MOSDT: Self-Distillation-Based Decision Transformer for Multi-Agent Offline Safe Reinforcement Learning

Yuchen Xia[1], Yunjian Xu[1,*]

[1]The Chinese University of Hong Kong

ycxia@link.cuhk.edu.hk, yjxu@mae.cuhk.edu.hk

香港中文大學
The Chinese University of Hong Kong

NEURAL INFORMATION PROCESSING SYSTEMS

Introduction

Motivation

While the applications of offline RL leverages span diverse domains, multi-agent offline safe RL (MOSRL), offering significant potential for distributed safety-critical applications, remains largely unstudied. For this reason, We introduce the first algorithm designed for MOSRL, MOSDT, alongside the first dataset and benchmark for this domain, MOSDB.

Novelties of MOSDT

- Different from most existing knowledge distillation-based multi-agent RL methods, we propose policy self-distillation (PSD) with a new global information reconstruction scheme by fusing the observation features of all agents, streamlining training and improving parameter efficiency.
- We adopt full parameter sharing across agents, significantly slashing parameter count and boosting returns up to 38.4-fold by stabilizing training.
- We propose a new plug-and-play cost binary embedding (CBE) module, which encodes cumulative costs as safety binary signals and embeds the signals into return features for efficient information aggregation.

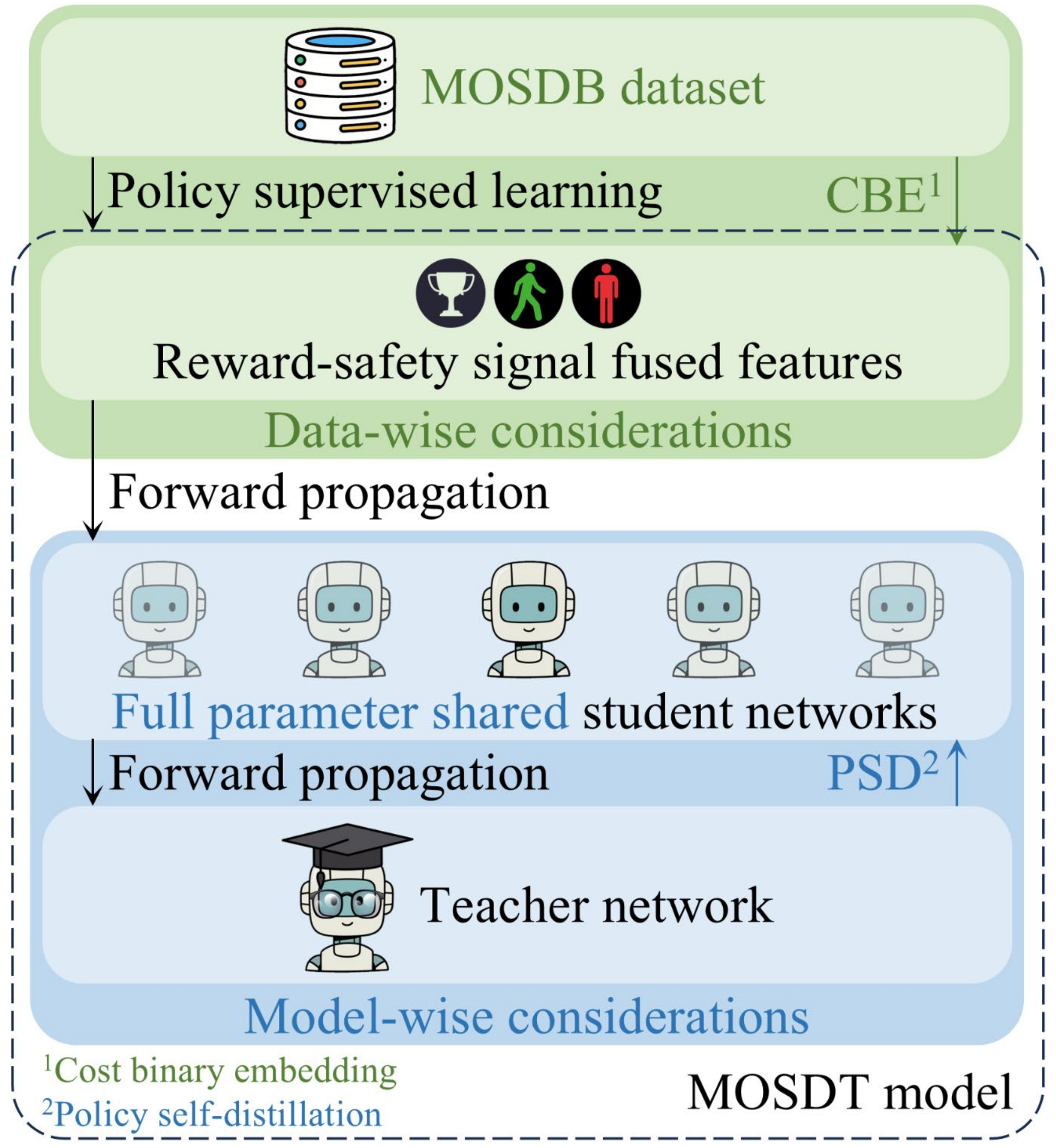
Experimental Results

On the strong MOSDB benchmark, MOSDT achieves state-of-the-art (SOTA) returns in 14 out of 18 tasks (across all base environments including MuJoCo, Safety Gym, and Isaac Gym) while ensuring complete safety, with only 65% of the execution parameter count of a SOTA single-agent offline safe RL method, CDT.

Paper

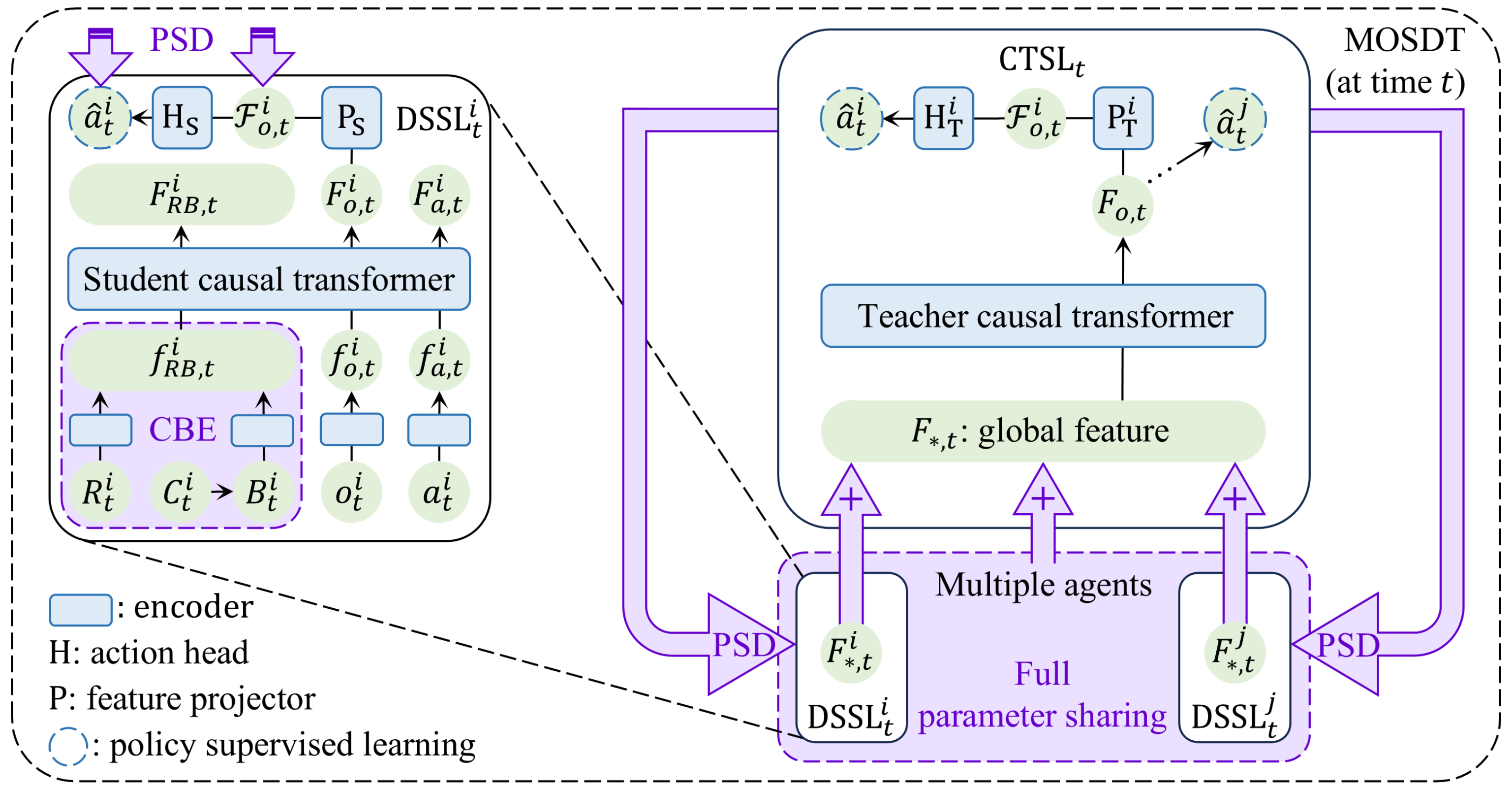
Code

Dataset and Results


MOSDB dataset
Policy supervised learning
CBE[1]
Reward-safety signal fused features
Data-wise considerations
Forward propagation
Full parameter shared student networks
Forward propagation
PSD[2]
Teacher network
Model-wise considerations
[1]Cost binary embedding
[2]Policy self-distillation
MOSDT model

Training MOSDT on the MOSDB dataset. The arrows represent the sequence of operations and data flow among the dataset, the student networks, and the teacher network under the proposed PSD framework.

Methods



The network structure of the proposed MOSDT algorithm. The purple areas indicate our proposed innovative modules. The green areas represent data, while the blue areas represent networks. We enlarge one of the DSSL modules and display it in detail in the upper left corner. We only illustrate the situation at time *t*.

Decentralized student supervised learning (DSSL)

In DSSL, each agent's student network processing its local trajectory. A critical component is CBE, which transforms the scalar cumulative cost into a binary safety signal. This signal is then concatenated with the return feature, creating a fused representation that explicitly informs the model about the reward-cost correlation from the very start of the sequence.

Further enhancing its efficiency and stability is the principle of full parameter sharing, where all student networks are identical copies. This design drastically reduces the total parameter count, mitigates the training instability common in multi-agent systems due to non-stationarity, and improves scalability to a larger number of agents.

Centralized teacher supervised learning (CTSL)

The CTSL process is built around PSD. Unlike conventional two-stage distillation, PSD performs distillation concurrently with supervised learning. The local features extracted by each student network are summed across all agents to reconstruct a global feature set. This aggregated information is processed by a shared teacher network, which has access to global state information and generates refined action predictions. The PSD mechanism then aligns the decentralized student policies with this centralized teacher by mainly minimizing the divergence between their action distributions and the difference between their intermediate feature representations.

Centralized Training and Decentralized Execution (CTDE)

The entire system is trained end-to-end by minimizing a joint loss function that combines the supervised learning objectives for both the student and teacher networks with the distillation loss from PSD. During execution, only the lightweight, parameter-shared student networks are used, enabling fully decentralized and efficient policy deployment.

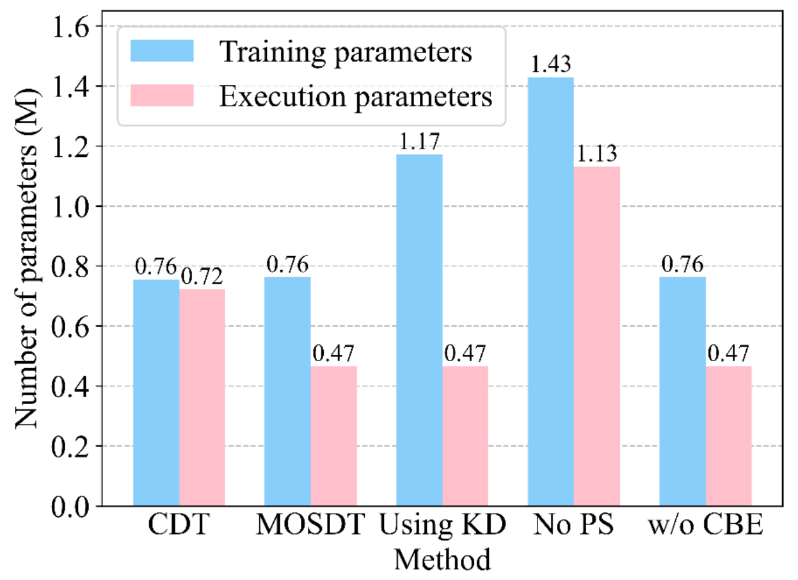
Experiments

MOSDB benchmark and the performance of MOSDT. Results are in the "return (cumulative cost)" format. The cost threshold 25. Blue: Safe policies with the highest rewards. Red: Unsafe policies.

Task	BC	BCQ-Lag	BEAR-Lag	CDT	COptiDICE	CPQ	MOSDT (ours)
MOS Velocity							
2x1Swimmer	5.08 (7.27)	8.13 (11.53)	5.24 (9.33)	9.25 (12.70)	0.97 (12.13)	8.20 (13.30)	11.64 (20.33)
2x3HalfCheetah	2043.94 (22.83)	2162.67 (62.43)	2181.00 (57.90)	2087.33 (40.03)	2029.00 (10.03)	406.90 (8.97)	2052.64 (22.27)
2x3Walker2d	1512.75 (0.00)	1578.59 (10.17)	1515.60 (3.53)	1526.43 (3.67)	1540.41 (0.00)	315.20 (15.97)	1584.87 (3.83)
2x4Ant	2361.63 (0.00)	2472.33 (6.77)	2217.89 (0.90)	2116.81 (7.77)	2125.14 (1.67)	-1488.45 (0.50)	2054.88 (0.87)
3x1Hopper	31.63 (0.30)	40.20 (1.83)	82.24 (8.33)	27.15 (0.00)	69.02 (7.40)	121.58 (1.07)	1122.23 (4.00)
4x2Ant	831.82 (0.00)	779.85 (0.00)	792.78 (0.00)	923.72 (0.00)	816.86 (0.00)	-466.82 (0.83)	2083.85 (3.53)
6x1HalfCheetah	447.13 (0.00)	339.63 (0.03)	334.62 (0.00)	397.80 (0.00)	321.46 (0.00)	-201.40 (0.87)	1853.64 (21.97)
9|8Humanoid	575.73 (20.70)	581.42 (19.27)	571.16 (18.57)	545.80 (20.87)	554.40 (15.27)	404.48 (18.53)	444.71 (22.80)
MOS Goal							
Multi-Ant1	23.41 (17.00)	15.78 (23.85)	21.31 (9.50)	28.94 (10.67)	33.75 (14.00)	0.61 (0.00)	38.38 (14.50)
Multi-Ant2	2.59 (8.00)	1.90 (21.05)	2.71 (16.17)	4.93 (17.50)	3.55 (17.33)	0.51 (0.00)	2.96 (7.50)
Multi-Point1	6.47 (17.17)	-1.43 (25.48)	4.28 (7.67)	9.25 (14.83)	2.11 (8.17)	2.84 (7.67)	9.65 (12.67)
Multi-Point2	-0.06 (4.33)	-8.59 (36.53)	1.20 (14.00)	3.07 (19.33)	-1.23 (18.83)	0.95 (9.67)	-1.08 (21.00)
MOS Isaac Gym							
CloseDrawerMA	-5.23 (1.00)	-5.15 (0.80)	-4.49 (3.57)	-3.45 (0.00)	-5.28 (0.00)	-5.43 (0.00)	-3.45 (0.00)
PickAndPlaceMA	-5.72 (3.73)	-4.92 (0.27)	-4.26 (0.00)	-5.50 (0.00)	-7.58 (2.67)	-5.32 (2.47)	-2.67 (0.00)
CatchFingerMA	0.19 (0.00)	0.20 (2.60)	0.23 (0.00)	0.18 (7.60)	0.22 (0.00)	0.11 (6.23)	0.25 (6.33)
CatchJointMA	0.21 (0.00)	0.19 (0.00)	0.20 (0.00)	0.26 (0.80)	0.24 (0.00)	0.15 (5.97)	0.31 (0.00)
OverFingerMA	0.43 (0.00)	0.46 (0.07)	0.44 (0.00)	0.52 (0.00)	0.44 (0.00)	0.39 (8.00)	0.52 (0.00)
OverJointMA	0.45 (0.00)	0.47 (0.00)	0.46 (0.00)	0.46 (5.63)	0.44 (0.00)	0.42 (8.00)	0.47 (1.03)
Summary	0 SOTA (safe)	3 SOTA (unsafe)	0 SOTA (unsafe)	4 SOTA (unsafe)	0 SOTA (safe)	0 SOTA (safe)	14 SOTA (safe)

Overall Performance

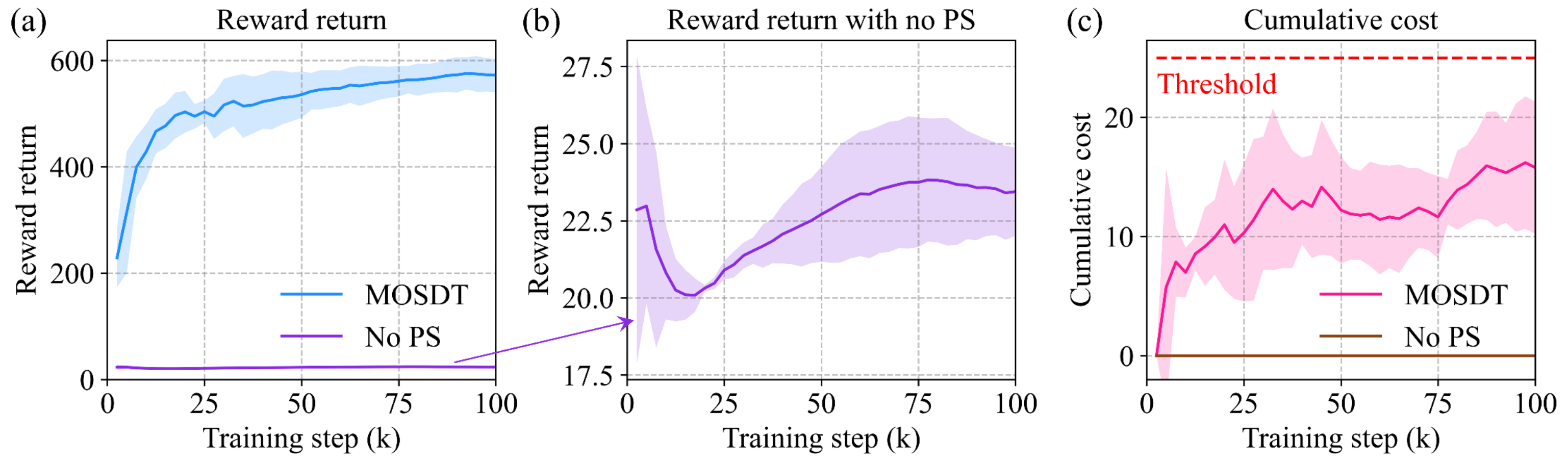
On the MOSDB benchmark, MOSDT achieves SOTA returns in 14 out of 18 tasks while guaranteeing complete safety. Despite MOSDT relying only on partial observations during execution, it significantly outperforms strong baselines—which are granted an advantage by operating in a centralized execution framework. MOSDT requires only 65% of the execution parameters of its base model.



Number of parameters. "Using KD": Using conventional KD instead of PSD. "No PS": No parameter sharing. MOSDT only requires 65% of the execution parameter count of CDT. PSD reduces the number of training/execution parameters by 47%/58%.

Ablation Studies

Ablation studies rigorously validate the contribution of each core component. The removal of PSD leads to unsafe policies and performance degradation in 12 tasks. Full parameter sharing is proven to be an important factor in training stability and performance scaling, especially in tasks with three or more agents, where it boosts returns by up to 38.4-fold compared to models without sharing. Furthermore, it reduces training and execution parameters by 47% and 58%, respectively. The CBE module provides the most widespread performance boost, enhancing returns in 14 out of 18 tasks.



Visualization of the training on the "3x1Hopper" task. "No PS": No parameter sharing. The purple curve in Subgraph (a) is enlarged and displayed in detail in Subgraph (b) for better viewing. Shaded areas represent sample standard deviations across multiple runs.