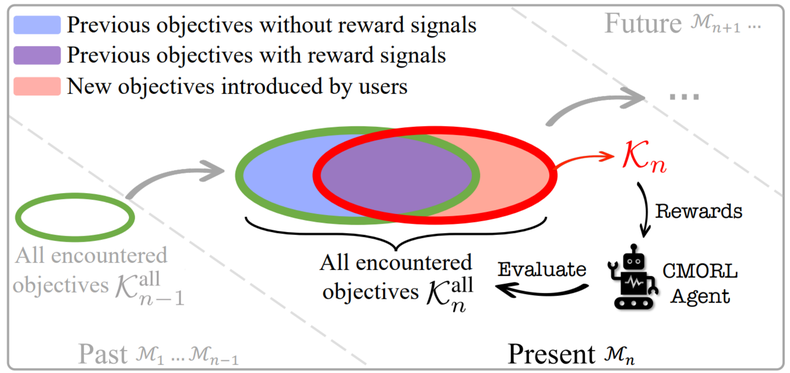
Continual Multi-Objective Reinforcement Learning via Reward Model Rehearsal

Lihe Li[1,2], Ruotong Chen[1,2], Ziqian Zhang[1,2], Zhichao Wu[1,2], Yi-Chen Li[1,2,3], Cong Guan[1,2], Yang Yu[1,2,3], Lei Yuan[1,2,3]

[1]National Key Laboratory for Novel Software Technology, Nanjing University; [2]School of Artificial Intelligence, Nanjing University; [3]Polixir Technologies

Introduction

MORL methods focus on learning to complete and trade-off a fixed set of objectives. But the objective set may change over time in some real-world scenarios.

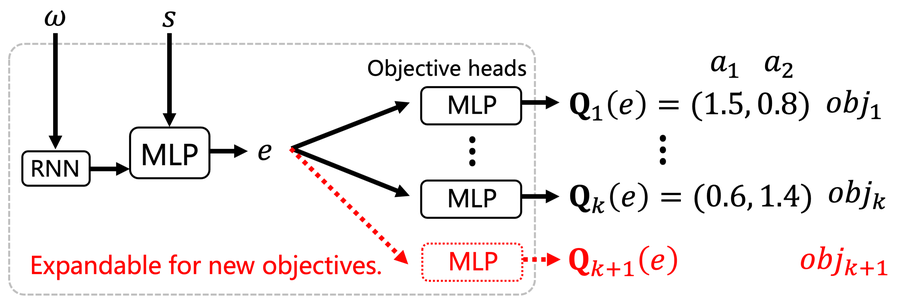


To bridge this gap, we formalize the continual MORL problem and try to solve it with our method ***Co**ntinual Multi-Objective **Re**inforcement Learning via **Re**ward Model **Re**hearsal (CORE3).*

Method

Dynamic Network for Evolving Objectives

- For new objectives, expand the agent network with multi-head architecture.
- No need to train the entire network from scratch.
- Results: ***Learn to complete new objectives faster.***



Agent Network

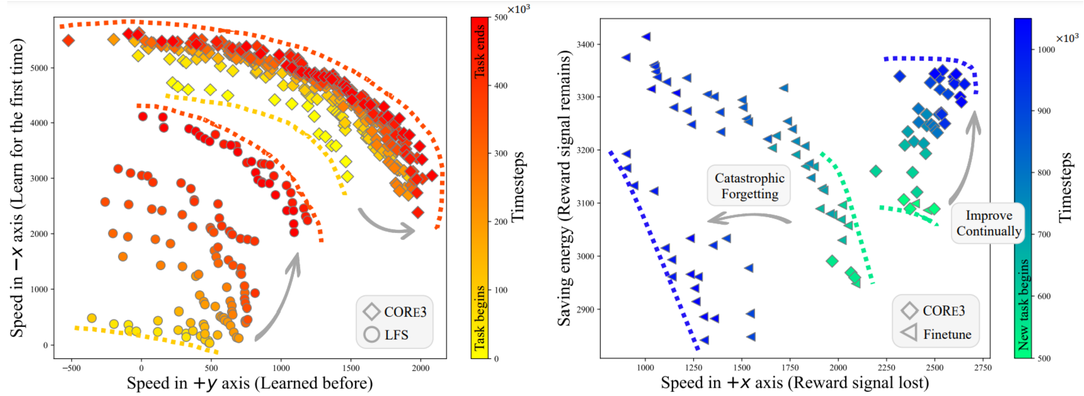
Multi-Objective Reward Model Rehearsal

- For old objectives lacking reward signals, we learn a multi-objective reward model $\widehat{\mathbf{R}}(s,a) = \left(\hat{R}_1(s,a), \dots, \hat{R}_M(s,a)\right)$ to recover these signals, and use them to train the agent, i.e., reward model rehearsal.
- Performance guarantee: For all preferences, performance loss $\leq \delta/(1-\gamma)$, where δ is the model prediction error.
- Results: ***Maintain the ability to complete old objectives lacking reward signals.***

Experiments

Envs	Metrics	Finetune	CLEAR	EWC	CORE3 (Oracle)	CORE3 (Ours)
FTN	HV (↑)	1.00 ± 0.07	0.85 ± 0.07	1.04 ± 0.07	1.10 ± 0.04	$\mathbf{1.06 \pm 0.00}$
	SP (↓)	1.00 ± 0.14	0.73 ± 0.02	$\mathbf{0.58 \pm 0.06}$	0.72 ± 0.01	0.82 ± 0.02
	BWT (↑)	-0.12 ± 0.02	0.00 ± 0.01	-0.14 ± 0.01	0.03 ± 0.00	$\mathbf{0.02 \pm 0.01}$
Grid	HV (↑)	1.00 ± 0.82	4.21 ± 0.66	2.26 ± 1.34	4.60 ± 0.26	$\mathbf{4.54 \pm 0.60}$
	SP (↓)	1.00 ± 0.58	$\mathbf{0.42 \pm 0.11}$	0.47 ± 0.10	0.37 ± 0.02	0.44 ± 0.10
	BWT (↑)	-0.44 ± 0.07	$\mathbf{-0.14 \pm 0.17}$	-0.58 ± 0.05	-0.07 ± 0.12	-0.17 ± 0.15
Ant	HV (↑)	1.00 ± 0.42	1.78 ± 0.37	1.03 ± 0.60	2.30 ± 0.18	$\mathbf{1.89 \pm 0.35}$
	SP (↓)	1.00 ± 0.56	$\mathbf{0.33 \pm 0.06}$	1.43 ± 1.78	1.06 ± 0.57	0.73 ± 0.72
	BWT (↑)	-0.27 ± 0.02	0.01 ± 0.18	-0.35 ± 0.1	0.34 ± 0.18	$\mathbf{0.11 \pm 0.16}$
Hopper	HV (↑)	1.00 ± 1.22	2.03 ± 3.49	1.01 ± 1.43	7.84 ± 0.29	$\mathbf{8.23 \pm 1.42}$
	SP (↓)	1.00 ± 0.76	7.59 ± 2.98	4.76 ± 3.54	0.35 ± 0.11	$\mathbf{0.21 \pm 0.10}$
	BWT (↑)	-0.40 ± 0.07	-0.38 ± 0.22	-0.37 ± 0.07	0.11 ± 0.09	$\mathbf{0.06 \pm 0.05}$
API (%) (↑)	HV	\	121.72	33.61	296.03	**293.18**
	SP	\	−126.81	−80.85	37.61	**45.37**
	BWT	\	69.09	−17.87	145.36	**109.09**

CORE3 learns to complete and trade-off all encountered objectives the best.



(a) CORE3 v.s. LFS on learning a new task.

(b) CORE3 v.s. Finetune on an ended task.

CORE3 learns new objectives faster and better remembers old objectives.