DeepStock: Reinforcement Learning with Policy Regularizations for Inventory Management

Yaqi Xie

Joint with Alibaba.com 天猫 TMALL COM

Xinru Hao, Jiaxi Liu,

Will Ma (Columbia GSB),

Linwei Xin (Cornell ORIE),

Lei Cao, Yidong Zhang



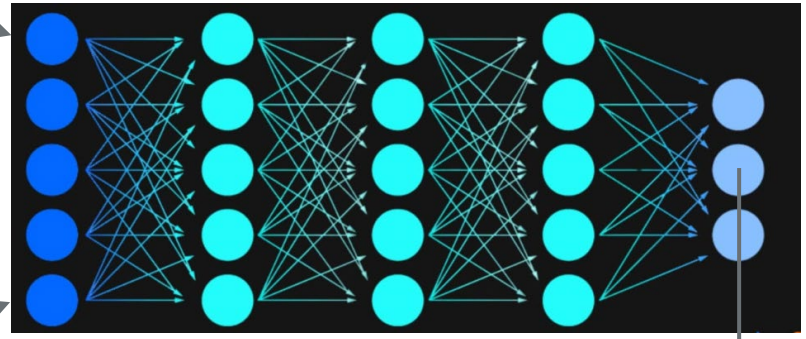



CHICAGO BOOTH

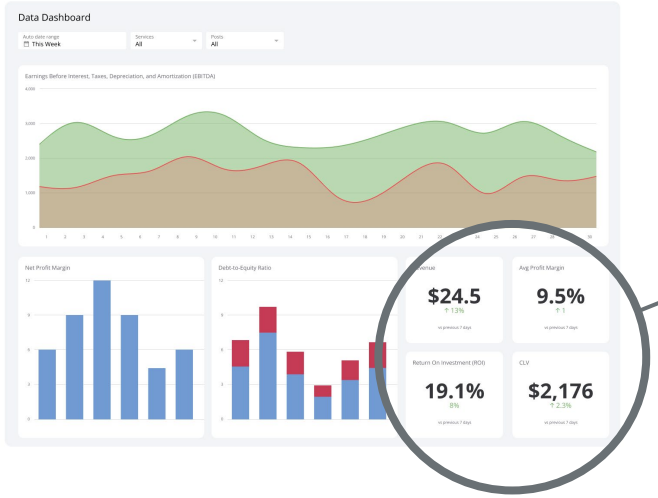
Inventory Management

Inventory state

Deep Neural Network Inventory Policy



Goal: order inventory to match upcoming demand (no more, no less)



Context vector

High-dimensional features about upcoming demand, 190-dimensional for Alibaba

Ordering decision

Model

- $t = 1, \ldots, T$: finite time horizon
- I^t: inventory at start of time t (Talk assumes 0 lead time)
- x^t: exogenous features of time t (from offline data)

} state $s^t = (I^t, x^t)$

- $a^t = \pi(I^t, x^t)$: action/order quantity at time t by policy π
- d^t: demand at time t
- $I^{t+1} = \max\{I^t + a^t - d^t, 0\}$: inventory for start of time $t + 1$ (lost-sales)

Decide π to minimize weighted loss of

- Stockout Rate: $\ell_{SR} = \frac{1}{T}\sum_{t=1}^{T} \mathbf{1}\{d^t \geq I^t + a^t\}$
- Turnover Time: $\ell_{TT} = \frac{\sum_{t=1}^{T} \max\{I^t + a^t - d^t, 0\}}{\sum_{t=1}^{T} \min\{d^t, I^t + a^t\}}$

} Practical performance metrics

Alibaba's setting

- Tmall: largest e-commerce platform in China
- Sells 100,000+ SKU's B2C across ≈ 20 warehouses
- Inventory for each of the **1,000,000+** SKU-warehouse pairs
  - Manage inventories separately; no multi-echelon network
  - Take weighted average across all pairs

Historical trajectories for SKU-warehouse's

SKU #	Date	*Lead Time (days)*	*Review Period (days)*	demand	short term fcst	long term fcst	historical avg	feat1	feat2	feat3	…
0	1-Jan	*10*	*4*	19	9	12.18	4.14	1	1	0	…
0	2-Jan	*10*	*4*	7	6.79	7.3	8.29	6	4	10	…
0	3-Jan	*10*	*4*	20	10.98	12.84	2.71	1	1	1	…
0	4-Jan	*10*	*4*	**13**	12.89	13.58	2	5	2	1	…
…	…	…	…	…	…	…	…	…	…	…	…
1	1-Jan	*7*	*3*	5	2.78	4.06	2.29	0	15	0	…
	2-Jan										
…	…	…	…	…	…	…	…	…	…	…	…
2	1-Jan	…	…	…	…	…	…	…	…	…	…
…											
…											
…											

Contextual Information

Offline historical data assumes:

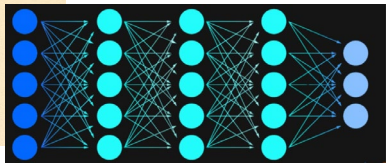
- **uncensored** demands (after extrapolation)
- **deterministic** lead times and review periods

Deep Reinforcement Learning (DRL)

SKU #	Date	*Lead Time (days)*	*Review Period (days)*	demand	short term fcst	long term fcst	historical avg	feat1	feat2	feat3
0	1-Jan	*10*	*4*	19	9	12.18	4.14	1	1	0
0	2-Jan	*10*	*4*	7	6.79	7.3	8.29	6	4	10
0	3-Jan	*10*	*4*	20	10.98	12.84	2.71	1	1	1
0	4-Jan	*10*	*4*	**13**	12.89	13.58	2	5	2	1
...	...	...	...	...	...	...	...	...	...	...
1	1-Jan	*7*	*3*	5	2.78	4.06	2.29	0	15	0
	2-Jan									
...		...	...	...	...	...	...	...	...	...
2	1-Jan	...	...	...	...	...	...	...	...	...
...										
...										
...										

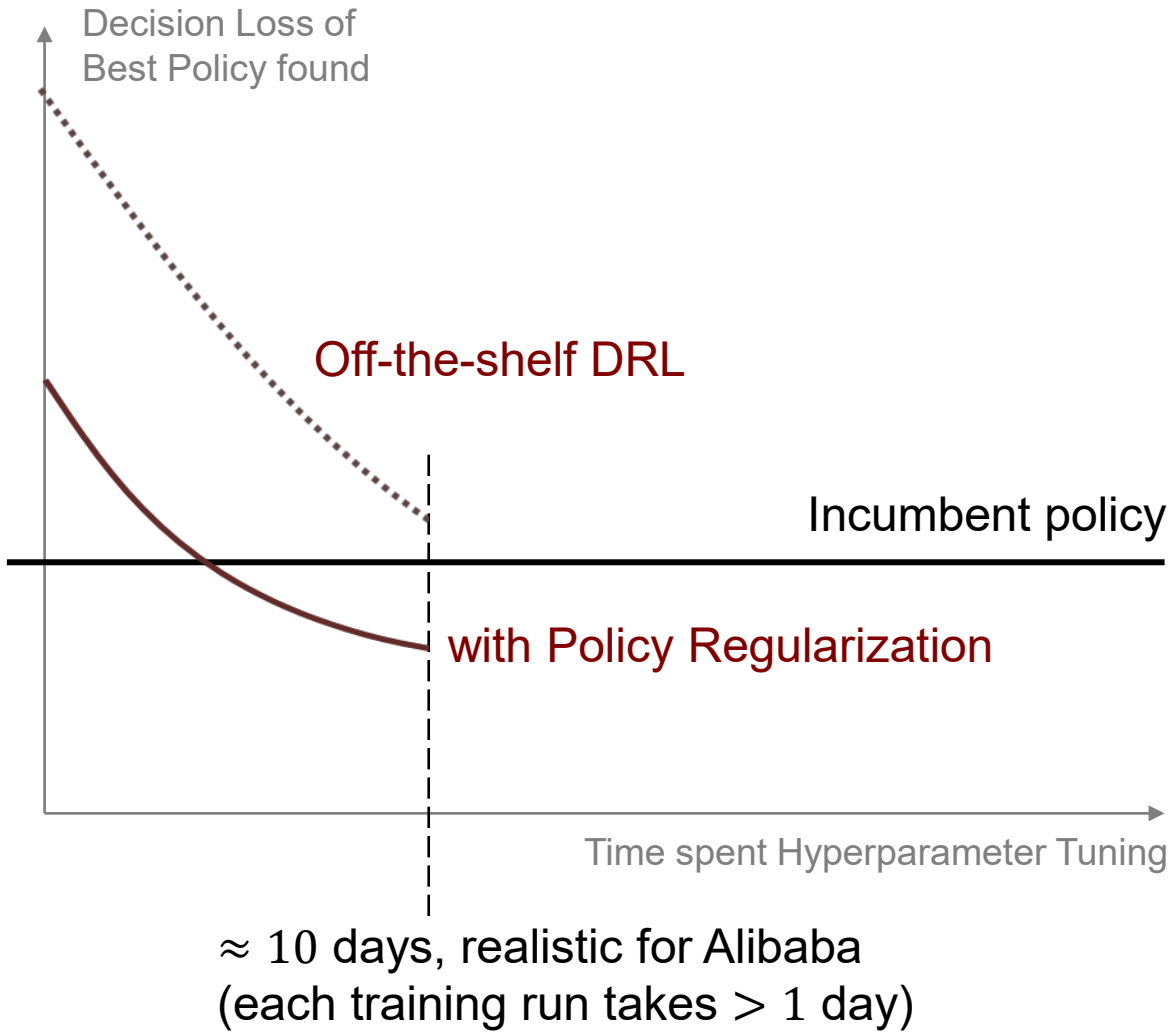
Offline historical data

Deep Neural Network Inventory Policy



Training Woes [Gijsbrechts Boute VanMieghem Zhang '25]

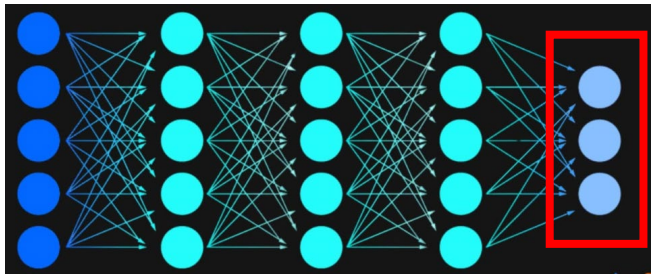
- Training takes a long time
- Need to try many hyperparameter configurations before training works



≈ 10 days, realistic for Alibaba
(each training run takes > 1 day)

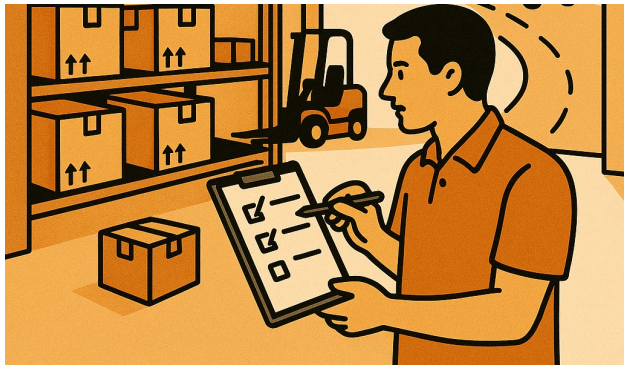
Inventory-specific Policy Regularizations

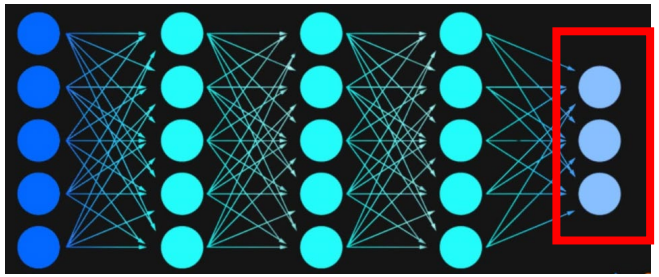
Deep Neural Network
$\mu(I^t, x^t)$



- "None": $\mu(I^t, x^t)$ = Order quantity
- "Base": $\mu(I^t, x^t)$ = Target inventory level
- "Coeff": $\mu(I^t, x^t)$ = Coefficients of linear relation w.r.t. selected features

"Base" Regularization





"Base Stock" = Target Inventory level

Simple Example:

- Demand each period is always 10 (no contexts)

"None":

Order quantity $\pi(I^t, x^t)$

$= \mu(I^t, x^t)$

I^t	Desired $\mu(I^t)$
11	0
10	0
9	1
8	2
…	…

"Base" Regularization:

Order quantity $\pi(I^t, x^t)$

$= \max\{ \mu(I^t, x^t) - I^t, 0 \}$

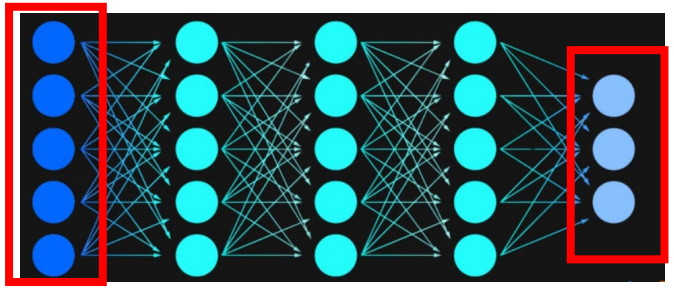
I^t	Desired $\mu(I^t)$
11	10
10	10
9	10
8	10
…	…

10 is called the "Base Stock"

“Coeff” Regularization

SKU #	Date	*Lead Time (days)*	*Review Period (days)*	demand	short term fcst	long term fcst	historical avg	feat1	feat2	feat3	…
0	1-Jan	*10*	*4*	19	9	12.18	4.14	1	1	0	…
0	2-Jan	*10*	*4*	7	6.79	7.3	8.29	6	4	10	…
0	3-Jan	*10*	*4*	20	10.98	12.84	2.71	1	1	1	…
0	4-Jan	*10*	*4*	13	12.89	13.58	2	5	2	1	…
	…	…	…	…	…	…	…	…	…	…	…
1	1-Jan	*7*	*3*	5	2.78	4.06	2.29	0	15	0	…
	2-Jan										
…		…	…	…	…	…	…	…	…	…	…
2	1-Jan	…	…	…	…	…	…	…	…	…	…
…											
…											
…											

State normalized across time



Coefficients

Example:

- Context vector includes recent demand and demand forecasts

“Coeff” Regularization:

Order quantity $\pi(I^t, x^t)$

$= \mu(I^t, x^t)_1 \times$ recent demand

$+ \mu(I^t, x^t)_2 \times$ historical demand

$+ \ldots$

Recent Demand	Desired Output 1
1000	1.1
1100	1.0
…	…

“Both” Regularization:

Order quantity $\pi(I^t, x^t)$

$= \max\{ \mu(I^t, x^t)_1 \times$ recent demand

$+ \mu(I^t, x^t)_2 \times$ historical demand

$+ \ldots - I^t, 0 \}$

DRL Methods: Model-Free Actor-Critic

Deep Deterministic Policy Gradient (DDPG) [Lillicrap et al. 15]

- Critic-dominated: Q-function

$$L(\phi, \mathcal{D}) = \underset{(s,a,r,s',d)\sim\mathcal{D}}{\mathrm{E}}\left[\left(Q_\phi(s,a) - \left(r + \gamma(1-d)\max_{a'} Q_\phi(s',a')\right)\right)^2\right]$$

- A deterministic policy optimized by taking gradients of Q in a continuous action space

Proximal Policy Optimization (PPO) [Schulman et al. 17]

- Actor-dominated:

$$L(s,a,\theta_k,\theta) = \min\left(\frac{\pi_\theta(a|s)}{\pi_{\theta_k}(a|s)}A^{\pi_{\theta_k}}(s,a),\ \ \mathrm{clip}\left(\frac{\pi_\theta(a|s)}{\pi_{\theta_k}(a|s)}, 1-\epsilon, 1+\epsilon\right)A^{\pi_{\theta_k}}(s,a)\right)$$

- Critic used for variance reduction
- A stochastic policy optimized by adjusting the probability distribution over actions

Differentiable Simulator (DS) [Madeka et al. 22, Alvo et al. 23]

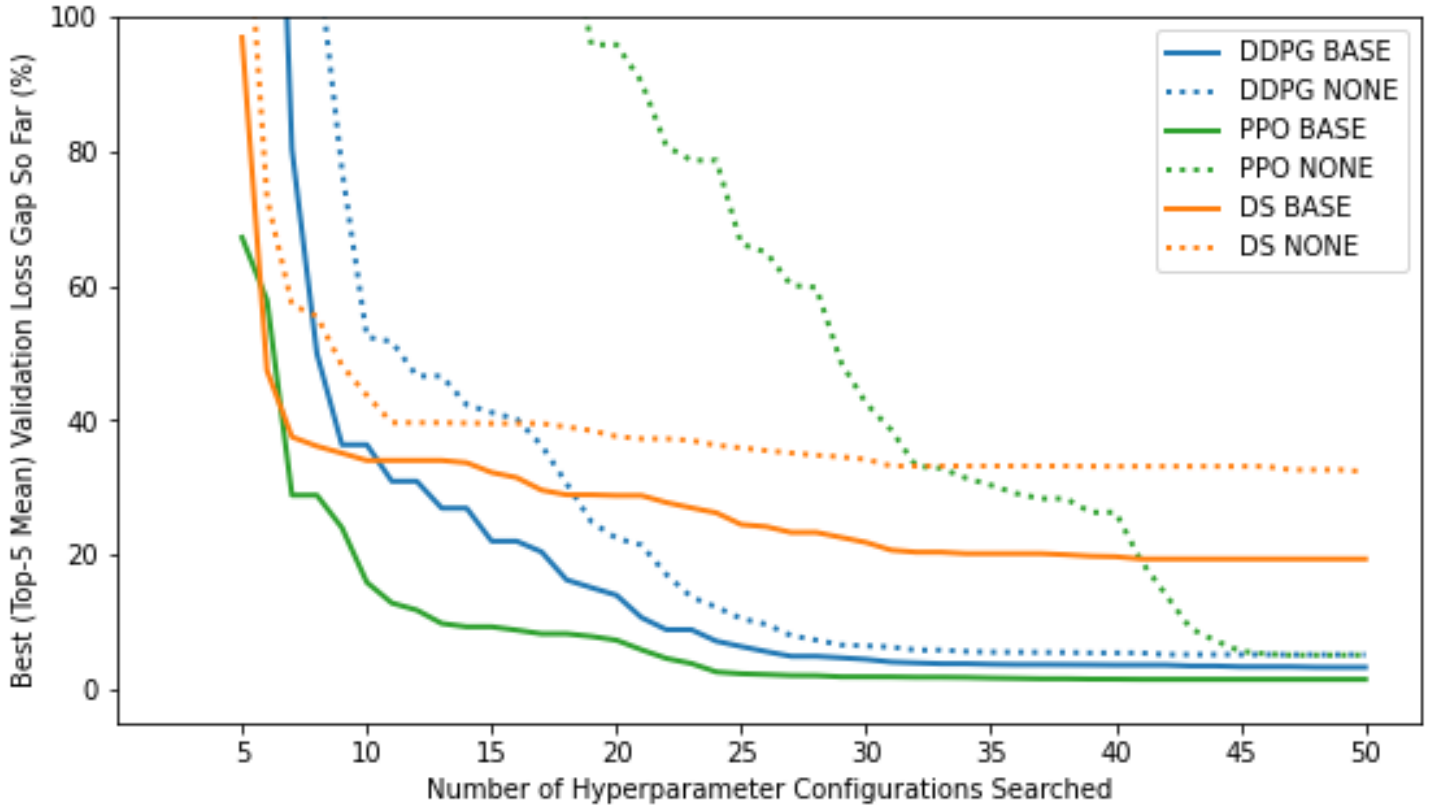
amazon

- No transition tuple; Not sensitive to hyperparameters

$$\min_\theta \mathbb{E}_\xi\left[\frac{1}{T}\sum_{t\in[T]} c(S_t, \pi_\theta(S_t), \xi_t)\middle| S_1\right] \quad \text{subject to} \quad S_{t+1} = f(S_t, \pi_\theta(S_t), \xi_t) \ \ \forall t \in [T],$$

Synthetic Data

- Independent and AR(1) demands over 29 days, with 1-dimensional context
- Train/Validate on 20 trajectories



Results:

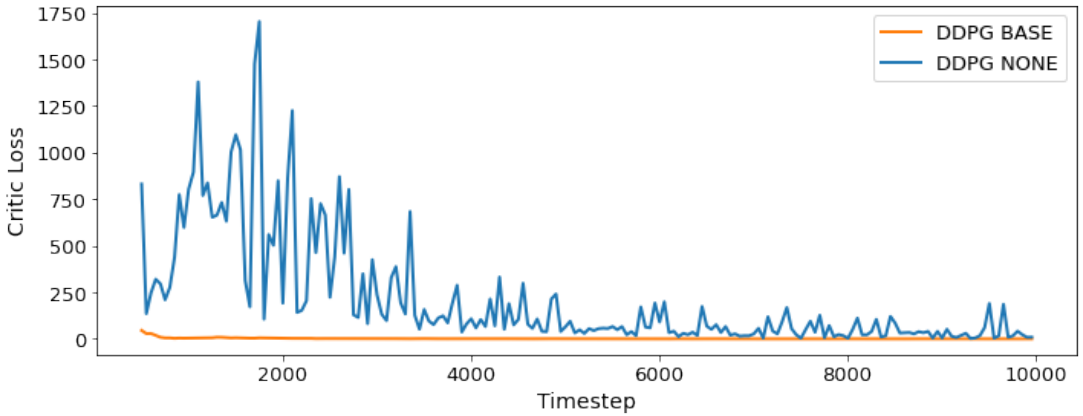
1. Policy Regularizations improve the performance of different DRL methods
2. Improve DS by the least, and DS has the worst final performance
3. Policy Regularizations have greater impact under limited hyperparameter search

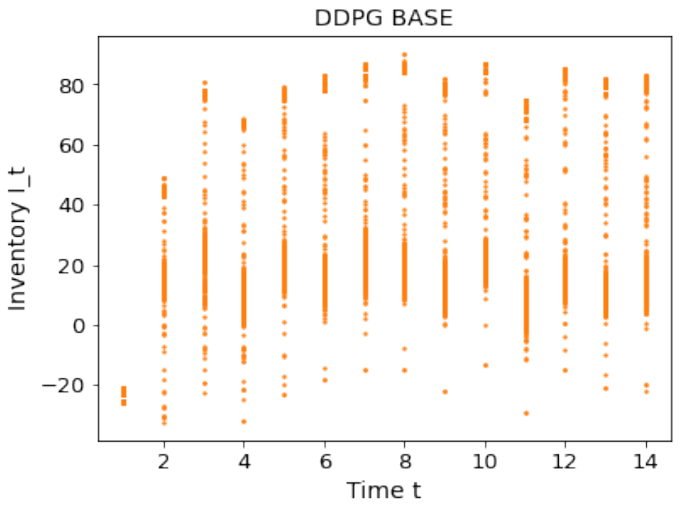
Why do Policy Regularizations help DRL training?

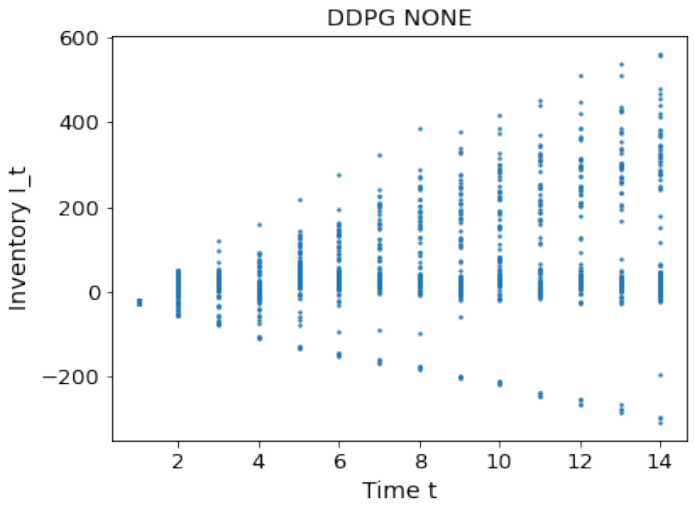
- Faster convergence of Q-function

$$\min_{Q(\cdot,\cdot)} \sum_{(s,a,r,s')} \left(Q(s,a) - r - \gamma \cdot \min_{a'} Q(s',a') \right)^2$$

Network output $\mu(s)$

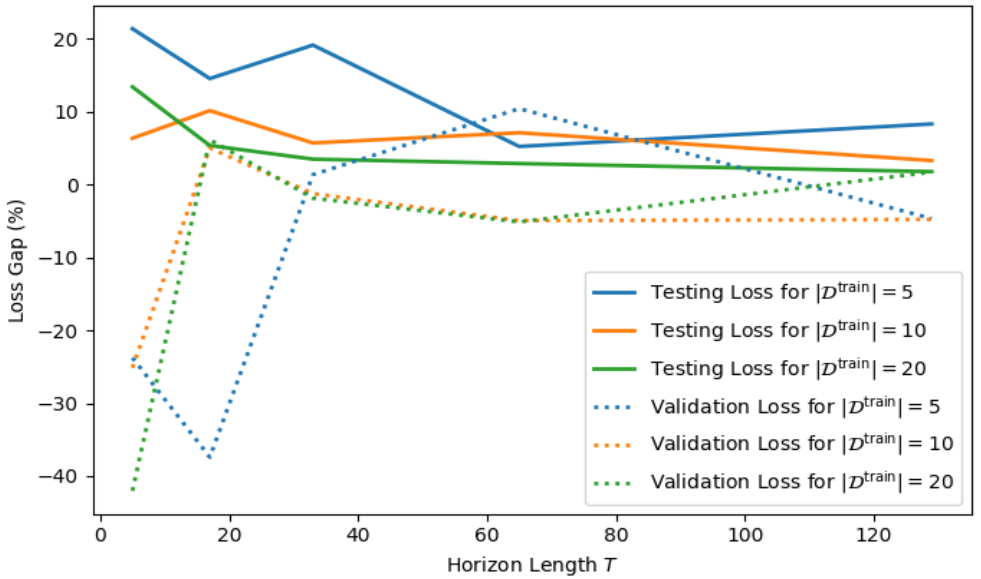






Why do Policy Regularizations help DS less & the final performance of DS worse?

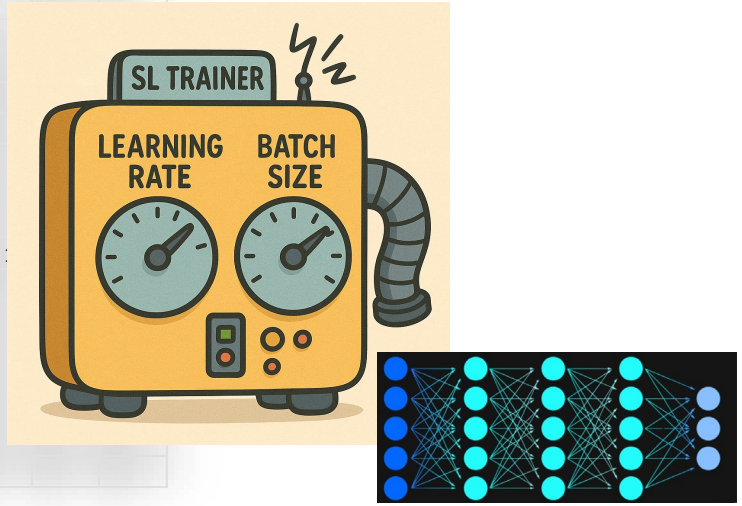
- DS does not use a Q-function or any intermediate $V(s)$
- Not cross-learning over time; overfitting to idiosyncrasies in trajectories



Alibaba's Offline Comparison

- Train/Validate on 55,000 SKU-warehouse combinations
- Test on chronologically-later days, different SKU-warehouse combinations

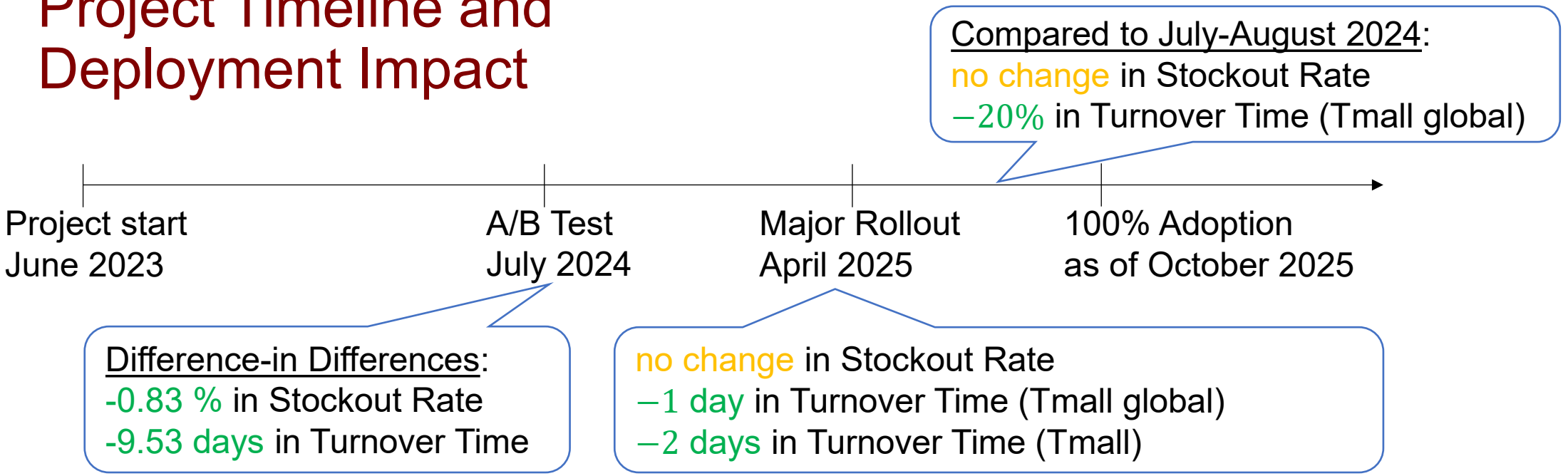
ys)	demand	short term fcst	long term fcst	historical avg	feat1	feat2	feat3	...
4	19	9	12.18	4.14	1			
4	7	6.79	7.3	8.29	6			
4	20	10.98	12.84	2.71	1			
4	**13**	12.89	13.58	2	5			
...	...	...	...	...	...	...		
3	5	2.78	4.06	2.29	0			
...	...	...	...	...	...	...		
...	...	...	...	...	...	...		



DRL methods	DDPG				DS			
Policy Regularization	None	Base	Coeff	Both	None	Base	Coeff	Both
Stockout Rate compared to "Both"	+10.10 %	+6.03 %	+4.41 %	-	+2.10 %	+2.18 %	+1.74 %	+1.91 %
Turnover Time compared to "Both"	+6.13 days	+6.46 days	-0.41 days	-	-1.25 Days	-2.81 days	+3.80 days	+0.23 days

(best results after hyperparameter searching for 10 days)

Project Timeline and Deployment Impact



Recognized by CTO, who highlights:

- Reliability when retraining every 1.5 months, on data from most recent 90 days
- Generality of **one** deployed policy for **all** 1,000,000+ inventories

Contributions of Paper [relative to past literature]

Simple Policy Regularizations and understanding **why** they work

Theory of learning with policy structure [Fan et al. 25, Xie et al. 25]

Encoding inventory policy structure into RL [De Moor et al. 22, Qi et al. 23, Maggiar et al. 25]

Reshape the narrative on best DRL method for Inventory

Papers advocating for SL method [Madeka et al. 22, Alvo et al. 23, Andaz et al. 24, Eisenach et al. 24, Maggiar et al. 25]

One policy to manage **all** 1,000,000+ inventories at Alibaba Tmall

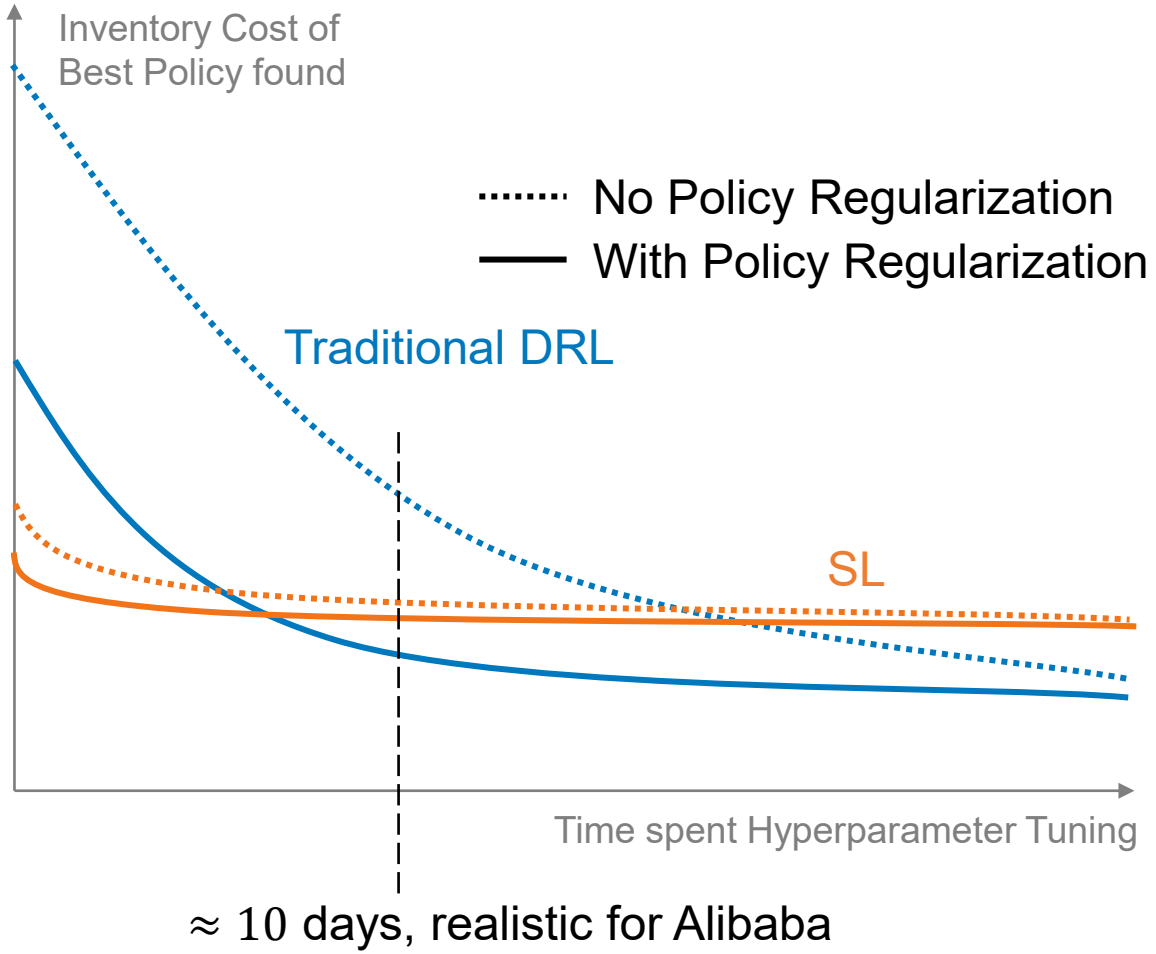
Other work on deployments [Liu et al. 22, Madeka et al. 22, Qi et al. 23]

Key Takeaways

I. Policy Regularizations improve the performance of DRL, especially under limited hyperparameter search

II. Policy Regularizations improve DRL methods differentially

III. Policy Regularizations enable a full-scale deployment of DRL



Thanks for Your Attention!

DeepStock: Reinforcement Learning with Policy Regularizations for Inventory Management
https://papers.ssrn.com/sol3/papers.cfm?abstract_id=5784782

CHICAGO BOOTH