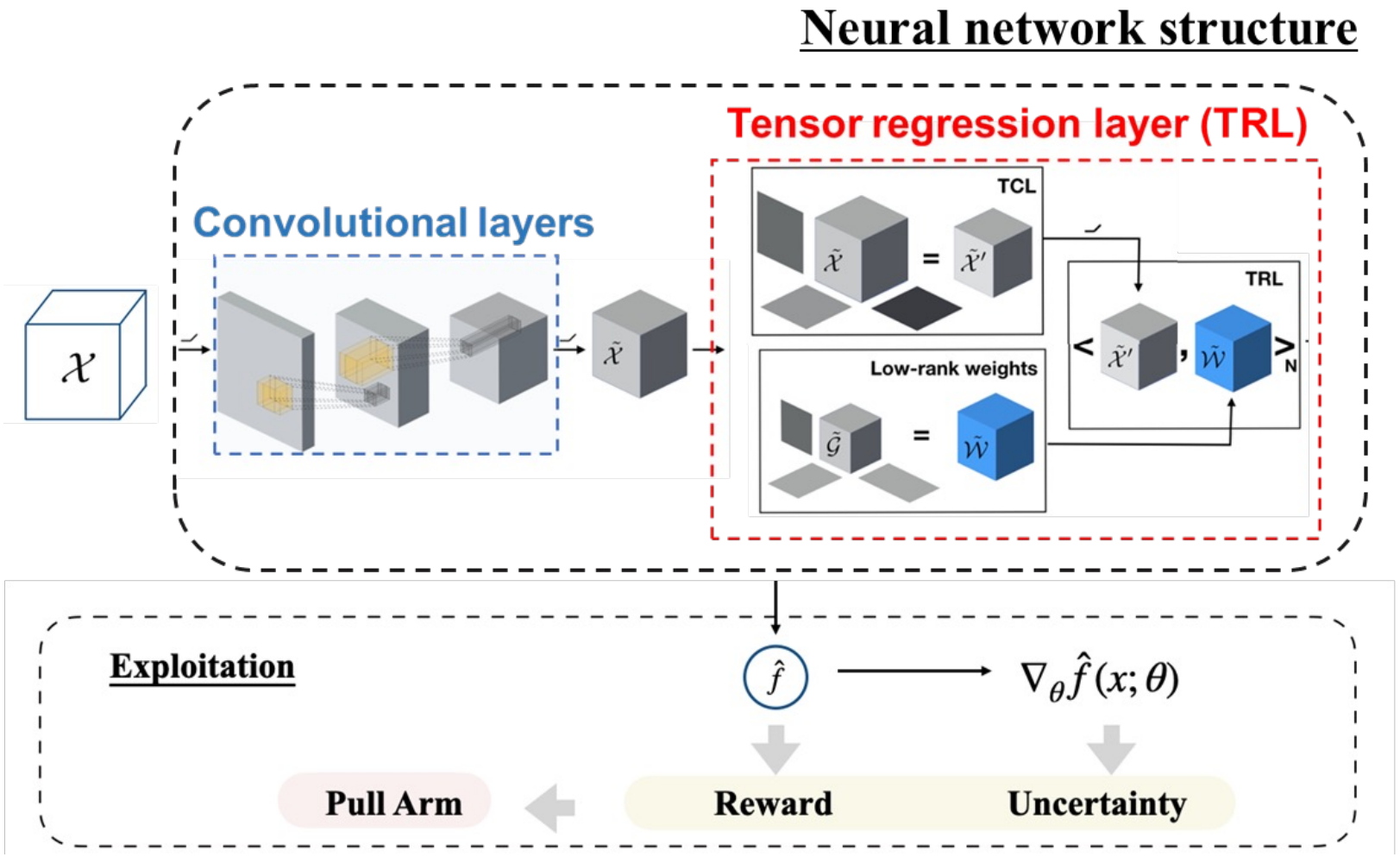
demand



preference

Nonlinear Reward

Neural Tensor Bandits



Experiments

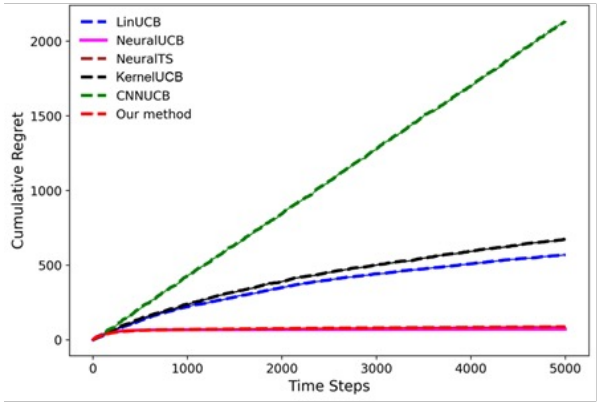
The process of generating rewards:

(1) Linear case: $y_t = \langle \mathcal{X}_t, \mathcal{W}^* \rangle + \eta_t$, where $\eta_t \sim N(0, 0.01^2)$.

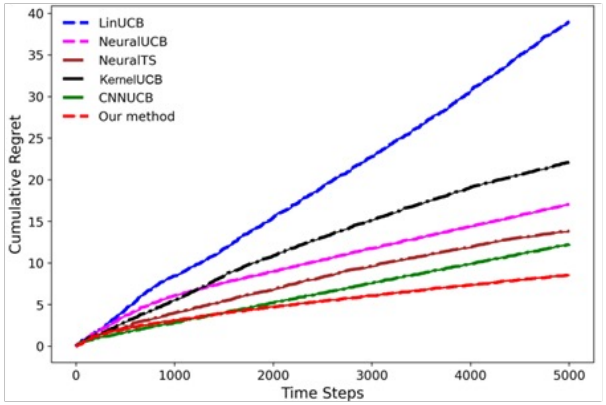
(2) Square case: $y_t = \langle \mathcal{X}_t, \mathcal{W}^* \rangle^2 + \eta_t$, where $\eta_t \sim N(0, 0.01^2)$.

(3) Cosine case: $y_t = cos(2\pi \langle \mathcal{X}_t, \mathcal{W}^* \rangle) + \eta_t$, where $\eta_t \sim N(0, 0.01^2)$.

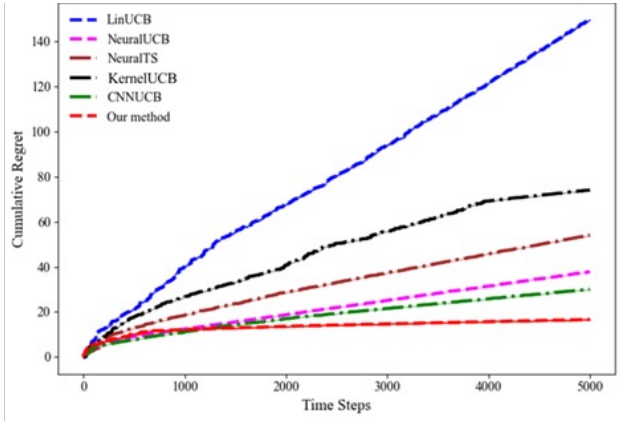
Comparison analysis with other popular algorithms:



(1) Linear case



(2) Square case



(3) Cosine case

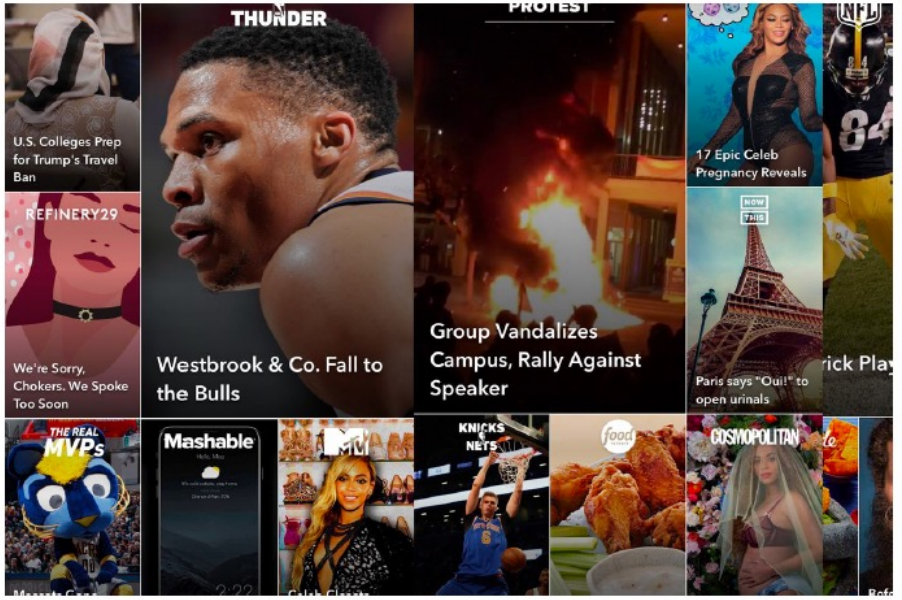
Outline

- Introduction
- Generalized Tensor Contextual Bandits
- **Deep RL for Online Assortment Customization**
- Distributed Q-Learning for DTRs

Li et al., POM, 2025

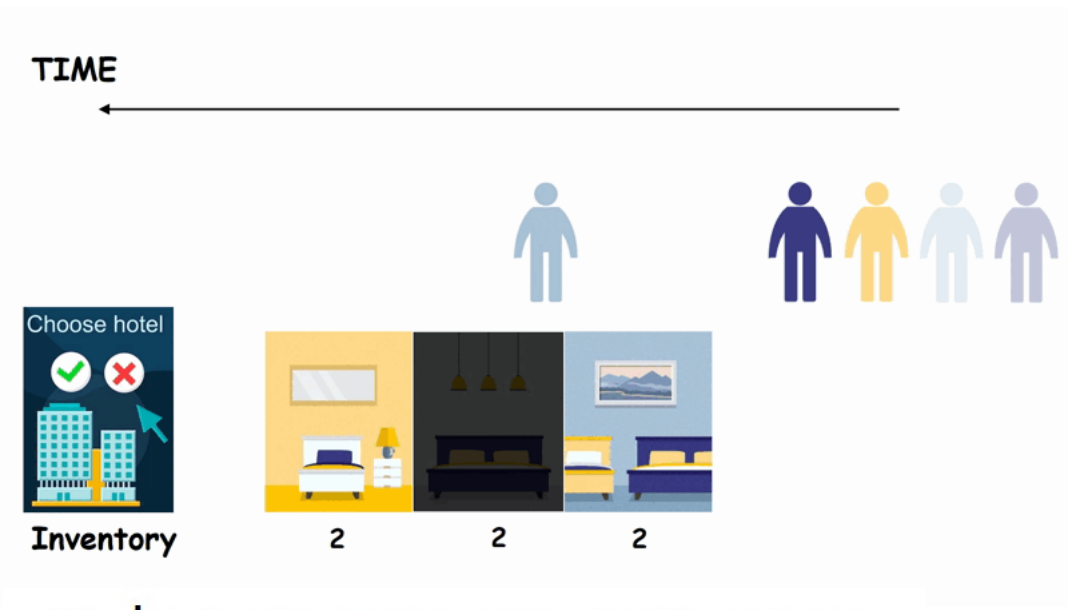
What is Assortment Optimization?

The collection of goods or services that a business provides to consumers.



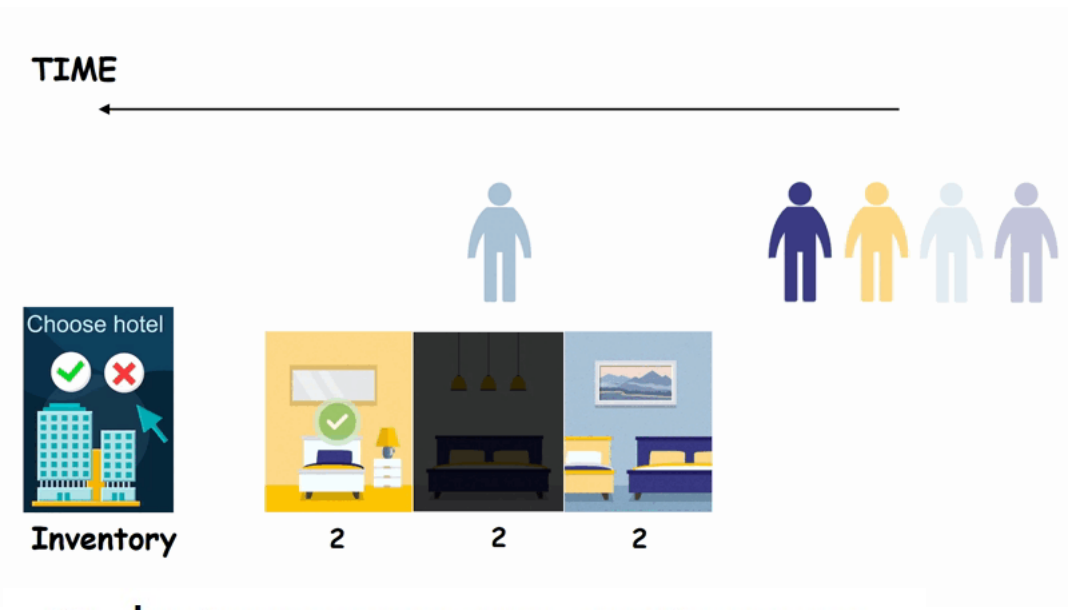
Talluri and Ryzin, 2004, MS; Rusmevichientong, Shen and Shmoys, 2010, MS; Rusmevichientong, Sumida and Topaloglu, 2020, MS; Gao et al., 2021, OR; Will Ma, 2023, MS ...

Online Assortment Optimization



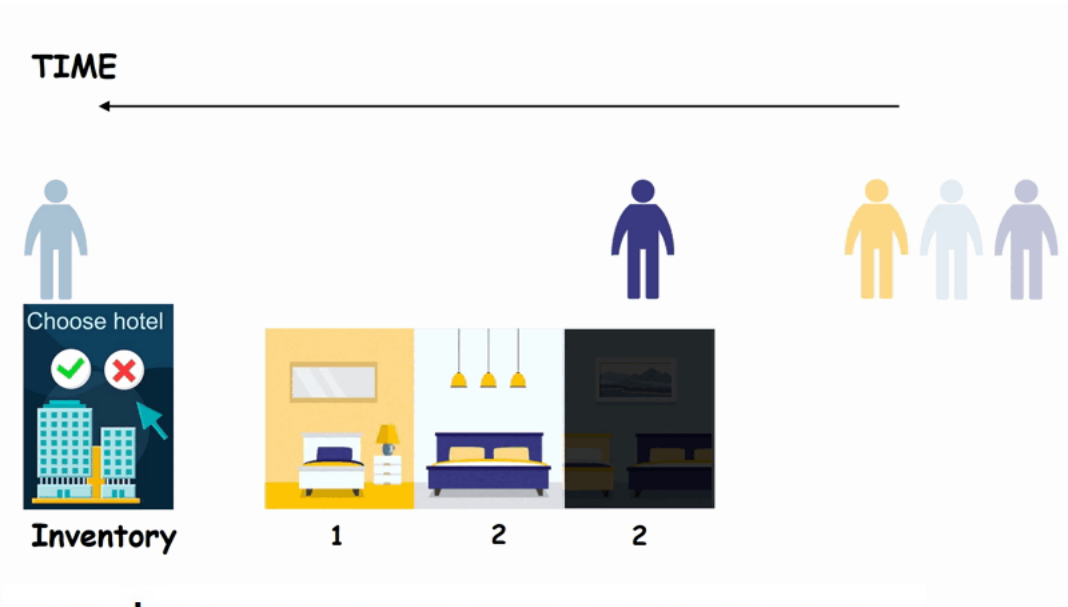
- heterogeneous customers
- limited inventory
- capacity constraint

Online Assortment Optimization



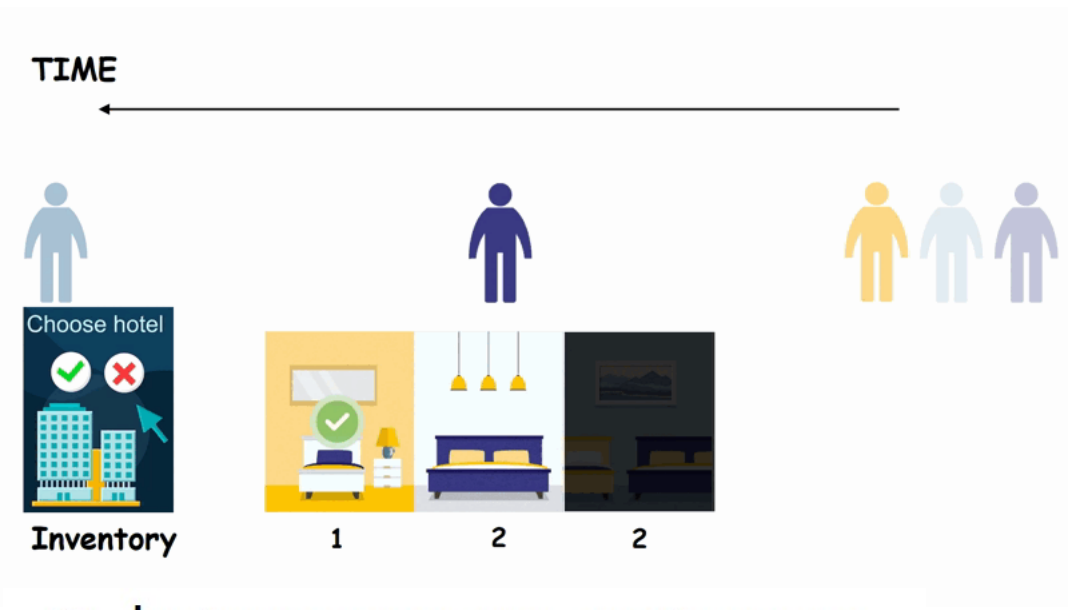
- heterogeneous customers
- limited inventory
- capacity constraint

Online Assortment Optimization



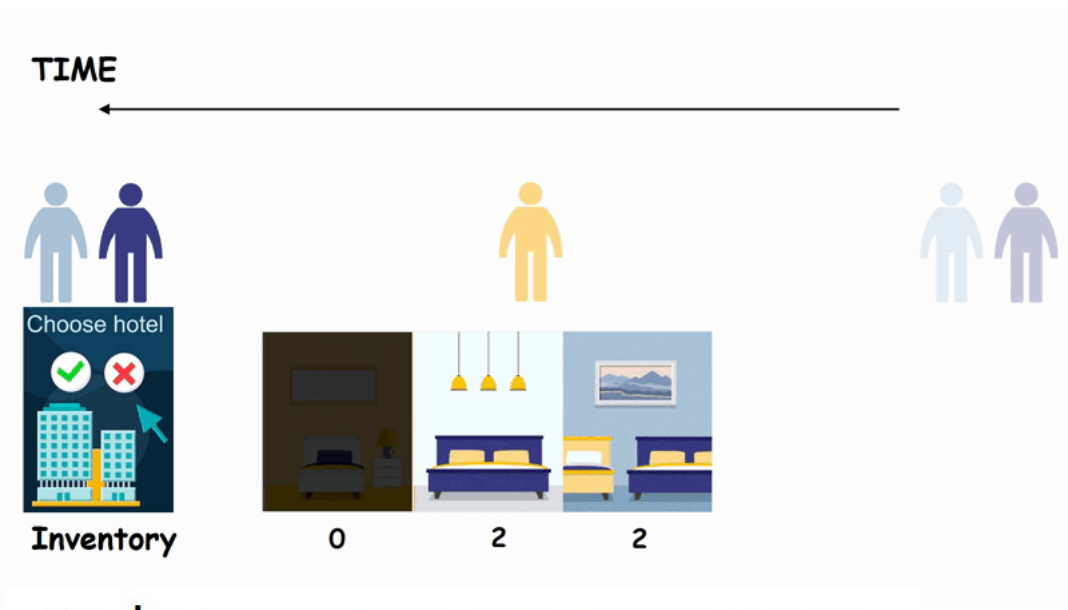
- heterogeneous customers
- limited inventory
- capacity constraint

Online Assortment Optimization



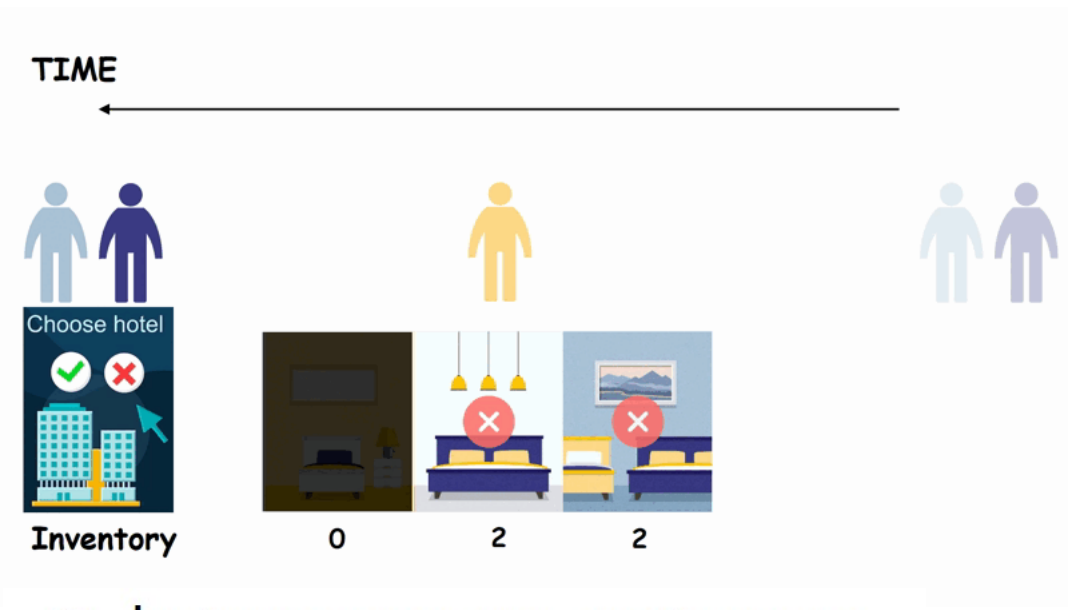
- heterogeneous customers
- limited inventory
- capacity constraint

Online Assortment Optimization



- heterogeneous customers
- limited inventory
- capacity constraint

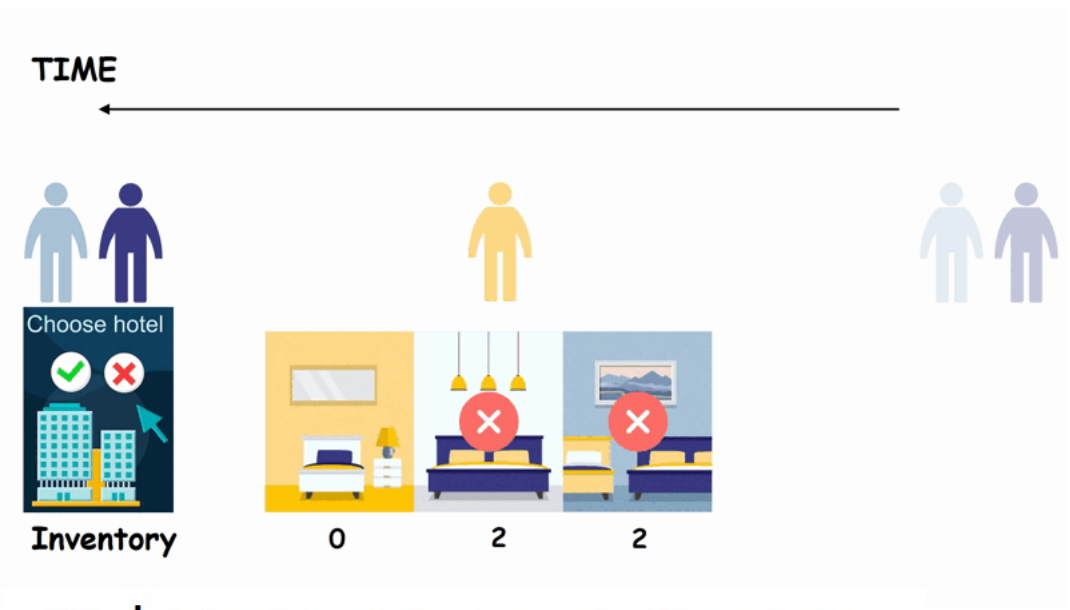
Online Assortment Optimization



- heterogeneous customers
- limited inventory
- capacity constraint

Online Assortment Optimization

- Data: Historical arrival sequences + transaction data



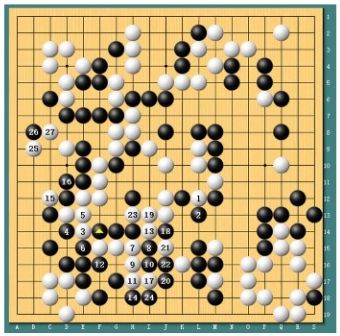
- heterogeneous customers
- limited inventory
- capacity constraint

Challenges

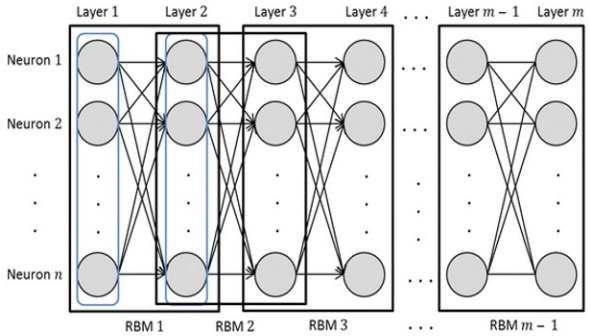
- Large action space, actions in different decision stages are time-dependent
- The arrival order of customer types is unknown beforehand
- Hard to capture complex customer choice behaviour
  - current works consider this problem under certain discrete choice models
    - Bernstein et al. (2015) considered the MNL choice model
    - Golrezaei et al. (2014), Rusmevichientong et al. (2020), and Gong et al. (2022) considered a class of parametric choice models under a regularity assumption
  - hard to optimize assortment under more complex choice models even in a single stage

Deep Reinforcement Learning

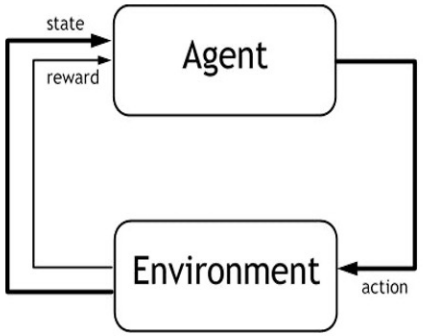
An agent learns by interacting with an environment



30 million



Deep neural networks



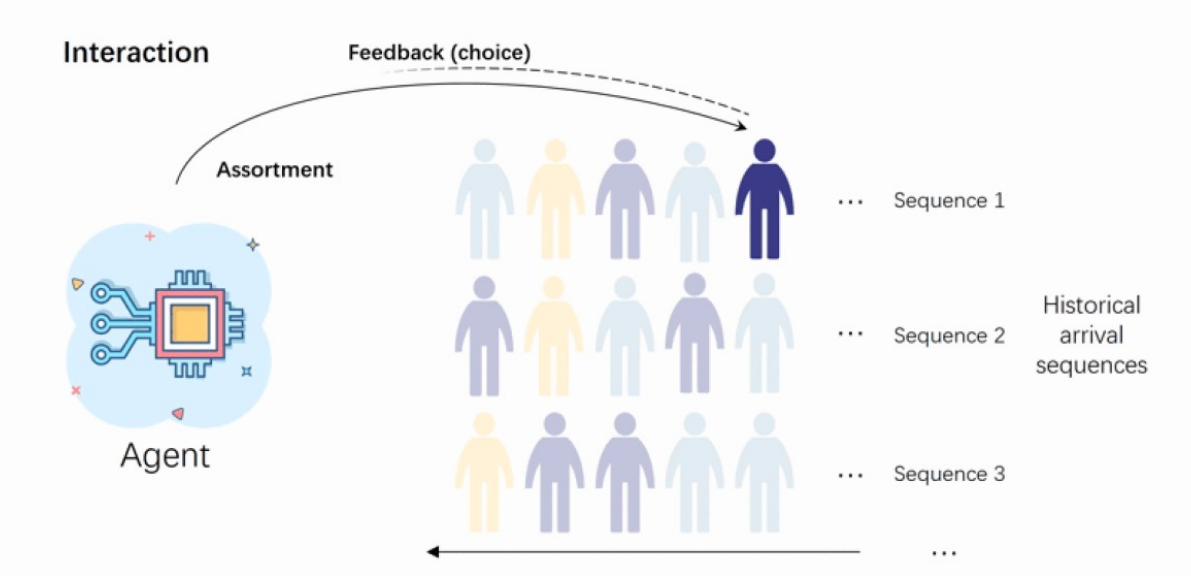
Reinforcement learning

Applications in Operations Management/Operations Research

- **Combinatorial optimization**: TSP (Bello et al., 2017, *ICLR*), VRP (Nazari et al., 2018, *NIPS*)
- **Inventory management**: Oroojlooyjadid et al., 2022, *M&SOM*; Gijsbrechts et al., 2022, *M&SOM*
- **Online Bipartite Matching**: Alomrani et al., 2021, *TMLR*
- ...

Our DRL based Procedure

- Interaction:
  Agents (seller) $\leftrightarrow$ Environment (simulated customers' behaviour)



- **Advantage**
  - Learn from historical arrival data without imposing an assumption on customer distribution
  - No restriction for choice model

Problem Setting

- Consider a finite selling horizon with length T
- Each time t comes a customer $\mathbf{z}_t \in \mathcal{Z}$, whose preference is expressed by $p_i^{\mathbf{z}_t}(\mathbf{a}_t)$ when faced with assortment $\mathbf{a}_t$:

$$p_i^{\mathbf{z}_t}(\mathbf{a}_t) = 0, \forall i \notin \mathbf{a}_t; \sum_{i \in \mathbf{a}_t} p_i^{\mathbf{z}_t}(\mathbf{a}_t) + p_0^{\mathbf{z}_t}(\mathbf{a}_t) = 1 \tag{1}$$

- The seller holds $\mathcal{N}$ products, with fixed price $\mathbf{r} = (r_1, \ldots, r_N)$ and initial inventory levels $\mathbf{I}_0 = (I_{01}, I_{02}, \ldots, I_{0N})$;
  No inventory replenishment during the horizon
- The seller's goal is to develop a customized assortment policy π to maximize the total expected revenue:

$$\max_{\mathbf{a}_t \sim \pi} \mathbb{E}_{T, \mathbf{z}_1, \ldots, \mathbf{z}_T} \left[\sum_{t=1}^{T} \sum_{i \in \mathbf{a}_t} r_i p_i^{\mathbf{z}_t}(\mathbf{a}_t) \right]. \tag{2}$$

MDP Formulation

- Markov property:

$$P\left(S_n = s_n \mid S_{n-1} = s_{n-1}, \ldots, S_0 = s_0\right) =$$
$$P\left(S_n = s_n \mid S_{n-1} = s_{n-1}\right)$$

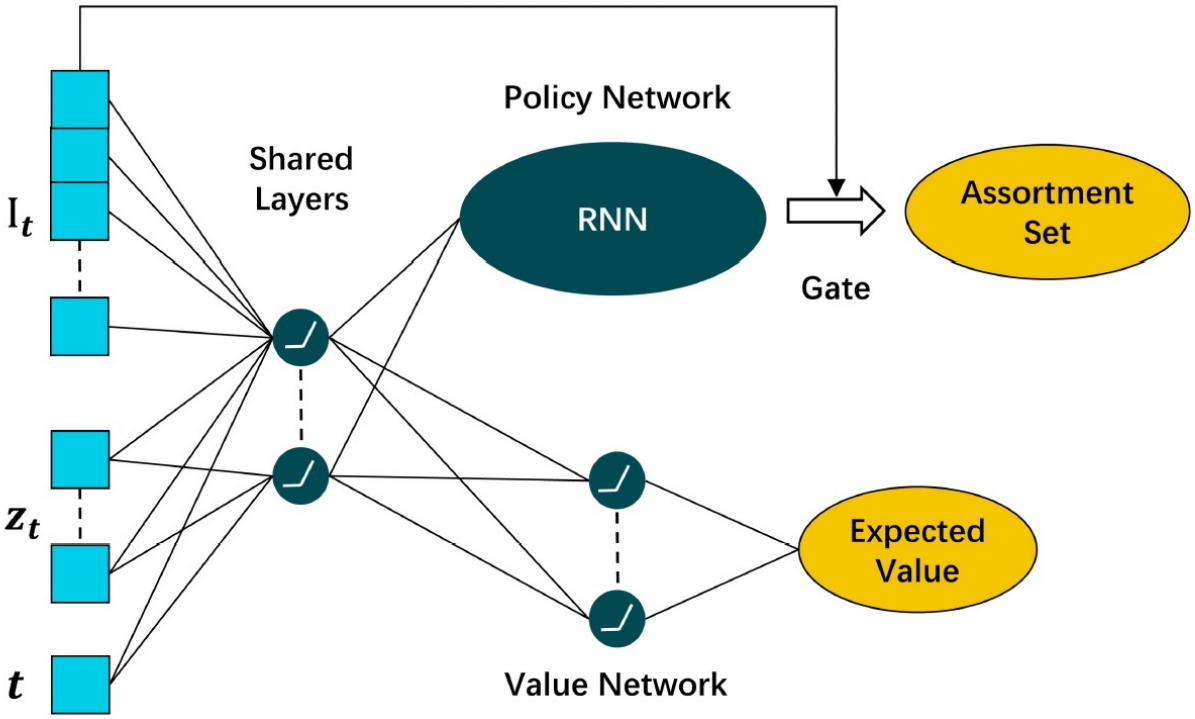
- Finite-horizon Markov Decision Process $\mathcal{M} = (\mathcal{T}, \mathcal{S}, \mathcal{A}, R, P)$

State	$\mathbf{S}_t = (\mathbf{I_t}, \mathbf{z}_t, t)$
Action	$\mathbf{a}_t \in A_t = \{\mathbf{a} \mid I_{ti} > 0, \forall i \in \mathbf{a}, \lvert\mathbf{a}\rvert < C\}$
Reward function	$R_t = \sum_{i=1}^{N} r_i(I_{ti} - I_{(t+1)i})$
Environment dynamics	$\mathbb{P}(I_{(t+1)i} = I_{ti} - 1) = p_i^{\mathbf{z}_t}(\mathbf{a}_t)$, $\mathbb{P}(I_{(t+1)i} = I_{ti}) = 1 - p_i^{\mathbf{z}_t}(\mathbf{a}_t)$, $\mathbf{z}_{t+1} \sim D(Z)$

- Policy π maps a state to an action

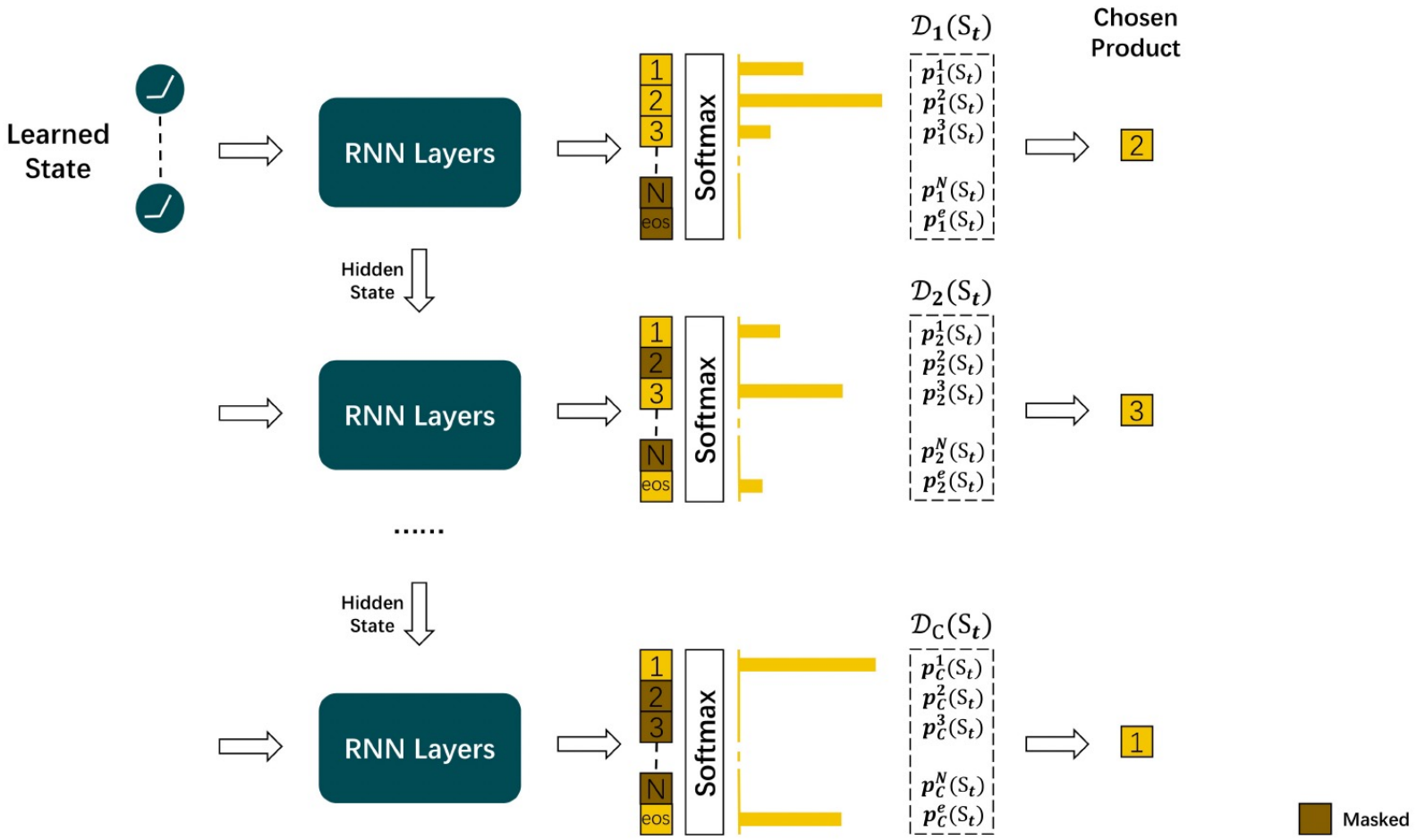
Model Architecture

- Q-learning-based Reinforcement Learning is hard to deal with the large action space.
- Actor-Critic Architecture
  - policy network: generate actions
  - value network: evaluate the performance of current policy



Policy Network

- Generate products one by one in a sequence



Training Algorithm - Advantage Actor-Critic (A2C)

- update

$$\theta_{k+1} = \theta_k - \alpha \nabla_\theta L_k \tag{3}$$

- loss

$$L_k = \beta_v L_k^v + \beta_p L_k^p + \beta_e L_k^e \tag{4}$$

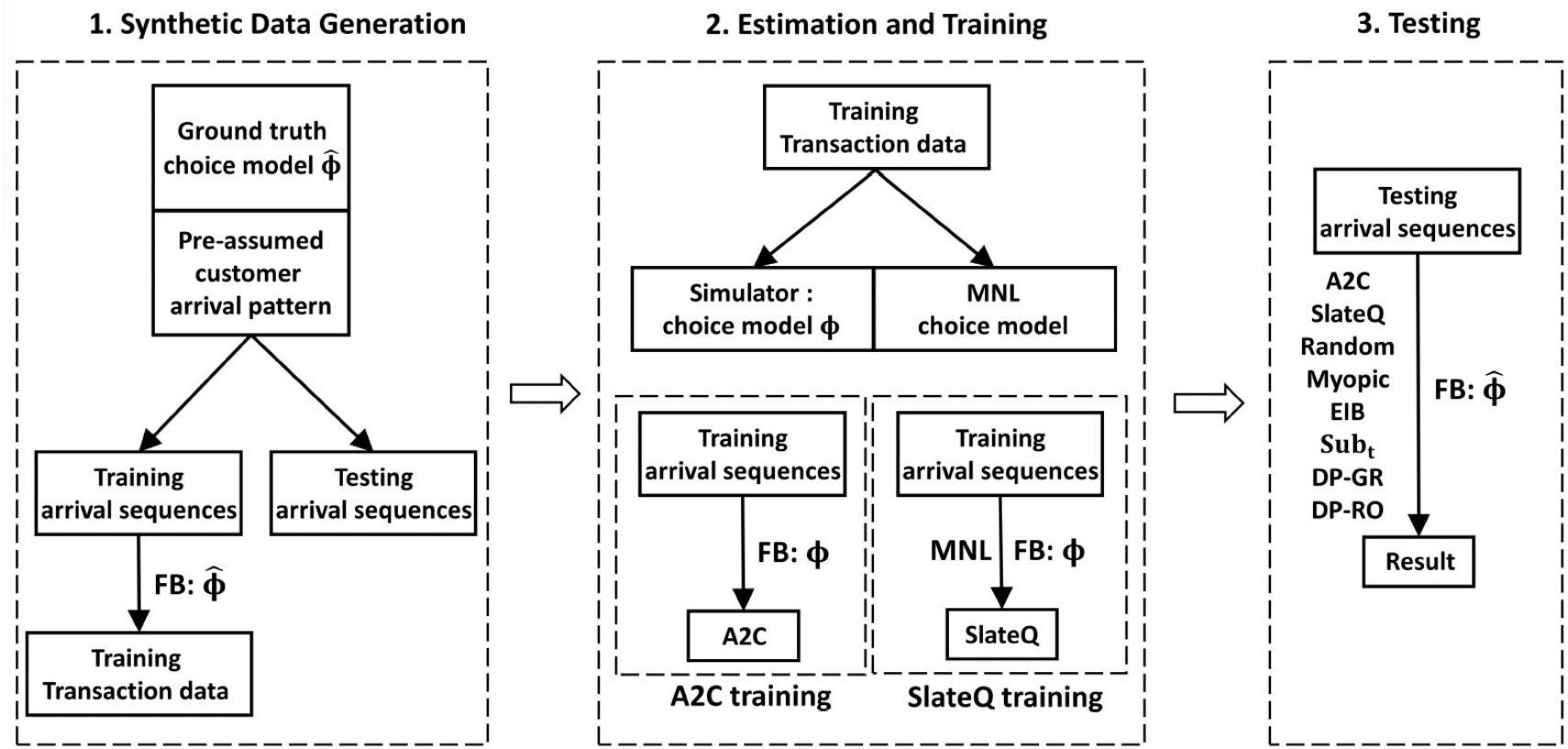
  - value loss L_k^v: how well the value network estimates
  - policy loss L_k^p: how well the generated action performs
  - entropy loss L_k^e: to ensure efficient exploration

- **Compared with policy gradient algorithms (e.g., REINFORCE), A2C helps stabilize the training process.**
- Actor-Critic Architecture is also used in the Literature (Gijsbrechts et al., 2022, *M&SOM*)

Comparison Methods

1. SlateQ-MNL (Ie et al., 2019, IJCAI): Q-learning-based DRL method
2. RA: offer $1 \sim C$ products randomly
3. Myopic-MNL: optimize one-period assortment based on MNL (doesn't consider limited inventory and future customers)
4. Sub_t-MNL (Bernstein, Kök, and Xie, 2015, *M&SOM*): heuristic which assumes the stochastic arrival pattern
5. EIB-MNL (Golrezaei, Nazerzadeh, and Rusmevichientong, 2014, *MS*): online algorithm which assumes the adversarial arrival pattern
6. DP-GR, DP-RO (Rusmevichientong, Sumida, and Topaloglu, 2020, *MS*): policies based on dynamic programming with linear value function approximations

Simulation With Ground Truth Feedback



Simulation With Ground Truth Feedback

- Ground truth choice model: rank-list choice model (Aouad, Feldman and Segev, 2023, *MS*)
  - Set the number of products as 10
  - 20 product-ranking lists
  - One customer type: a specific distribution over lists
  - Set the number of customer types as 4, from customer type 1 to 4, customers become more choosy
- Customer arrival pattern: a customer of type j arrives at time period t is proportional to $e^{-\kappa|t-\tau^j|}$
- Set the total number of customers as $T \sim U(110, 130)$
- 400 training sequences, 100 testing sequences

Offline LP Upper Bound

- **Offline**: after T and $\{z_1, z_2, \ldots, z_T\}$ have been revealed
- Solve an **LP** to get the relaxed maximum expected revenue

$$\text{maximize} \sum_{t=1}^{T} \sum_{S \in S} \sum_{i=1}^{n} r_i p_i^{z_t}(S) y^t(S)$$

$$\text{subject to} \sum_{t=1}^{T} \sum_{S \in S} p_i^{z_t}(S) y^t(S) \leq c_i \quad 1 \leq i \leq n$$

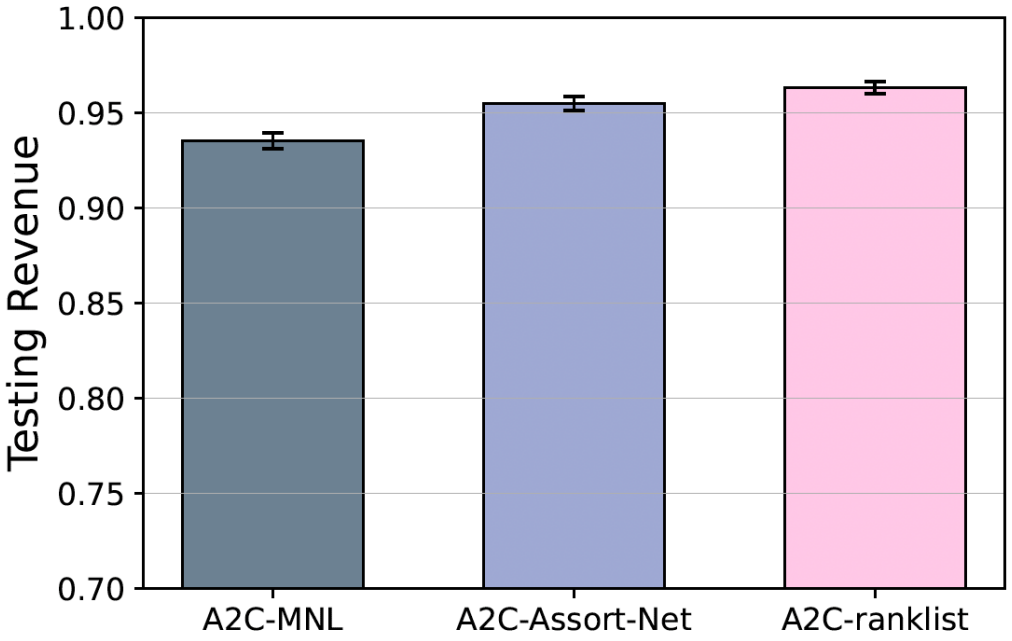
$$\sum_{S \in S} y^t(S) = 1 \quad 1 \leq t \leq T$$

$$y^t(S) \geq 0 \quad 1 \leq t \leq T, S \in S$$

- The optimal objective value can't be achieved since $y^t(S)$ is fractional, so it's an **Upper Bound**

Impact of Simulators

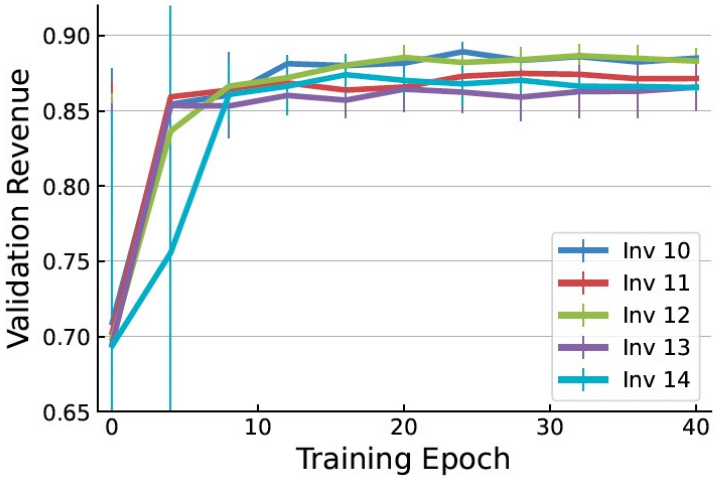
- Comparison of A2C with different simulators

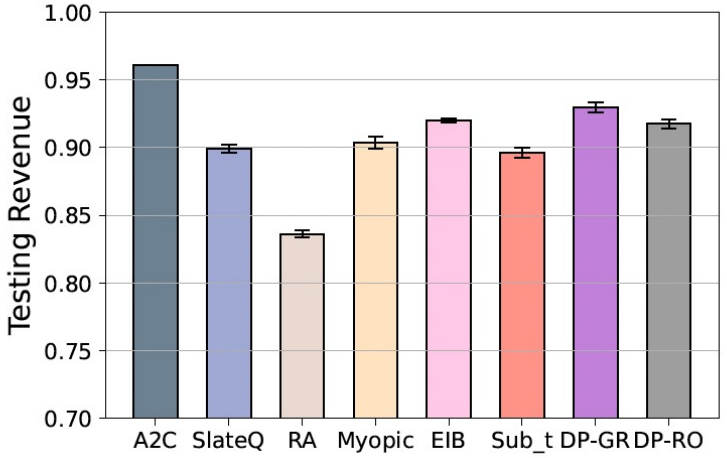


- **The more precise the simulator is, the better the A2C.**

Numerical Results I

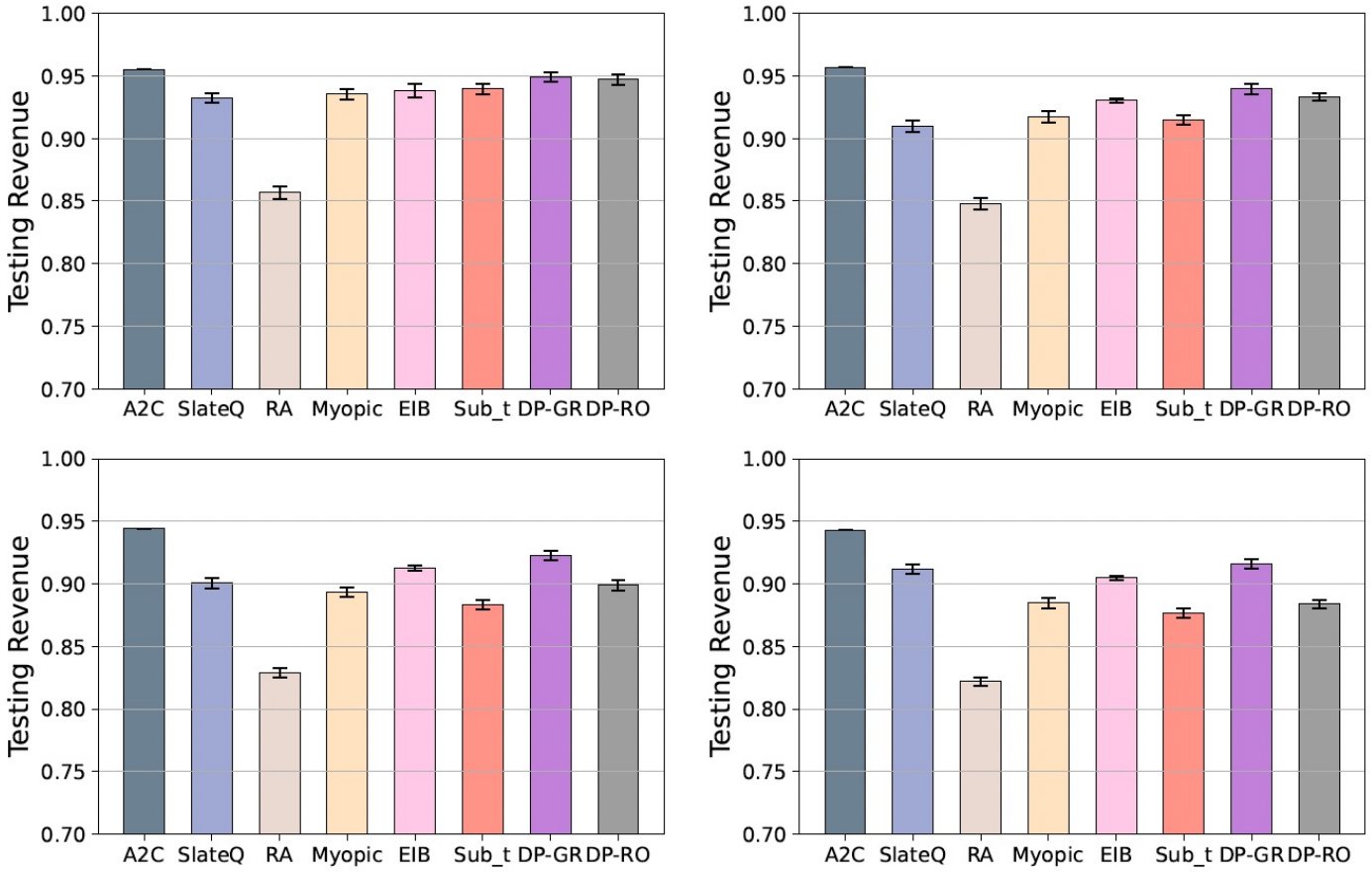
- Customer arrival pattern: more choosy customers arrive later
- Set the initial inventory level as 10, 11, 12, 13, 14
- Load factor $\mathbb{E}[T]/\sum_{i=1}^{10} I_{i0}$ equals 1.2, 1.1, 1.0, 0.92, 0.86
- Training curves (left)
  - One training epoch: interacting with all training customer sequences
- Result with LF 1.0 (right)





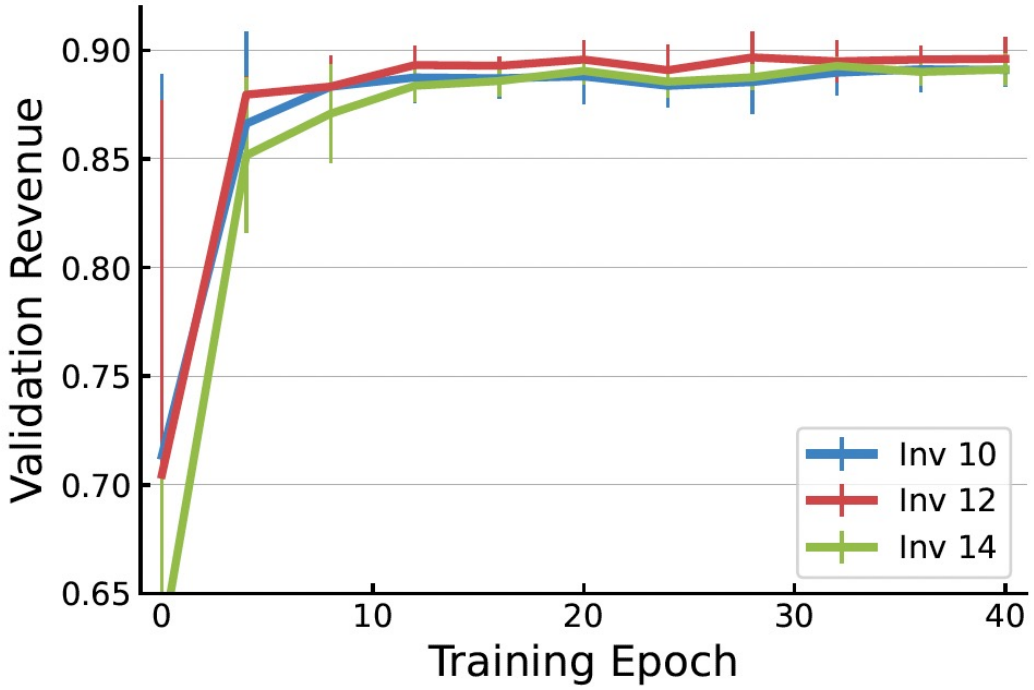
Numerical Results I

- Results with LF 1.2, 1.1, 0.92, 0.86 (from left to right, up to down)



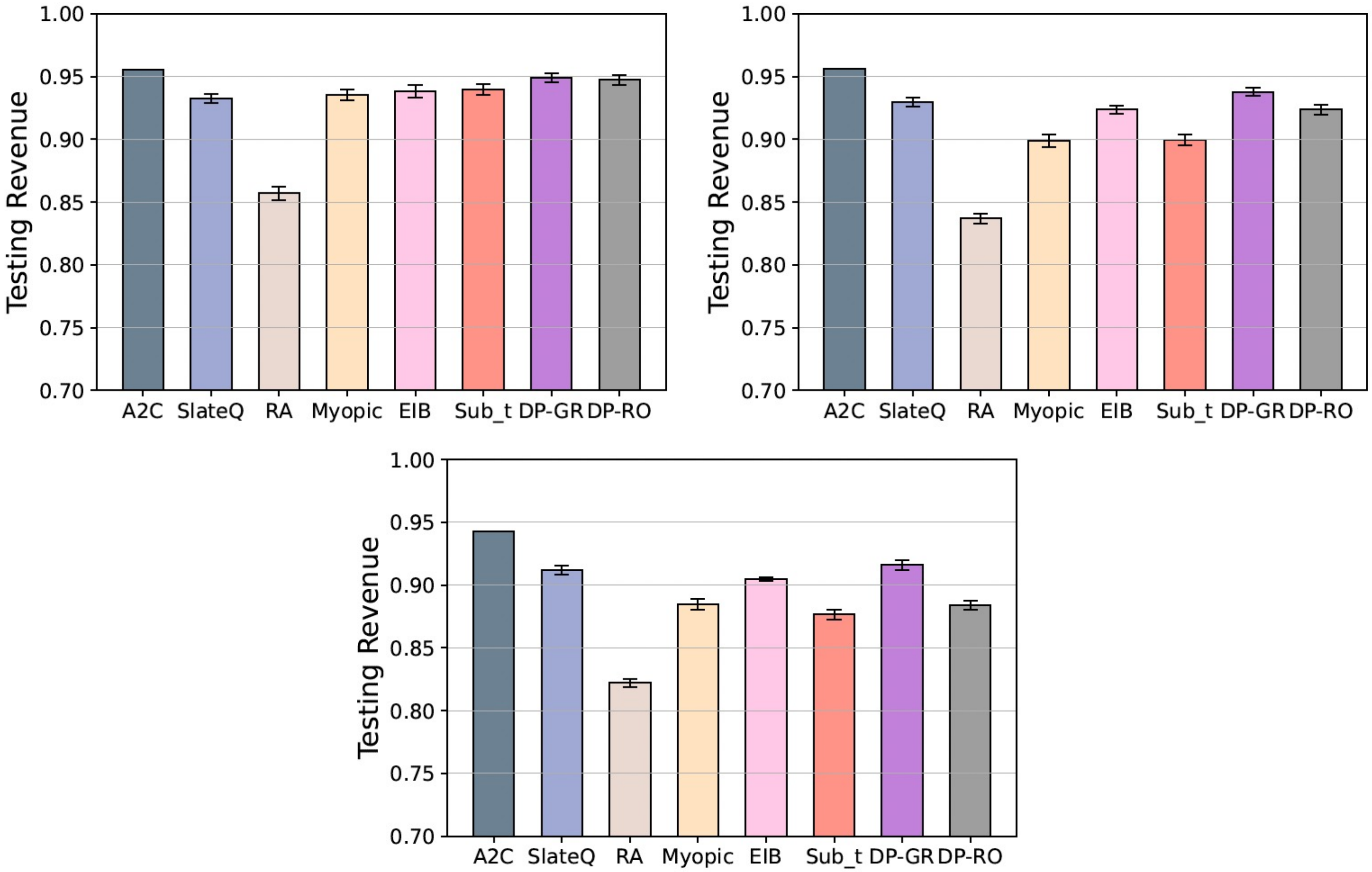
Numerical Results II

- Customer arrival pattern: uniform arrival
- Set the initial inventory level as 10, 12, 14
- Training curves



Numerical Results II

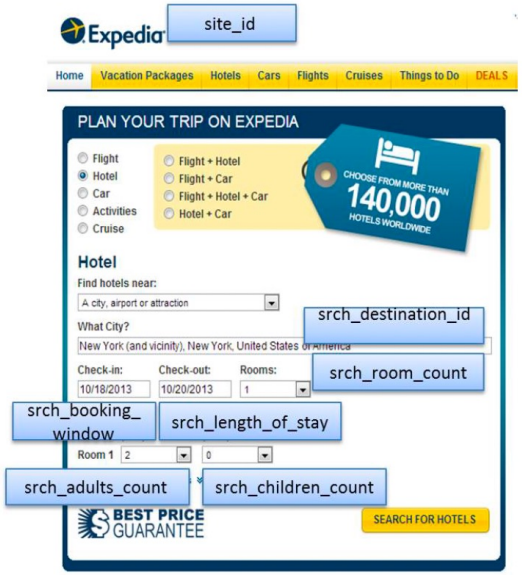
- Results with initial inventory level 10, 12, 14 (from left to right, up to down)



Real-World Data Experiment

- Expedia dataset

srch_id	date_time	site_id	visitor_location_country_id	visitor_hist_starrating	visitor_hist_adr_usd	prop_country_id	prop_id	prop_starrating	prop_review_score
2	2012/11/5 21:28	15	55	NULL	NULL	98	3105	3	2
2	2012/11/5 21:28	15	55	NULL	NULL	98	6399	3	0
2	2012/11/5 21:28	15	55	NULL	NULL	98	7374	4	3.5
2	2012/11/5 21:28	15	55	NULL	NULL	98	7771	3	4.5
2	2012/11/5 21:28	15	55	NULL	NULL	98	12938	3	0
2	2012/11/5 21:28	15	55	NULL	NULL	98	25579	3	4
2	2012/11/5 21:28	15	55	NULL	NULL	98	26540	3	3.5
2	2012/11/5 21:28	15	55	NULL	NULL	98	27090	4	0
2	2012/11/5 21:28	15	55	NULL	NULL	98	30434	4	3.5
2	2012/11/5 21:28	15	55	NULL	NULL	98	37331	3	4
2	2012/11/5 21:28	15	55	NULL	NULL	98	78858	4	4.5
2	2012/11/5 21:28	15	55	NULL	NULL	98	91899	3	3
2	2012/11/5 21:28	15	55	NULL	NULL	98	131173	4	5



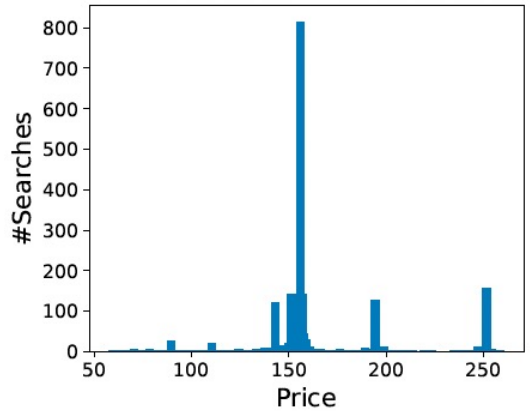
- choose **destination id 8192** - 126 hotels
- choose **30 most popular hotels** - 86.% transaction records

Real-World Data Experiment

- Product features

(1)	prop star rating
(2)	prop review score
(3)	prop brand bool
(4)	prop location score1
(5)	prop log historical price
(6)	price usd

- Customer segmentation

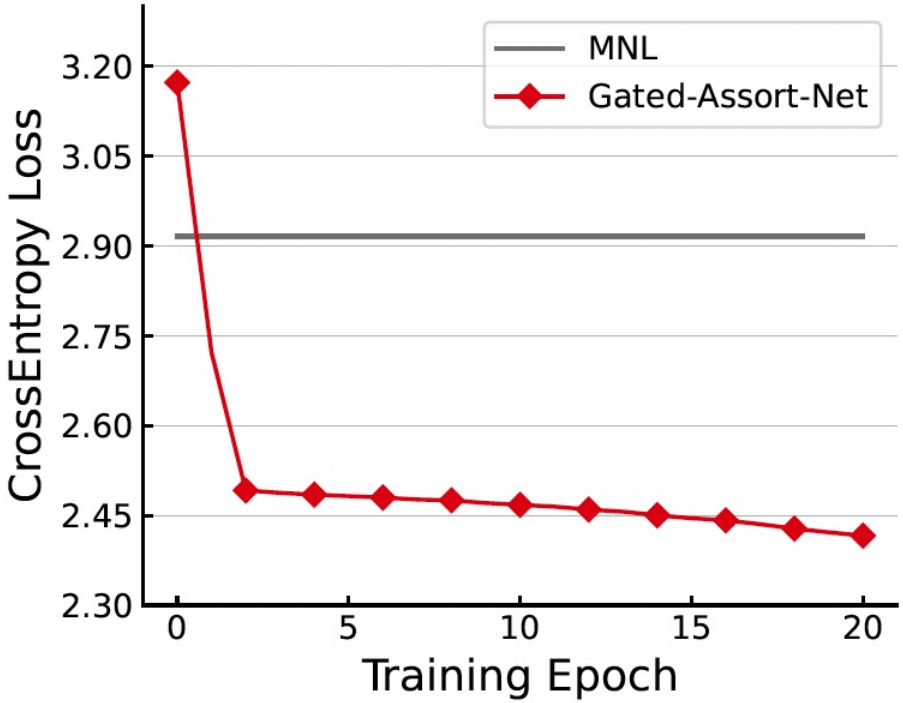


- Statistics of extracted data

# customer types	# hotels	# days	# search queries	# rows
4	30	211	4272	105158

Real-World Data Experiment

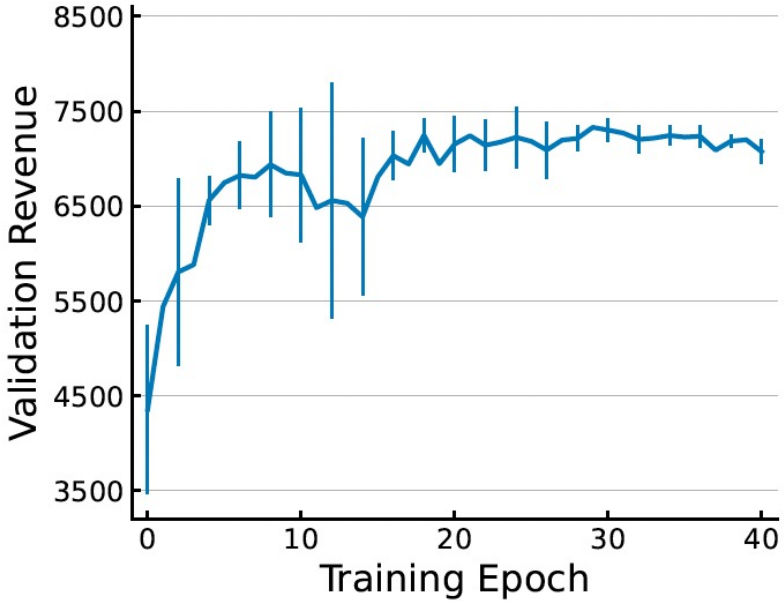
- Customer choice behaviour

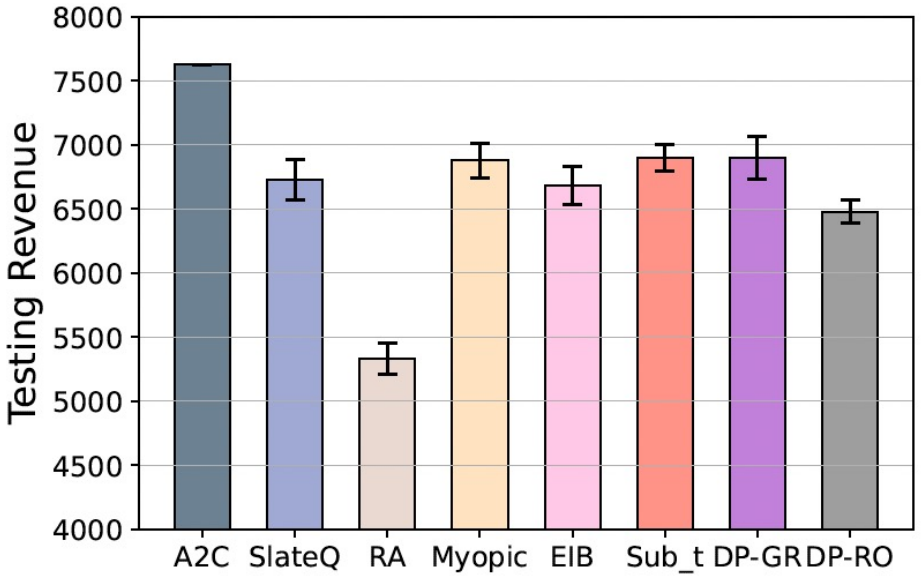


- **Gated-Assort-Net (Cai et al., 2022, arxiv) fits real transaction data better.**

Real-World Data Experiment

- Training curve and testing result





Scalability Test

- Previously, # products = 30
- Extra experiments on different numbers of products
- Both the testing result and the testing time demonstrate the scalability of our approach.

	N=10	N=30	N=100
A2C	133.4(12.0s)	143.0(12.1s)	202.8(15.0s)
Random	100(3.6s)	100(3.9s)	100(5.3s)
Myopic	113.0(16.8s)	129.1(34.1s)	88.0(75.3s)
EIB	107.1(16.7s)	125.2(34.0s)	68.3(72.1s)
Sub_t	113.6(109.0s)	129.4(886.0s)	86.4(7312.1s)
DP-Greedy	113.0(17.4s)	129.4(33.4s)	102.4(87.9s)

Extend to Reusable Products

- **A more general setting**
  Reusable products will come back into the inventory after some period of time since they are purchased by coming customers

Application scenarios

- fashion item rental
- cloud computing service
- shared parking

Modification

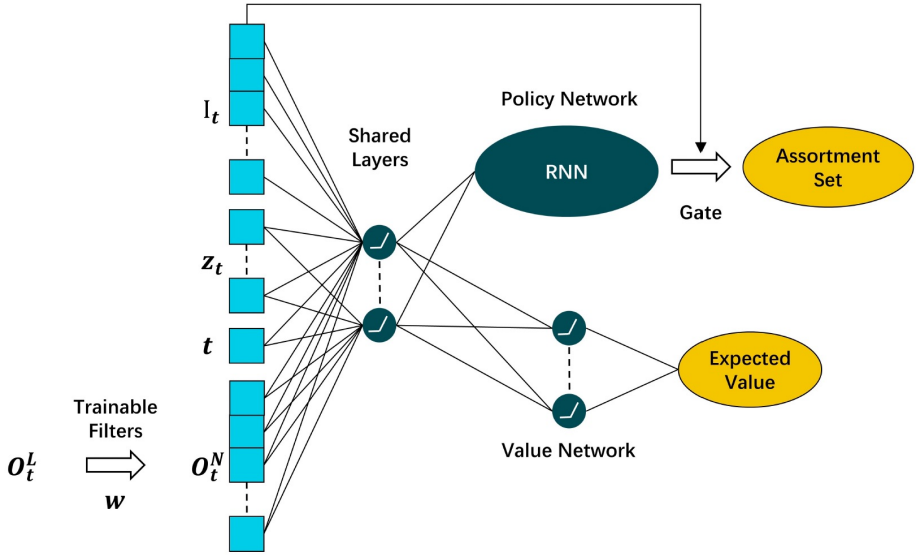
Modified state

$$\mathbf{S}_t = (\mathbf{I_t}, \mathbf{z}_t, t, \mathbf{O}_t^L)$$

- **past sales** $\mathbf{O}_t^L = \{\mathbf{o_{t-L}}, \ldots, \mathbf{o_{t-1}}\}, \mathbf{o_t} \in \{0, j\}^N, j \in \mathcal{Z}$

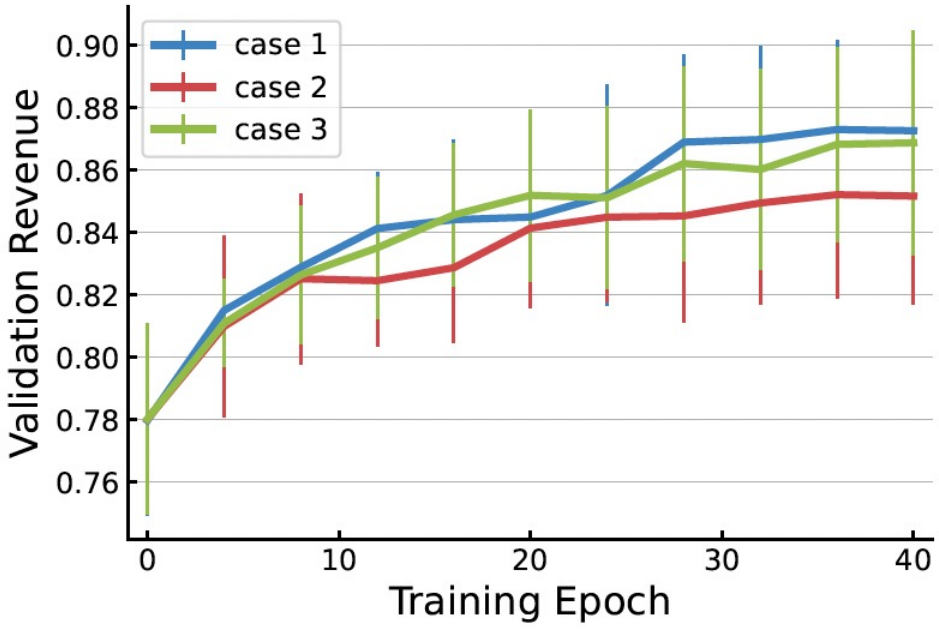
Time-series filters $[w_1, \ldots, w_N] \in R^{N \times L}$

$$\mathbf{O}_t^N = \sigma((\mathbf{O}_t^L)^T \cdot [w_1, \ldots, w_N]^T) \in R^N$$

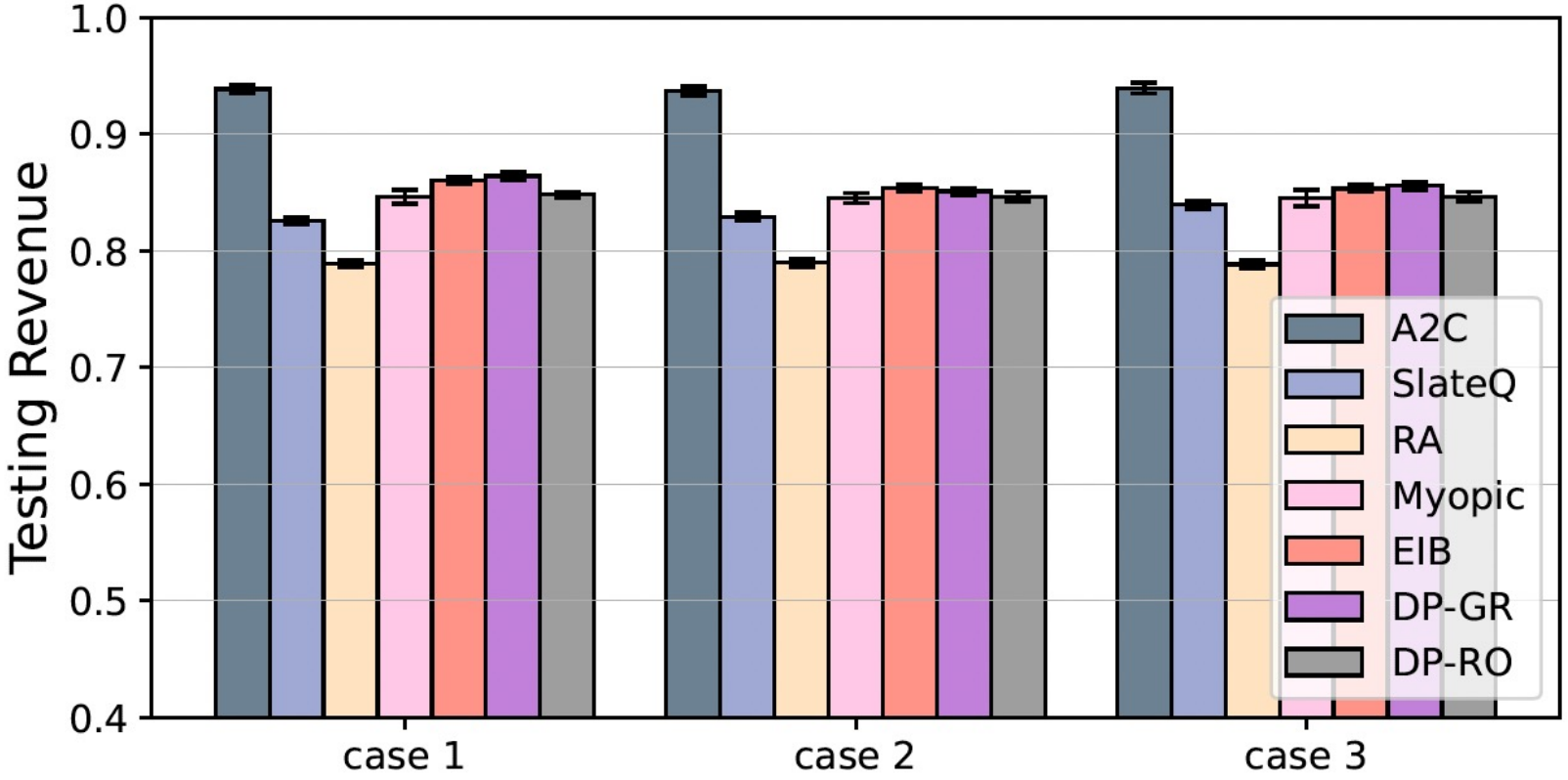


Numerical Results III

- Usage time distributions: negative binomial
  1. (m, i): product i bought by customer type m
  2. Case 1- Duration of (m, i): $\text{NegBin}\left(2, \eta_i\right), i \in \mathcal{N}$
  3. Case 2- Duration of (m, i): $\text{NegBin}\left(2, \eta_m\right), m \in \mathcal{M}$
  4. Case 3- Duration of (m, i): $\text{NegBin}\left(2, \eta\right)$



Numerical Results III



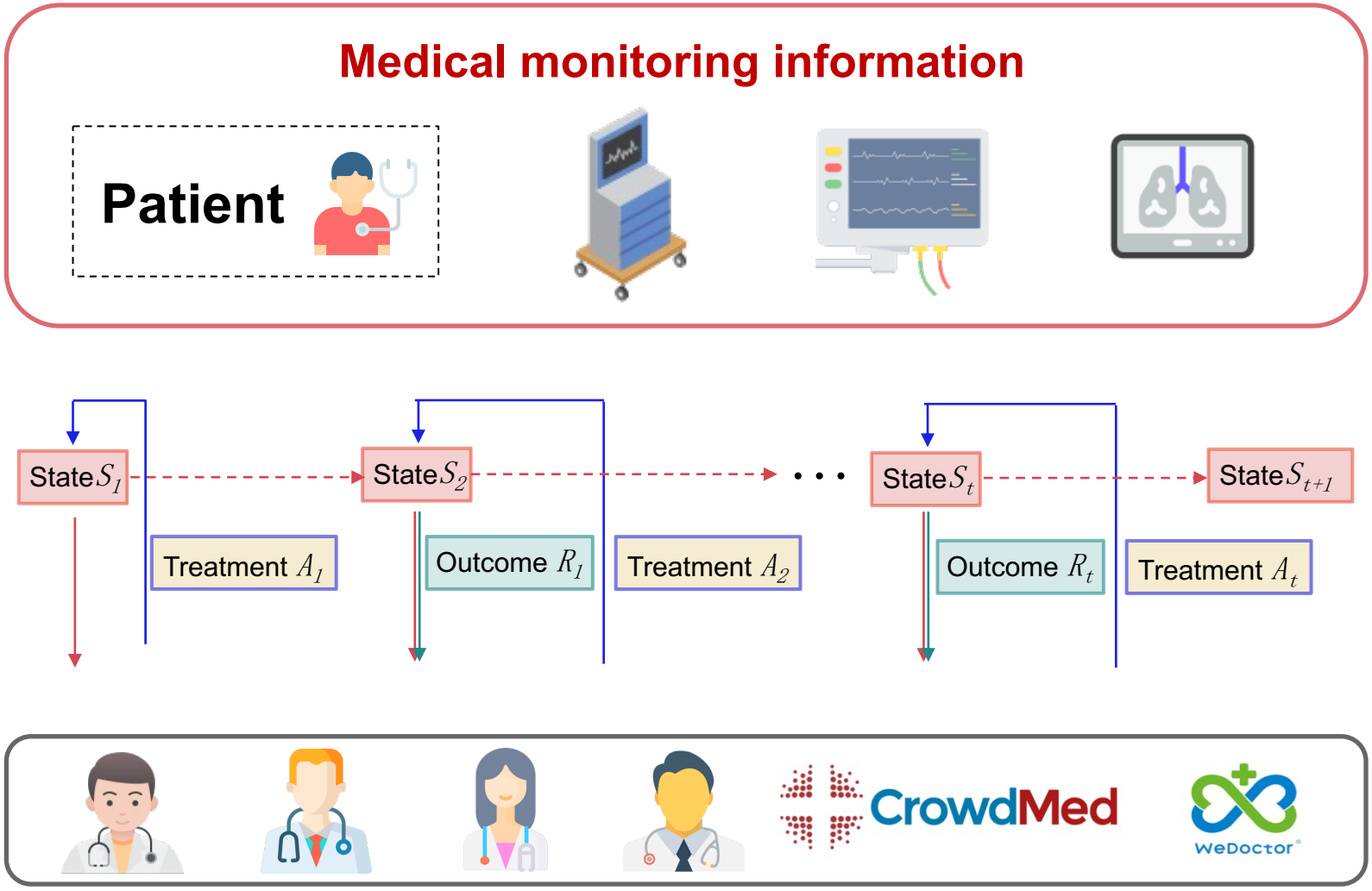
https://github.com/DRL-OM/DRL-assortment

Outline

- Introduction
- Generalized Tensor Contextual Bandits
- Deep RL for Online Assortment Customization
- **Distributed Q-Learning for DTRs**

Wang et al., IJOC, major revision

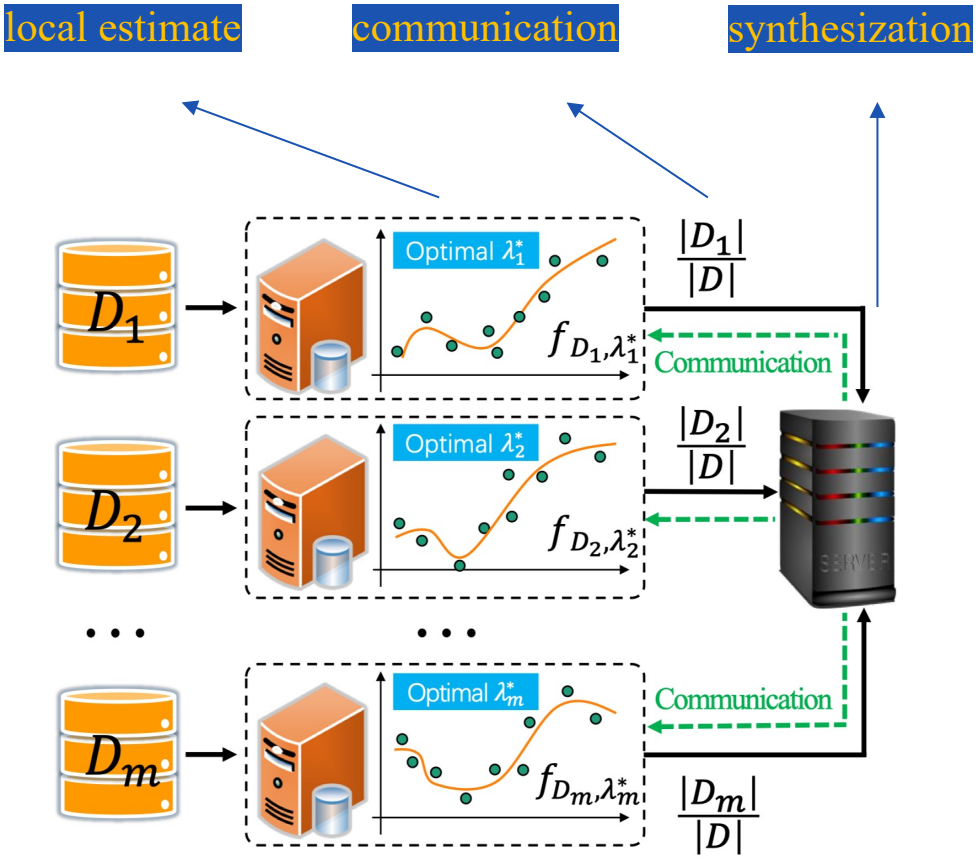
Dynamic Treatment Regimes (DTRs)



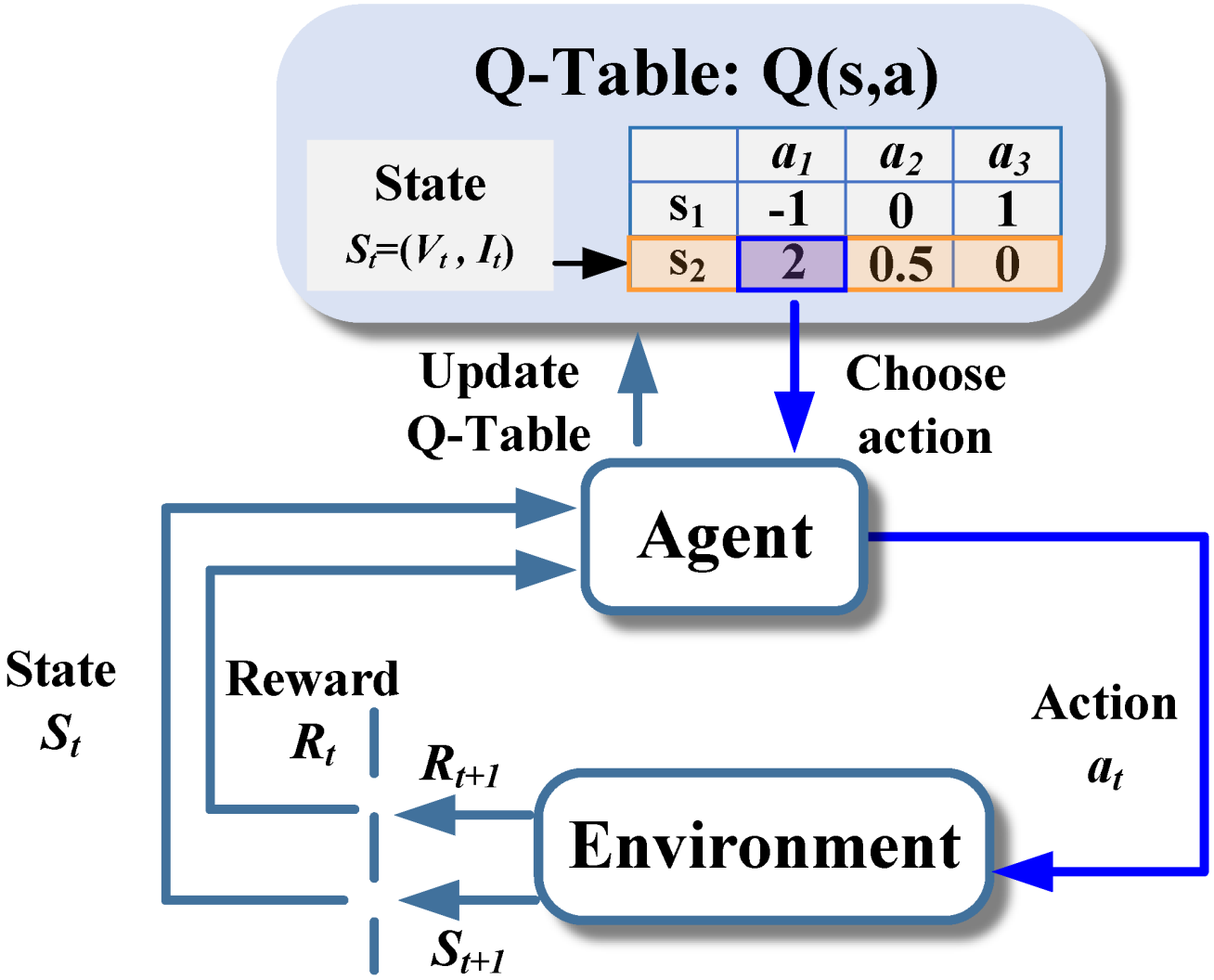
Distributed Learning

- Data silo

- Distributed learning based on a divide-and-conquer strategy



Q-Learning


Q-Table: Q(s,a)
State
$S_t=(V_t, I_t)$
a_1 a_2 a_3
s_1 -1 0 1
s_2 2 0.5 0
Update Q-Table
Choose action
Agent
State S_t
Reward R_t
R_{t+1}
Action a_t
Environment
S_{t+1}

Least Squares Formulation for Q-Learning

For any policy, $V^*(s_1) - V_\pi(s_1) = E_\pi\left[\sum_{t=1}^{T} V_t^*(s_t, a_{t-1}) - Q_t^*(s_t, a_t) \mid S_1 = s_1\right]$

advantage of time

Find good Q-function

$$Q_t^*(s_t, a_t) = E\left[R_t + V_{t+1}^* \mid S_t = s_t, A_t = a_t\right] \quad + \quad V_t^*(s_t, a_{t-1}) = \max_{a_t} Q_t^*(s_t, a_t)$$

Bellman equation

$$Q_t^*(s_t, a_t) = E\left[R_t + \max_{a_{t+1}} Q_{t+1}^*(s_{t+1}, a_t, a_{t+1}) \mid S_t = s_t, A_t = a_t\right]$$

For each *t*, a standard least-square regression problem

$f_\rho(x) = E[y \mid x]$	y	x
$\updownarrow$	$\updownarrow$	$\updownarrow$
$Q_t^*(s_t, a_t)$	$R_t + \max_{a_{t+1}} Q_{t+1}^*(s_{t+1}, a_t, a_{t+1})$	(s_t, a_t)

$$Q_t^*(s_t, a_t) = \arg\min_{Q_t} E\left[\left(R_t + \max_{a_{t+1}} Q_{t+1}^*(s_{t+1}, a_t, a_{t+1}) - Q_t(s_t, a_t)\right)^2\right]$$

Kernel-based Q-Learning

Kernel Ridge Regression (KRR) to solve $f_\rho(x) = E[y \mid x]$

- Formulation:

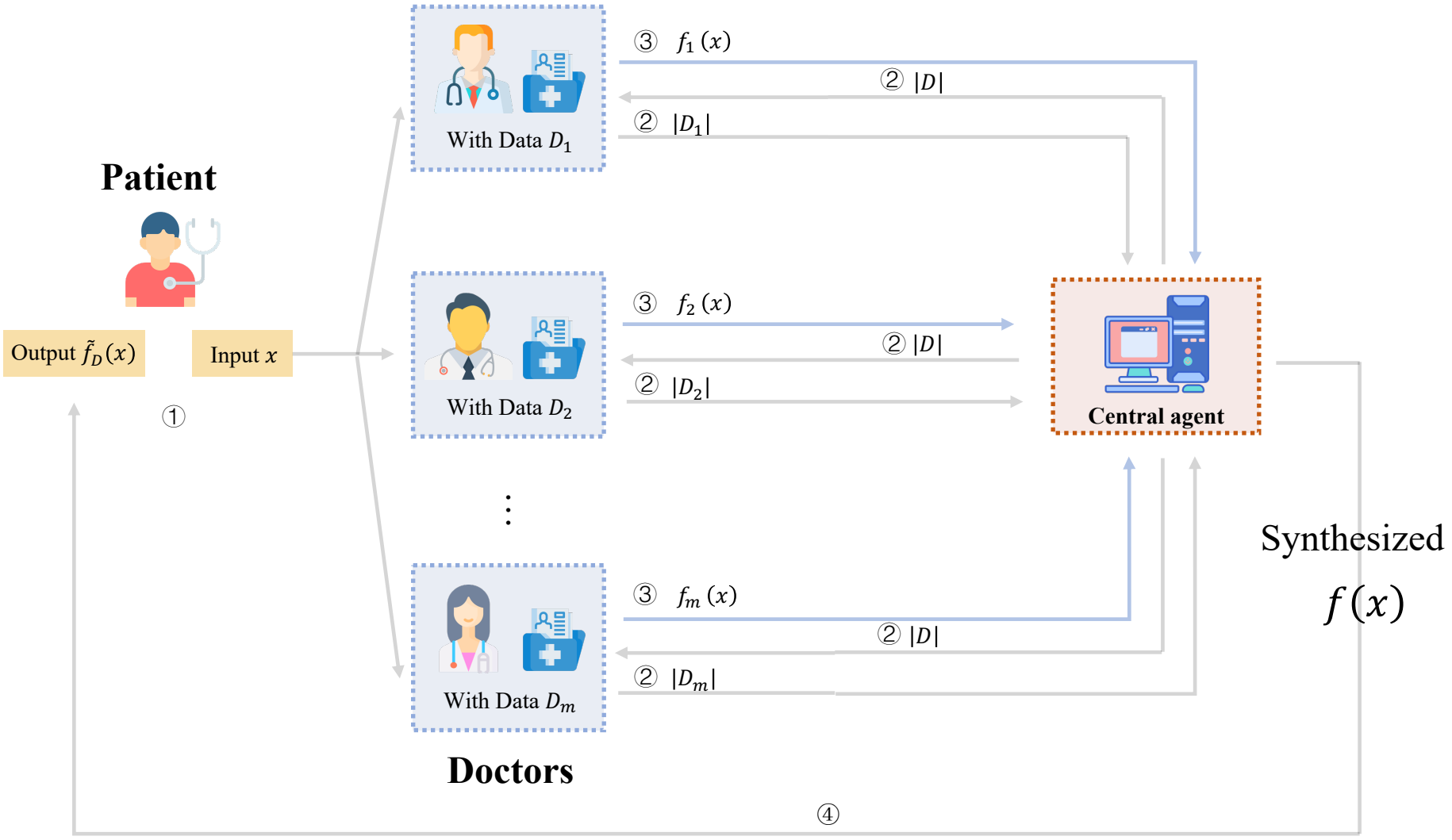
$$f_{D,\lambda} = \arg\min_{f \in H_K} \frac{1}{m} \sum_{i=1}^{m} (f(x_i) - y_i)^2 + \lambda \|f\|_K^2$$

- Advantages: Stable, highly accurate, interpretable

The reasons for using kernel-based Q-learning:

- Strong generalization guarantees
- Various scalable kernel method variants provide guidance
- Better generalization than linear methods and lower computational cost than deep Q-learning

Distributed Q-Learning: DKRR-DTR



Theoretical Results

(Theorem.) Under mild assumption, with $\frac{1}{2} \leq r \leq 1$ and $0 < s \leq 1$, if $\lambda_1 = \cdots = \lambda_T = |D|^{-\frac{1}{2r+s}}$, $|D_1| = \cdots = |D_m|$, and

the number of data subsets

the total number of samples

$$m(\log m + 1) \leq \frac{|D|^{\frac{2r+s-1}{4r+2s}}}{\log |D|}$$

then

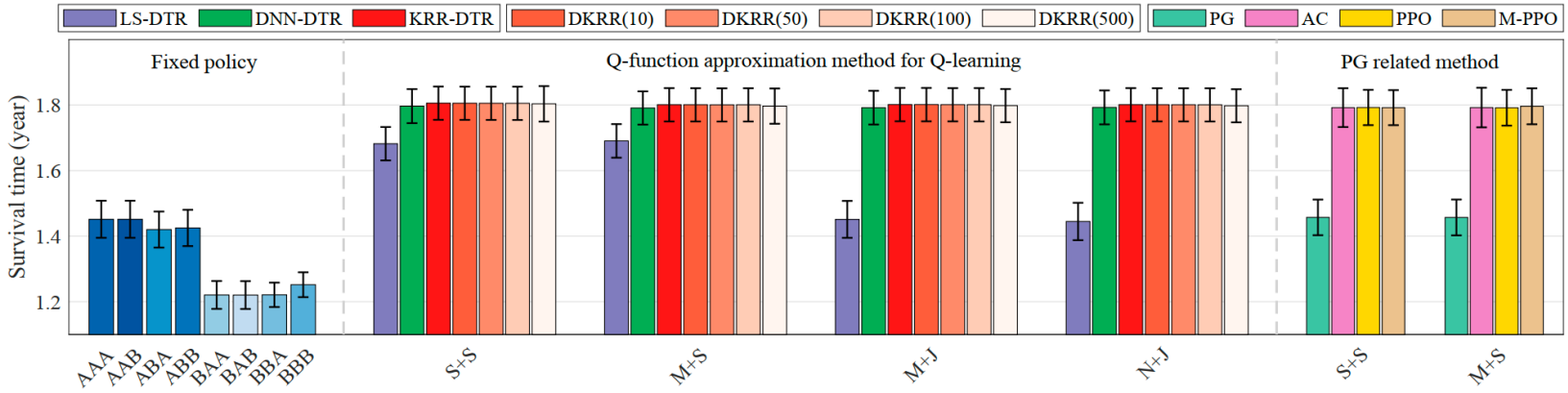
prediction accuracy

$$E\left[V_1^*(S_1) - V_{\bar{\pi}_{D,\vec{\lambda}},1}(S_1)\right] \leq C_2(T,\mu)|D|^{-\frac{r}{2r+s}}$$

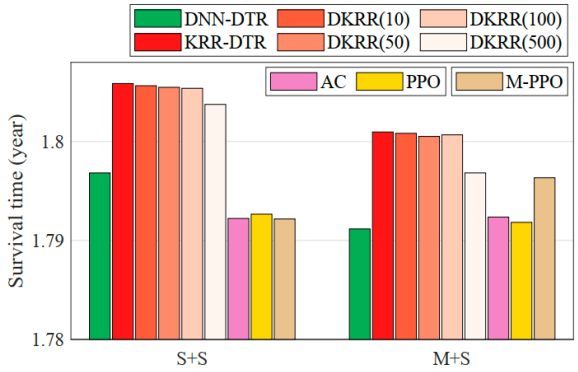
where $C_2(T,\mu) := C_2(2\mu\hat{C})^T T \sum_{t=1}^{T} 2^{-t} \sum_{\ell=t}^{T} \prod_{k=\ell}^{T-1} \left((T-k+2)\left(2\mu^{1/2}\right)^{k-\ell} + 1\right)$, and $\hat{C}$ and C_2 are constants depending only on M, C_0, κ, r, s, and $\max_{t=1,\ldots,T} \|h_t\|_{\mathcal{L}_t^2}$.

- The proposed DKRR-DTR possesses all the advantages of non-distributed learning version KRR-DTR.
- The computational cost is $\frac{1}{m^2}$ times smaller than that of KRR-DTR.

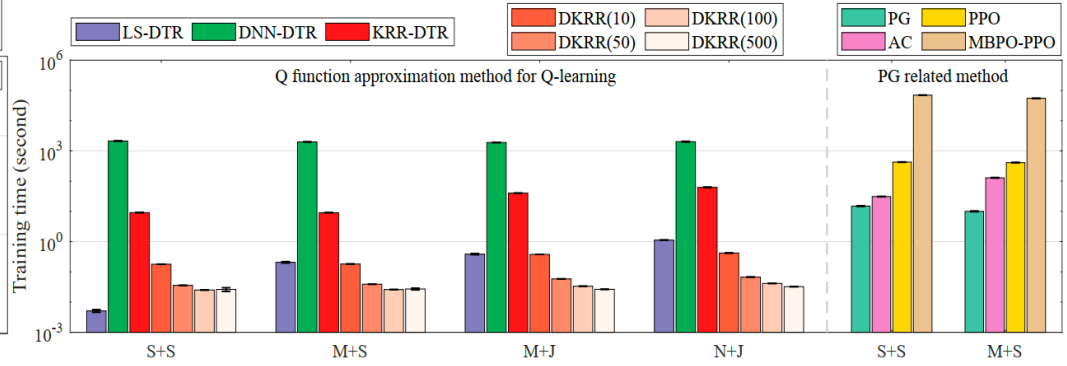
Clinical Trials for Cancer Treatment



(a) Comprehensive comparison of survival time



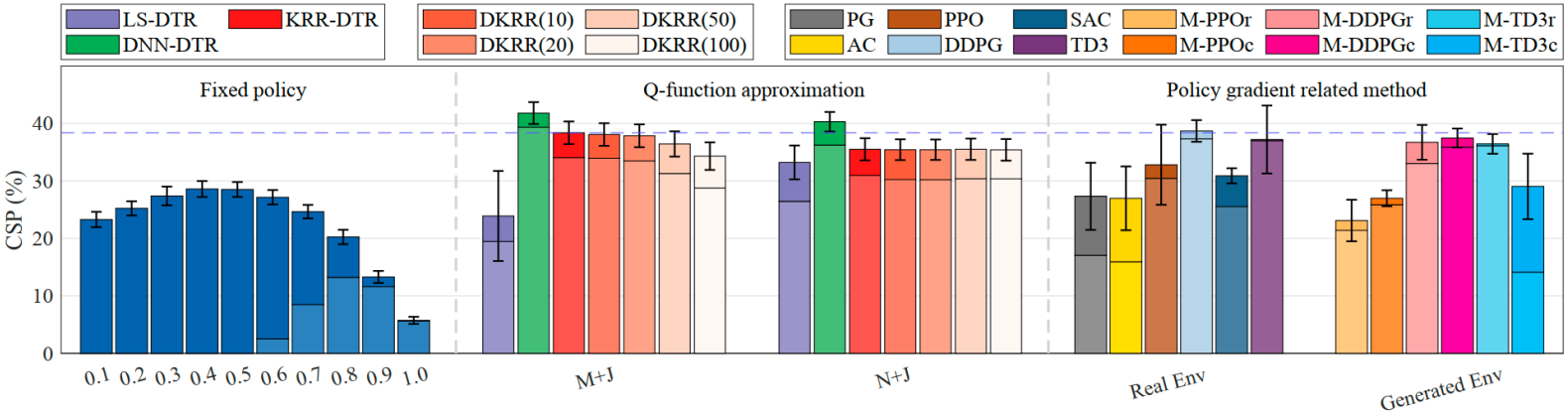
(b) Detailed comparison of survival time



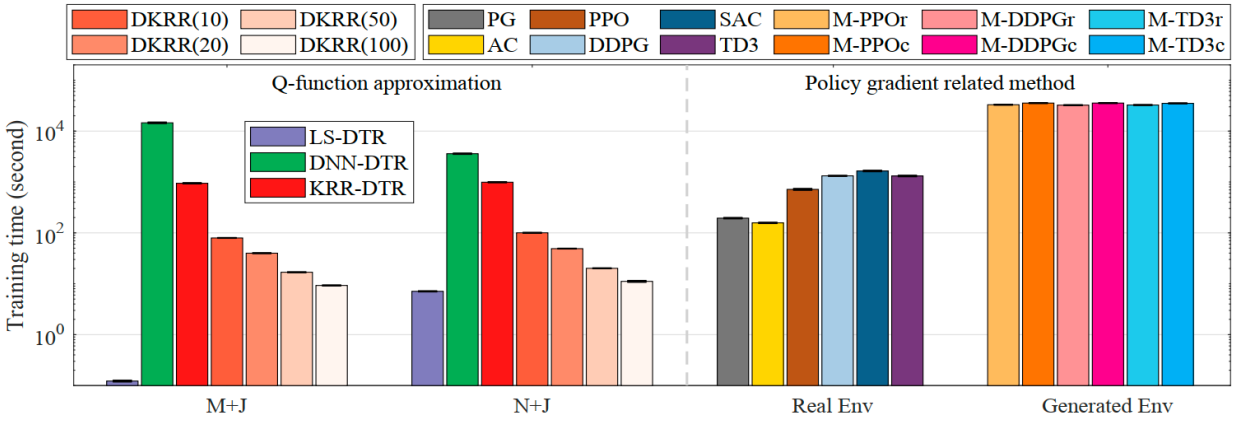
(c) Comprehensive comparison of training time

A small number of treatment options

Clinical Trials for Cancer Treatment



(a) Comprehensive comparison of CSP

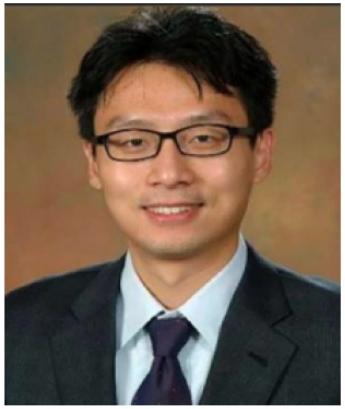


(b) Comprehensive comparison of training time

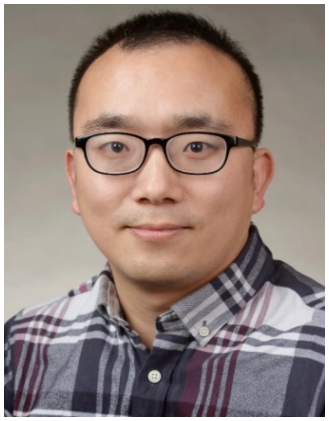
A large number of treatment options

Main Collaborators

Ningyuan Chen
University at Toronto



Shaojie Tang
University at Buffalo



Shao-Bo Lin
Xi'an Jiaotong University

Chenhao Wang
XJTU&Tongji

Tao Li
XJTU&HKUST

Qianxin Yi
XJTU&CUHK

Yiyang Yang
XJTU&CUHK-Shenzhen

Thank you!

Q&A