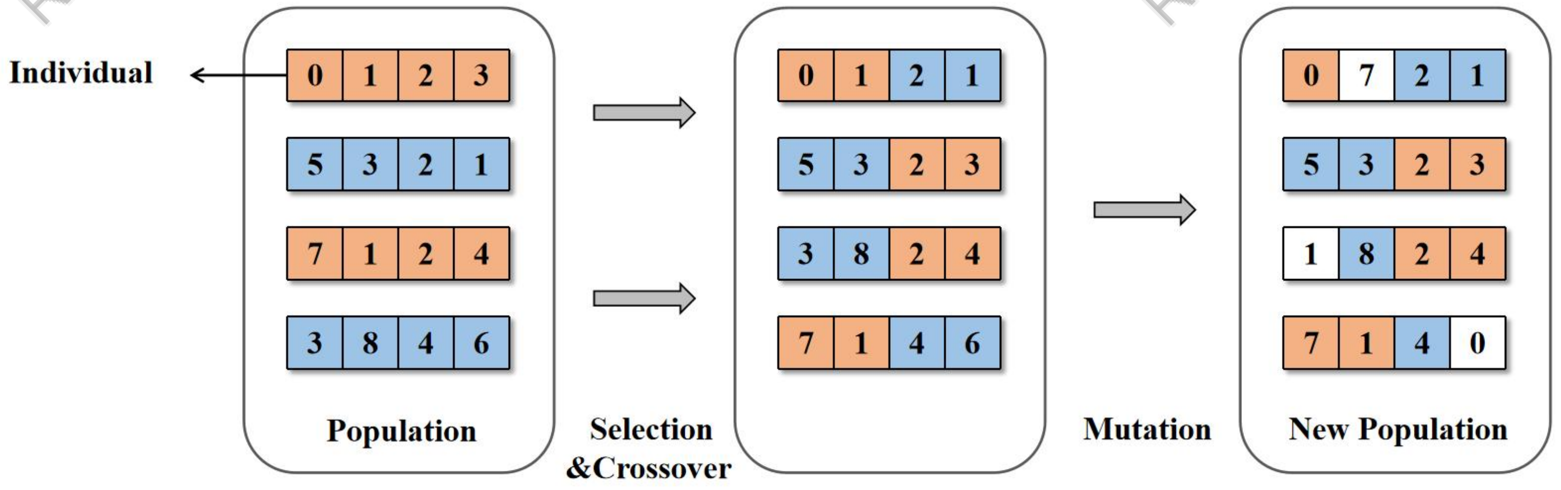
ERL-Re^2: Efficient Evolutionary RL with Shared State Representation and Individual Policy Representation

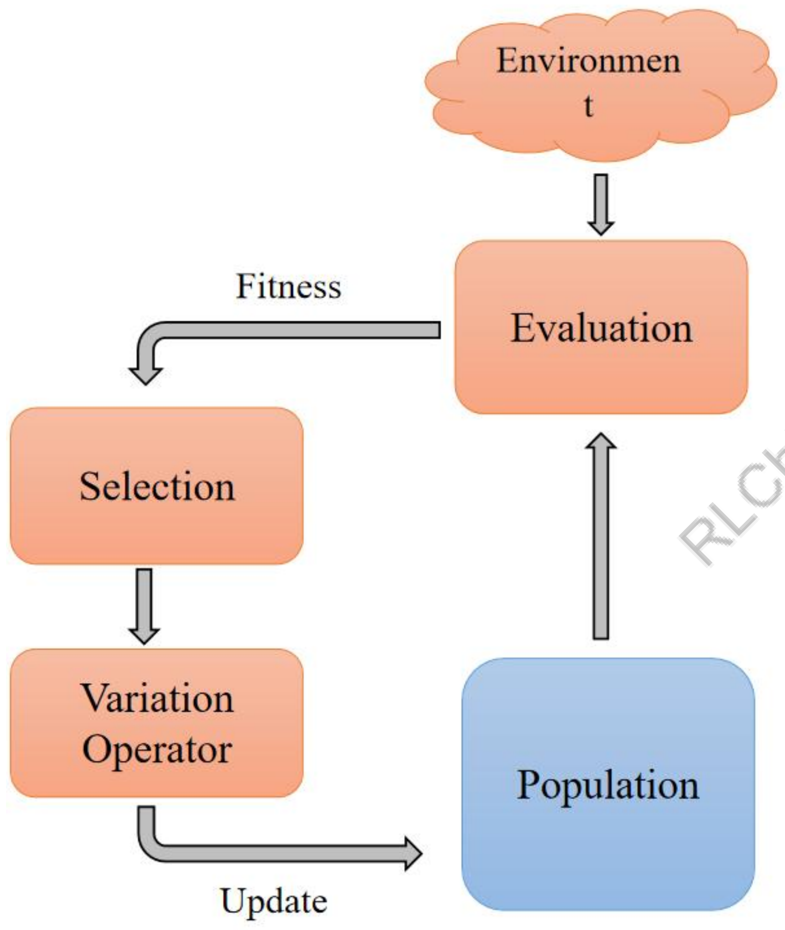
Pengyi Li
Tianjin University

ICLR 2023

Background: Evolutionary Algorithm (EA)

- Biologically inspired
- A class of black-box optimization methods
- Derivative-free, high robustness
- E.g., Genetic Algorithms (GA)



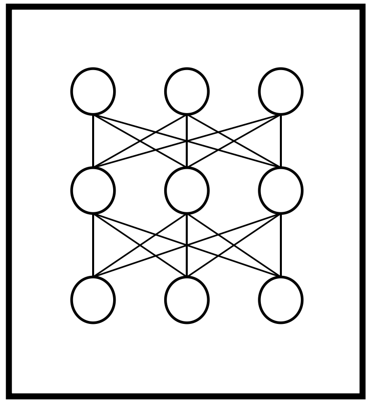


Background: EA in Markov Decision Process

- Structure of the individual
- Definition of fitness
- Take GA as an example, Crossover and mutation operator

Structure

Output: a



Input: s

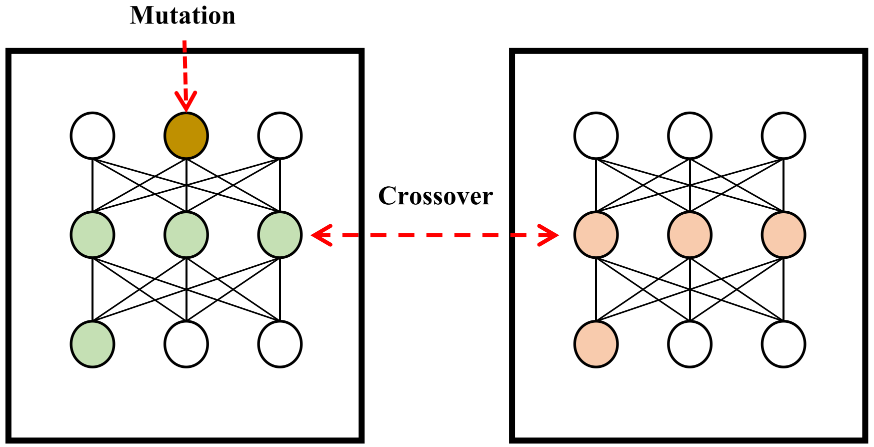
Fitness

$$\mathbb{P} = \{\pi_1, \pi_2, \ldots, \pi_n\}$$

$$f(\pi) = \frac{1}{e} \sum_{i=1}^{e} [\sum_{t=0}^{T} r_t \mid \pi]$$

$$\{f(\pi_1), f(\pi_2), \ldots, f(\pi_n)\}$$

Crossover and Mutation

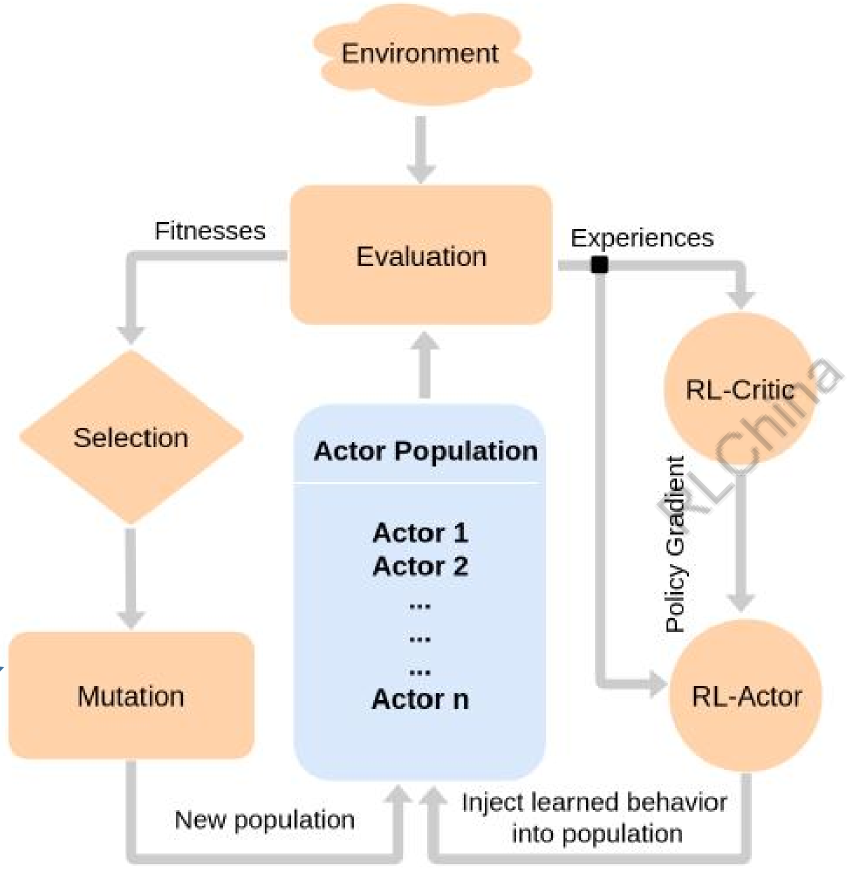


Parent A

Parent B

Background: Evolutionary Reinforcement learning

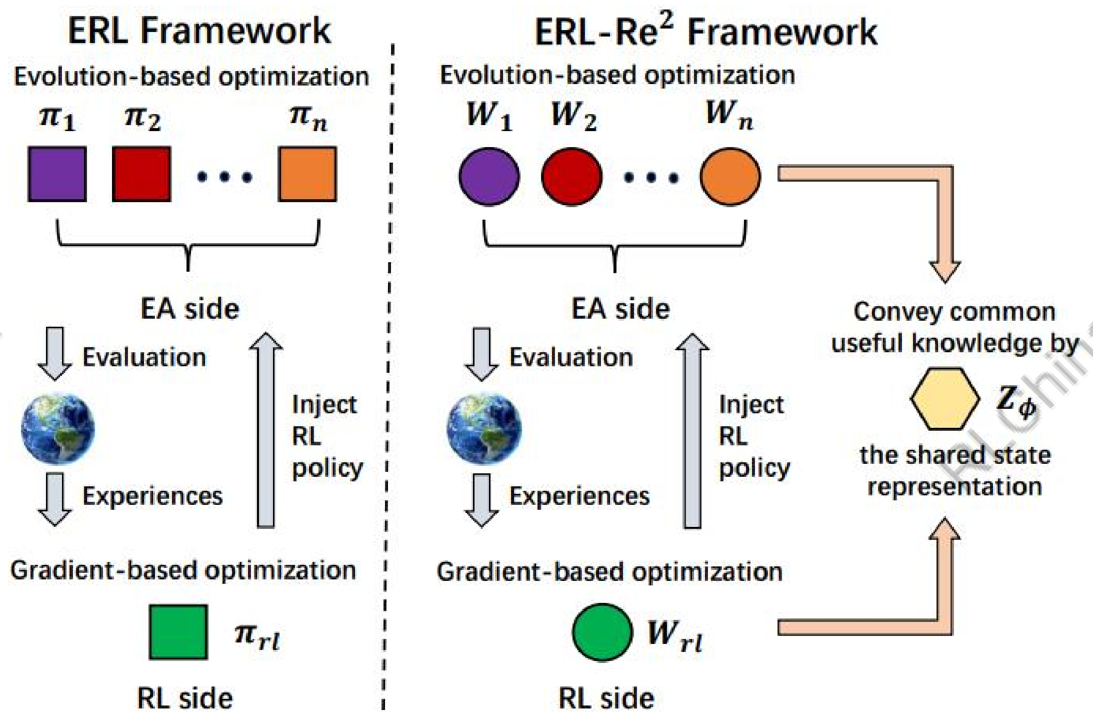
- Evolutionary Algorithm (EA) is a class of black-box optimization, which is demonstrated to be competitive with RL.
  - Different from RL, EA is gradient-free and offers several strengths: strong exploration ability, robustness, and stable convergence.
  - Despite the advantages, one major bottleneck of EA is the low sample efficiency due to the iterative evaluation of the population.
- Reinforcement learning (RL) can be learned efficiently by trial-and-error with reliable gradient updates and has higher sample efficiency than EA. However, RL is widely known to be unstable, poor in exploration, and struggling when the gradient signals are noisy and less informative.
- **Lots of works focus on how to infuse the distinct and complementary advantages of EA and RL for policy optimization.**



Current mainstream framework ERL to integrate EA and RL

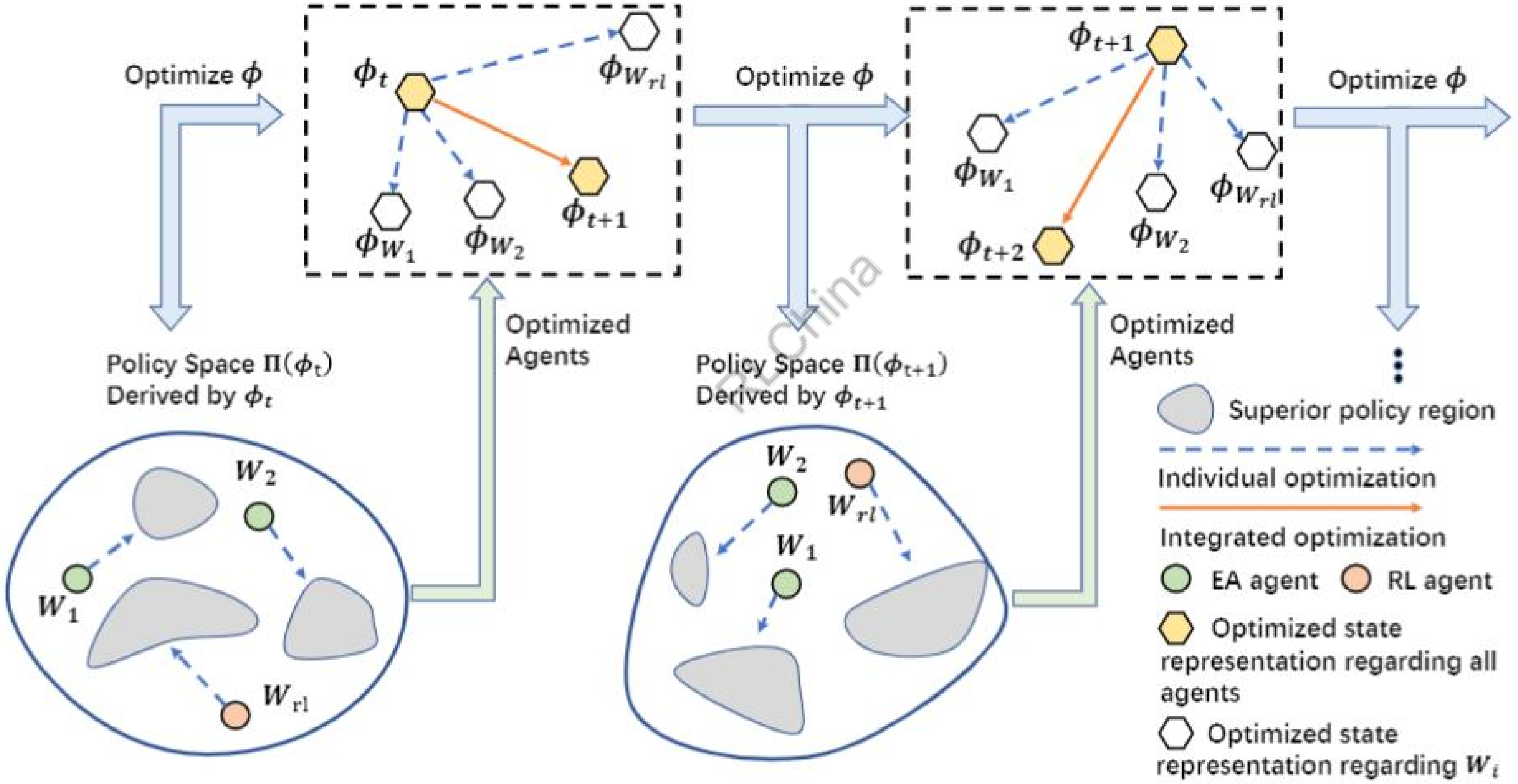
Motivation

- Existing works on combining RL and EA have two common drawbacks:
  - All agents learn their own state representation individually, neglecting efficient sharing of common useful features;
  - Parameter-level policy optimization guarantees no semantics of behavior evolution for the EA side and can be inefficient for the RL side. Consequently, it prevents the full use of the complementary advantages.



- To address the aforementioned two drawbacks,
  - We propose a novel approach ERL-Re^2 to integrate EA and RL based on the concept of two-scale representation;
  - We devise behavior-level crossover and mutation which have clear genetic semantics;
  - We empirically show that TSR-ERL consistently outperforms related methods and improves both its EA and RL components significantly

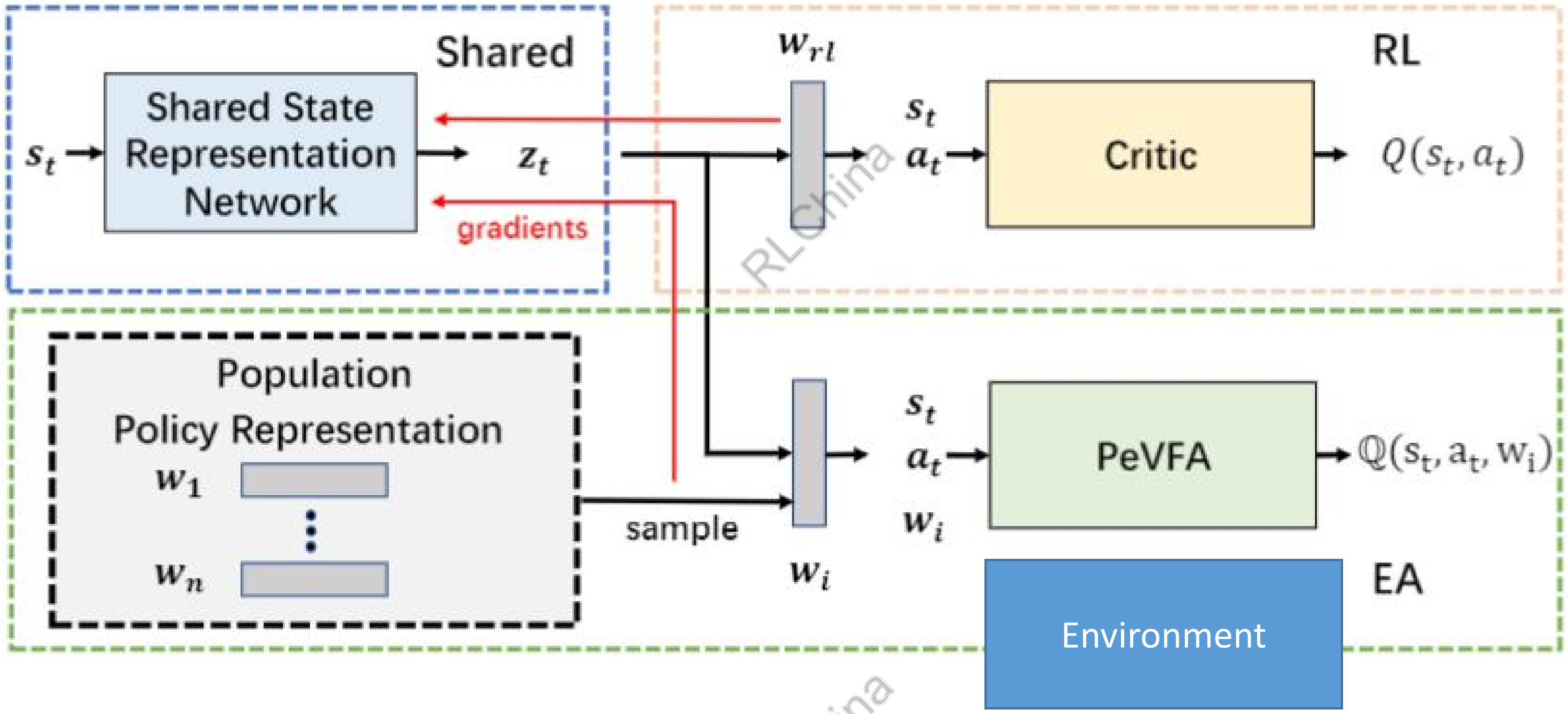
ERL-Re²: Efficient Evolutionary RL with Shared State Representation and Individual Policy Representation



Al the policies are composed of the nonlinear shared state representation Z_Φ and an individual linear policy representation W. In an iterative fashion, the two components are optimized at two scales.

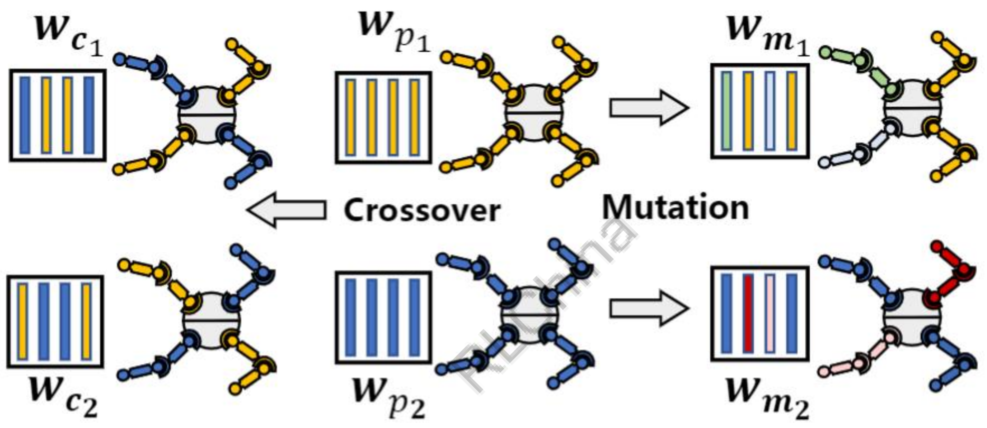
ERL-Re²: Efficient Evolutionary RL with Shared State Representation and Individual Policy Representation

Optimizing the Shared State Representation for A Superior Policy Space



ERL-Re²: Efficient Evolutionary RL with Shared State Representation and Individual Policy Representation

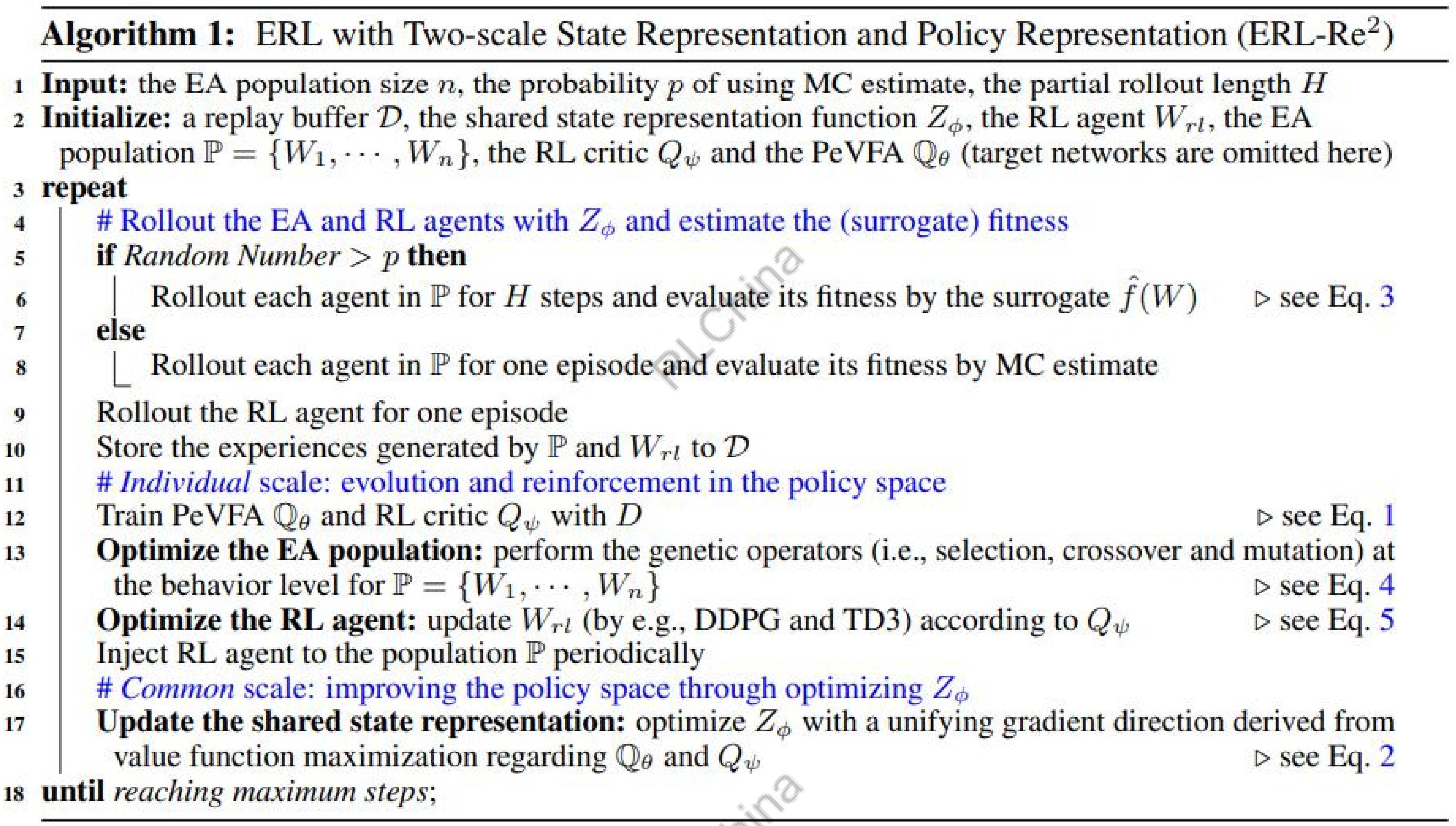
- Optimizing the Policy Representation with the **behavior-level** crossover and mutation in EA



- Surrogate Fitness Evaluation based on the PeVFA and partial rollout
  - Most prior methods rollout each agent in the population for one or several episodes and calculate the MC fitness - **major bottleneck** of EA especially when the population is large
  - Rollout the EA population for H steps with prob p
  - Compute the surrogate fitness for each agent W_i by H-step TD

$$\hat{f}(W_i) = \sum_{t=0}^{H-1} \gamma^t r_t + \gamma^H \mathbb{Q}_\theta(s_H, \pi_i(s_H), W_i)$$

Pseudocode



Algorithm 1: ERL with Two-scale State Representation and Policy Representation (ERL-Re^2)

```
Input: the EA population size $n$, the probability $p$ of using MC estimate, the partial rollout length $H$
Initialize: a replay buffer $\mathcal{D}$, the shared state representation function $Z_\phi$, the RL agent $W_{rl}$, the EA
  population $\mathbb{P} = \{W_1, \cdots, W_n\}$, the RL critic $Q_\psi$ and the PeVFA $\mathbb{Q}_\theta$ (target networks are omitted here)
repeat
    # Rollout the EA and RL agents with $Z_\phi$ and estimate the (surrogate) fitness
    if Random Number $> p$ then
        Rollout each agent in $\mathbb{P}$ for $H$ steps and evaluate its fitness by the surrogate $\hat{f}(W)$    ▷ see Eq. 3
    else
        Rollout each agent in $\mathbb{P}$ for one episode and evaluate its fitness by MC estimate
    Rollout the RL agent for one episode
    Store the experiences generated by $\mathbb{P}$ and $W_{rl}$ to $\mathcal{D}$
    # Individual scale: evolution and reinforcement in the policy space
    Train PeVFA $\mathbb{Q}_\theta$ and RL critic $Q_\psi$ with $D$    ▷ see Eq. 1
    Optimize the EA population: perform the genetic operators (i.e., selection, crossover and mutation) at
      the behavior level for $\mathbb{P} = \{W_1, \cdots, W_n\}$    ▷ see Eq. 4
    Optimize the RL agent: update $W_{rl}$ (by e.g., DDPG and TD3) according to $Q_\psi$    ▷ see Eq. 5
    Inject RL agent to the population $\mathbb{P}$ periodically
    # Common scale: improving the policy space through optimizing $Z_\phi$
    Update the shared state representation: optimize $Z_\phi$ with a unifying gradient direction derived from
      value function maximization regarding $\mathbb{Q}_\theta$ and $Q_\psi$    ▷ see Eq. 2
until reaching maximum steps;
```

ERL-Re²: Efficient Evolutionary Reinforcement Learning with Shared State Representation and Individual Policy Representation. ICLR 2023

Experiments

MuJoCo Continuous Control Benchmark

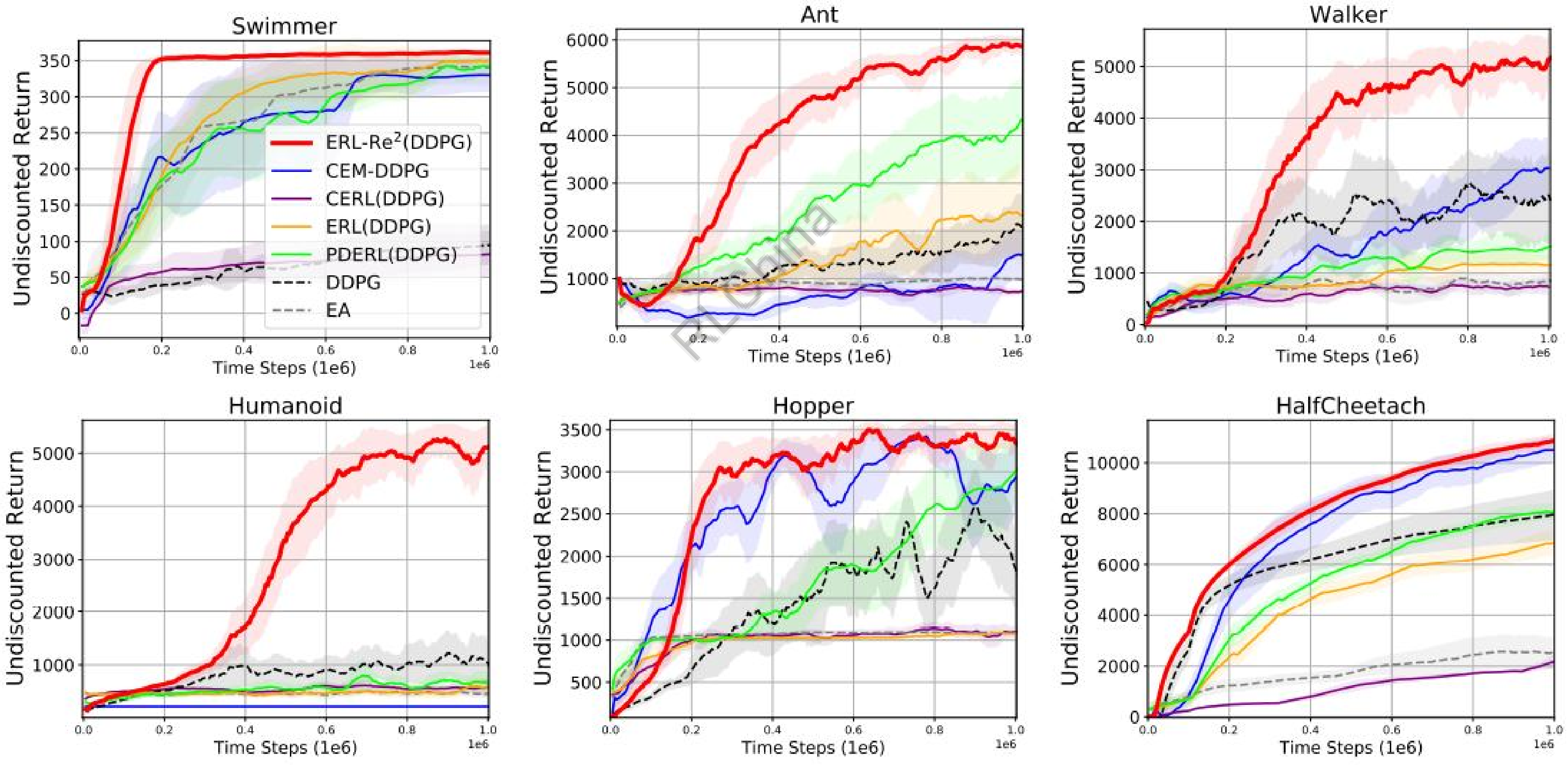


Figure 5: Performance comparison between ERL-Re^2 and baselines (all in the **DDPG** version).

Experiments

MuJoCo Continuous Control Benchmark

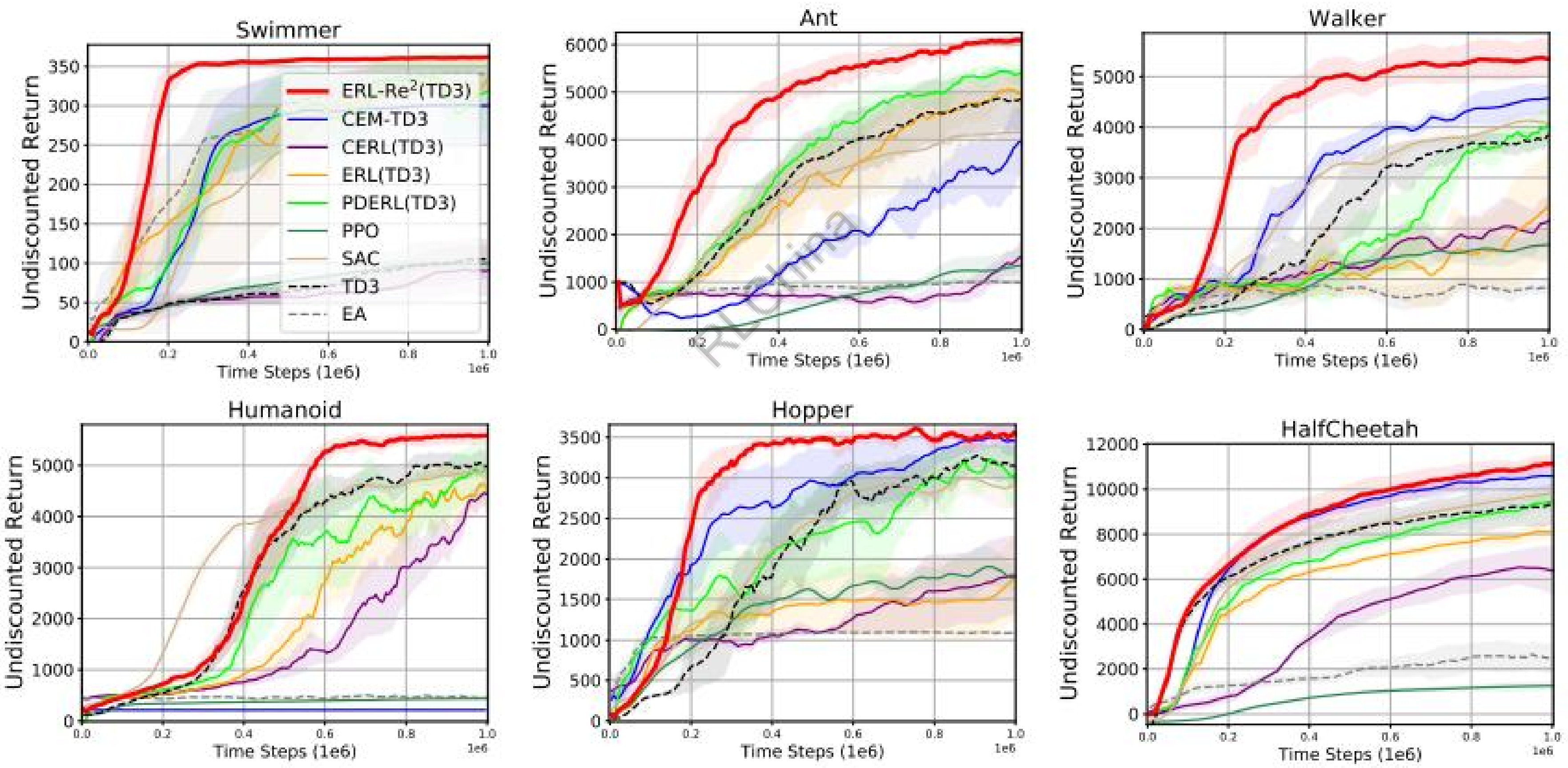


Figure 4: Performance comparison between ERL-Re^2 and baselines (all in the **TD3** version).

Experiments

DeepMind Control Suite (Visual-input Control)

- Implement ERL-Re^2 based on RAD (Laskin et al., NeurIPS 2020)

Environments	Cartpole Swingup	Cheetah Run	Finger Turn hard	Walker Walk
ERL-Re^2 (RAD)	854.453 $\pm$ 17.582	508.619 $\pm$ 22.622	197.35 $\pm$ 83.388	567.417 $\pm$ 139.278
RAD	728.66 $\pm$ 149.219	472.594 $\pm$ 29.595	126.725 $\pm$ 48.954	498.867 $\pm$ 142.509
ERL(RAD)	527.81 $\pm$ 126.48	329.709 $\pm$ 132.674	93.6 $\pm$ 130.53	287.969 $\pm$ 57.168
PDERL(RAD)	471.164 $\pm$ 220.221	331.903 $\pm$ 29.614	150 $\pm$ 70.711	252.492 $\pm$ 93.01
CEM-RAD	694.909 $\pm$ 44.74	421.574 $\pm$ 20.891	50.25 $\pm$ 67.67	66.042 $\pm$ 19.23
Steps	100k	100k	100k	100k

StarCraft Multiagent Challenge (Cooperative MARL)

- Extend and implement ERL-Re^2 based on FACMAC (Peng et al., NeurIPS 2021)
- Use MERL (Majumdar et al., ICML 2020) as the MA-ERL baseline

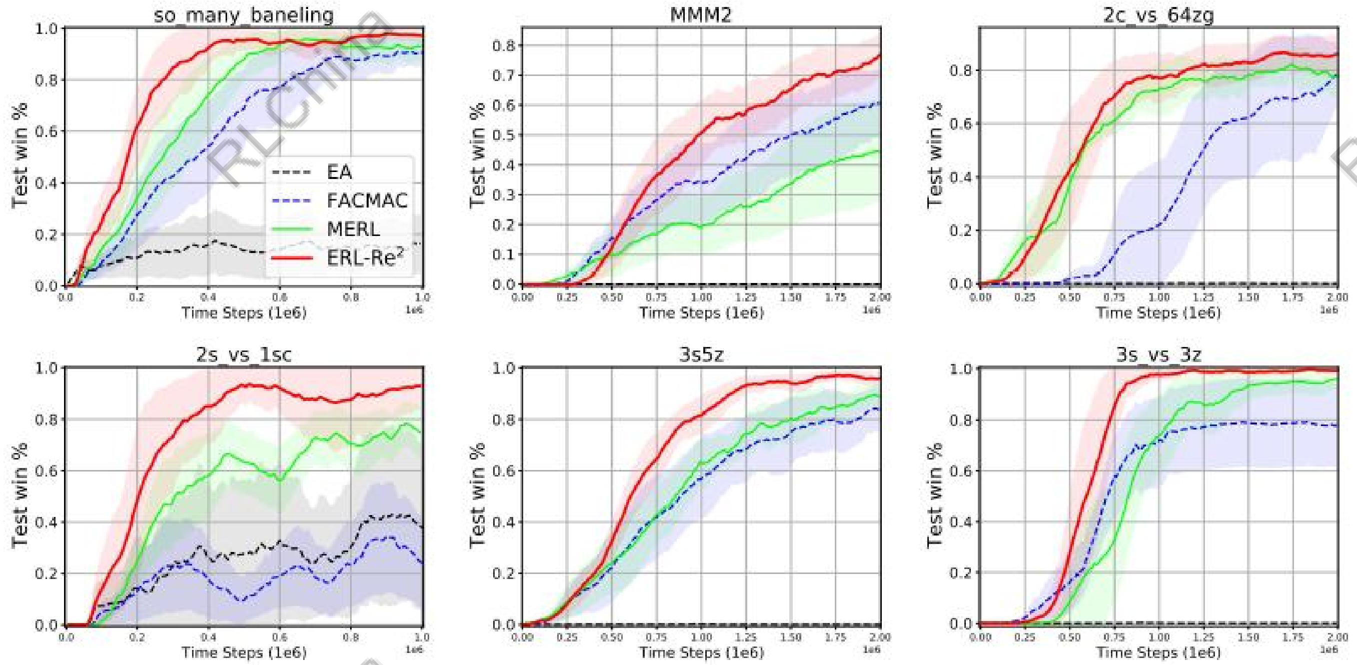


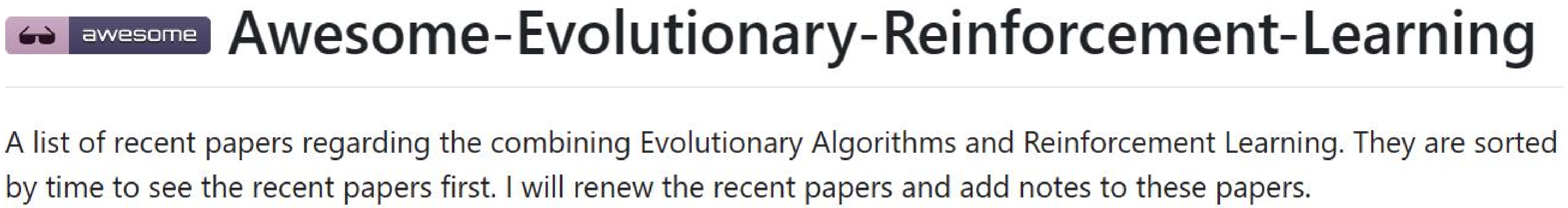
Figure 25: Performance comparison between ERL-Re^2 and baselines (all in the FACMAC version).

- Re^2 Code Release:

  https://github.com/yeshenpy/ERL-Re2

- Getting started with ERL:

  https://github.com/yeshenpy/Awesome-Evolutionary-Reinforcement-Learning



awesome Awesome-Evolutionary-Reinforcement-Learning

A list of recent papers regarding the combining Evolutionary Algorithms and Reinforcement Learning. They are sorted by time to see the recent papers first. I will renew the recent papers and add notes to these papers.

Mail: lipengyi@tju.edu.cn

Thanks Q&A