SELF-INFERENCE OF OTHERS' POLICIES FOR HOMOGENEOUS AGENTS IN COOPERATIVE MULTI-AGENT REINFORCEMENT LEARNING

Qifeng Lin School of Computer Science and Engineering, Sun Yat-Sen University

Qing Ling School of Computer Science and Engineering, Sun Yat-Sen University

Background

- Multi-agent reinforcement learning (MARL) remains to be a challenging research field and has various applications in multi-robot control, multi-player games, etc.
- In cooperative MARL, agents are trained to cooperatively achieve a global goal
- Partial observability: only local observation available rather than global states
  - is one of the critical challenges in MARL
  - motivates a training paradigm named centralized training and decentralized execution (CTDE)
- Policy inference: infer policies of other agents
  - plays an important role in MARL
  - is helpful to improve coordination efficiency

Related Works

- Fully observable scenarios
  - AMS-A3C and AMF-A3C share learned parameters and add extra policy features, respectively
  - Attention Multi-agent DDPG (ATT-MADDPG) introduces attention mechanism
  - × Hard to access global states in real world
- Partially observable scenarios
  - Extra hidden representations of other agents' policies are required
    - ∗ Deep reinforcement opponent network (DRON)
    - ∗ Deep policy inference Q-network (DPIQN) and deep policy inference recurrent Q-network (DPIRQN)
    - ∗ Multi-agent DDPG with policy inference (MADDPG-PI)
  - × Huge resource consumption

⇒ A self-inference approach to infer other agents' policies under

- cooperative MARL
- partially observable
- CTDE
- homogeneous agents

setting, called MADDPG-SI.

Advantages: significantly reduces computation and storage consumption.

Method

Partially observable Markov decision process (POMDP) with N agents

$$<\mathcal{S}, \mathcal{O}, \mathcal{A}, \mathcal{P}, \mathcal{R}>$$

- $\mathcal{S}$: the sets of state space
- $\mathcal{O}$: the joint observation spaces $\{\mathcal{O}_1, \dots, \mathcal{O}_N\}$
- $\mathcal{A}$: the joint action spaces $\{\mathcal{A}_1, \dots, \mathcal{A}_N\}$
- $\mathcal{R}$: the joint rewards $\{\mathcal{R}_1, \dots, \mathcal{R}_N\}$

The long-term return of agent i is $R_i = \sum_{t=0}^{T} (\gamma)^t r_i^t$

- γ is a discount factor
- T is the time horizon
- $r_i^t \in \mathcal{R}_i$ is the instantaneous reward at time t

Goal: find optimal policies $\boldsymbol{\mu}_i$:$\mathcal{O}_i \times \mathcal{A}_i \to [0,1]$ to maximize $R = \sum_{i=1}^{N} R_i$.

Further, we parameterize the policy $\boldsymbol{\mu}_i$ with θ_i.

For critic part in MADDPG-SI, the update rule of each agent i is by minimizing the critic loss:

$$\begin{aligned} \mathcal{C}(\phi_i) &= \mathbb{E}_{\mathbf{x},a,r,\tilde{\mathbf{x}} \sim \mathcal{D}}[(Q_i^{\boldsymbol{\mu}_{\theta_i}}(\mathbf{x}, a_1, \dots, a_N|\phi_i) - y_i)^2], \\ y_i &= r_i + \gamma Q_i^{\boldsymbol{\mu}_{\theta_i'}}(\tilde{\mathbf{x}}, a_1', \dots, a_N'|_{a_j' = \boldsymbol{\mu}_{\theta_i'}(o_j)}) \end{aligned} \quad (1)$$

$\mathcal{D}$ is the experience replay buffer.

Only use agent i's model to infer policies of other agents

For actor part in MADDPG-SI, the update rule of each agent i is by minimizing the actor loss

$$\mathcal{L}(\theta_i) - \beta \mathcal{P}_{SI}(\theta_i) \quad (2)$$

where

$$\begin{aligned} \mathcal{L}(\theta_i) &= -Q_i^{\boldsymbol{\mu}_{\theta_i}}(\mathbf{x}, a_1, \dots, a_N)|_{a_i = \boldsymbol{\mu}_{\theta_i}(o_i)} \\ \mathcal{P}_{SI}(\theta_i) &= -\sum_{j \neq i}^{N} \mathbb{E}_{o_j, a_j}[\log \boldsymbol{\mu}_{\theta_i}(a_j|o_j)] \end{aligned} \quad (3)$$

and β is a positive scale factor that balances learning from its own experience and learning from other agents' experience.

Compared with MADDPG-PI [1], MADDPG-SI requires less deep neural networks as given by

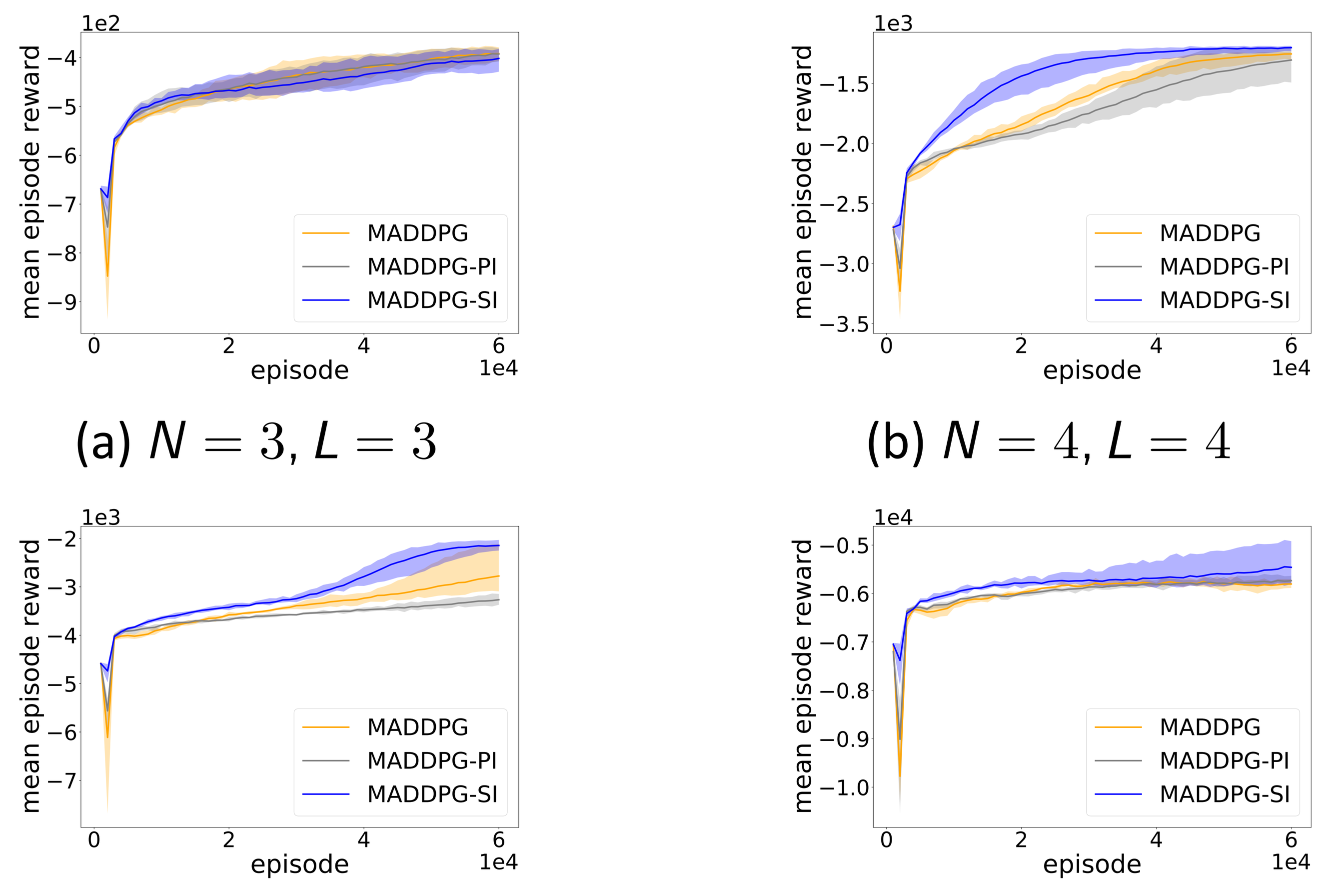
$$\begin{aligned} f_{PI} &= 2N(N+1), \\ f_{SI} &= 4N. \end{aligned} \quad (4)$$

Therefore, the space complexities for MADDPG-PI and MADDPG-SI are $O(N^2)$ and $O(N)$, respectively.

Experimental Results

- Environment: cooperative navigation with N agents and L landmarks [1]
  - Task: agents occupy all the landmarks cooperatively
  - A shared reward:
    - ∗ sums up negative distances between agents and landmarks
    - ∗ every collision between the agents contributes -1
- Four settings: (N=3, L=3), (N=4, L=4), (N=5, L=5), and (N=6, L=6).

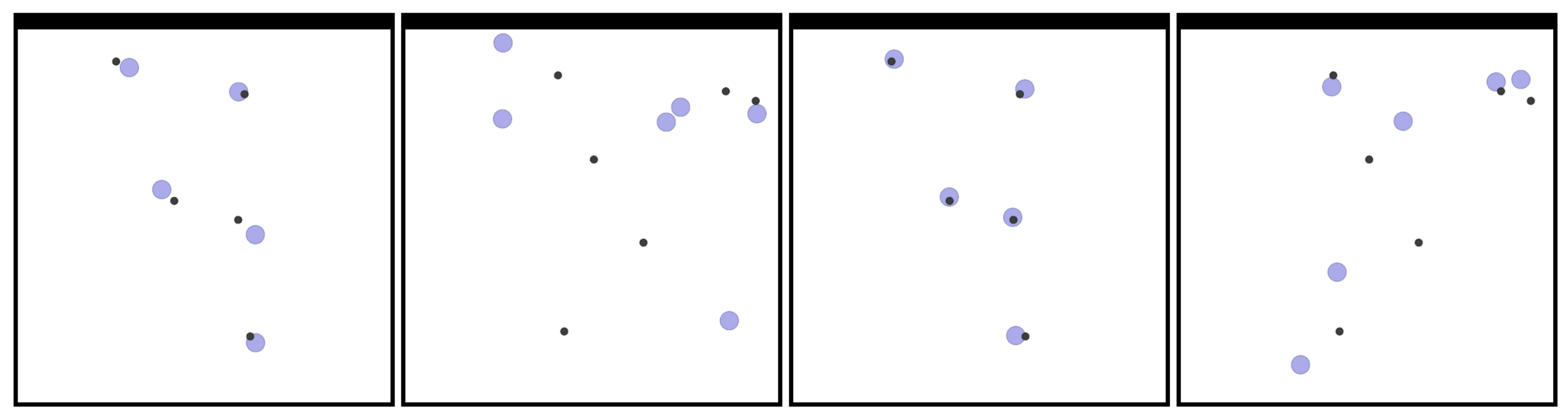
As shown in **Fig. 1**, MADDPG-SI can achieve almost equivalent performance and even outperform MADDPG and MADDPG-PI in some cases. **Fig. 2** shows that the agent of MADPPG-SI can be closer to landmarks compared with the one of MADDPG.



(a) $N=3, L=3$ (b) $N=4, L=4$

(c) $N=5, L=5$ (d) $N=6, L=6$

Fig. 1. Reward after 60000 episodes from 5 runs.



(a) $N=5, L=5$ (b) $N=6, L=6$ (c) $N=5, L=5$ (d) $N=6, L=6$

Fig. 2. Illustration of execution-stage performance of the agents trained by MADDPG in (a) and (b), and MADDPG-SI in (c) and (d). Small dark circles are landmarks and blue circles indicate agents.

References

[1] R. Lowe, Y. Wu, A. Tamar, J. Harb, O. P. Abbeel, and I. Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. In *Advances in Neural Information Processing Systems*, pages 6379–6390, 2017.