Robust Multi-Agent Reinforcement Learning By Minimax Deep Deterministic Policy Gradient

Shihui Li[1] Yi Wu[2] Xinyue Cui[3] Honghua Dong[3]
Fei Fang[1] Stuart Russell[2]
[1]CMU [2]UC Berkeley [3]Tsinghua U.

Overview (TL;DR)

- Improve the **robustness** of deep MARL policies – agent can still work well when opponent changes
- Introduce **minimax** into the MARL and optimize the objective using **adversarial learning** techniques
- A new algorithm **M3DDPG** (**M**ini**M**ax **M**ulti-agent **D**eep **D**eterministic **P**olicy **G**radient) – a minimax extension of MADDPG
- GitHub: https://github.com/dadadidodi/m3ddpg

Motivation

Multi-agent reinforcement learning (MARL) is promising for building intelligent agents

- Learns complex strategies with little human supervisions: StarCraft, Dota, robotics, green-security games, etc
- Provides a natural curriculum: agents help other agents get better

Deep RL policy is brittle & tends to overfit

- The environment is non-stationary from each single agent's perspective → unstable
- Naïve MARL converges to a brittle equilibrium
- Opponent changes strategy → perform worse!

Robustness is critical for practical AI!

Minimax MARL

Idea: learning minimax policy

- We consider continuous action
- Assume opponents perform adversarially
- Naturally leads to MADDPG formulation – decentralized Q function and policy $\mu_i(o_i;\theta_i)$

Key point: defining minimax Q function

- $Q_{M,i}(x, a_1, \dots, a_N)$: the Q value for agent i at state s with actions $a_1, \dots, a_N$ from N agents, *assuming other agents are adversarial*
- $Q_{M,i}(x, a_1, \dots, a_N) = r(x, a_1, \dots, a_N) + \gamma \min_{a'_{j\neq i}} Q_{M,i}(x', a'_1, \dots, a'_N)$
- $J(\theta_i) = \min_{a_{j\neq i}} \mathbb{E}_{x_0}\left[Q_{M,i}(x_0, a_1, \dots, a_N)\right]$

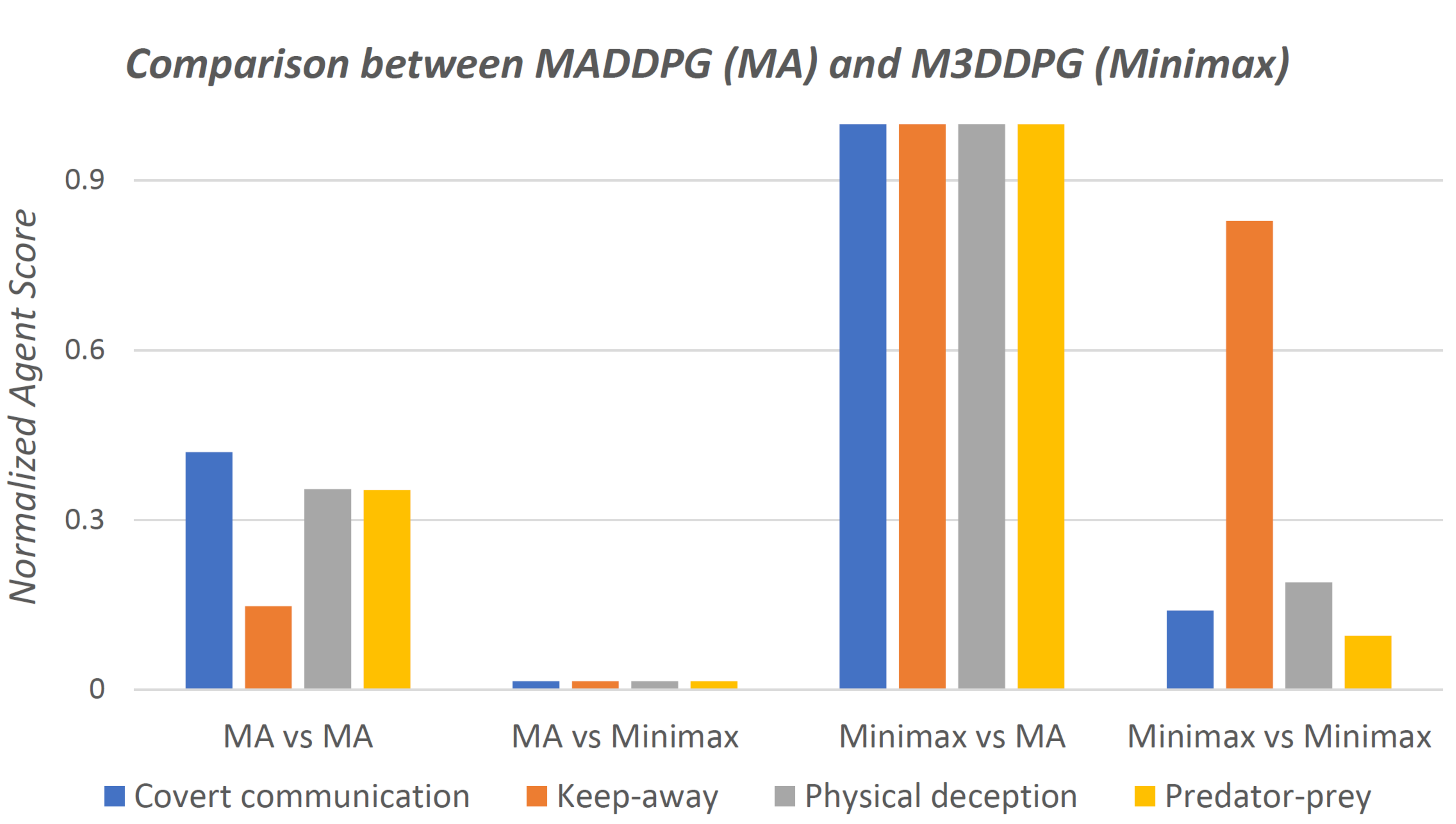
Multi-Agent Adversarial Learning

Adversarial learning for optimizing $J(\theta_i)$

- Replace $\min$ by one-step gradient descent
- Similar to *adversarial training*, each agent performs worst perturbation from its current policy
- Advantages: fast, differentiable, natural curriculum

$$\nabla_{\theta_i} J(\theta_i) = \mathbb{E}_{\mathbf{x},a\sim\mathcal{D}}\left[\begin{array}{l} \nabla_{\theta_i}\boldsymbol{\mu}_i(o_i)\nabla_{a_i}Q^{\boldsymbol{\mu}}_{\mathrm{M},i}(\mathbf{x}, a^\star_1, \dots, a_i, \dots a^\star_N)\Big| \\ \quad a_i = \boldsymbol{\mu}_i(o_i) \\ \quad a^\star_j = a_j + \hat{\epsilon}_j, \quad \forall j \neq i \\ \quad \hat{\epsilon}_j = -\alpha_j \nabla_{a_j} Q^{\boldsymbol{\mu}}_{\mathrm{M},i}(\mathbf{x}, a_1, \dots, a_N) \end{array}\right] \quad (13)$$

$$\begin{aligned} \mathcal{L}(\theta_i) &= \mathbb{E}_{\mathbf{x},a,r,\mathbf{x}'}[(Q^{\boldsymbol{\mu}}_{\mathrm{M},i}(\mathbf{x}, a_1, \dots, a_N) - y)^2], \quad (14) \\ y &= r_i + \gamma Q^{\boldsymbol{\mu}'}_{\mathrm{M},i}(\mathbf{x}', a'^\star{}_1, \dots, a'_i, \dots, a'^\star{}_N) \\ a'_k &= \boldsymbol{\mu}'_k(o_k), \quad \forall 1 \leq k \leq N \\ a'^\star{}_j &= a'_j + \hat{\epsilon}'_j, \quad \forall j \neq i \\ \hat{\epsilon}'_j &= -\alpha_j \nabla_{a'_j} Q^{\boldsymbol{\mu}'}_{\mathrm{M},i}(\mathbf{x}, a'_1, \dots, a'_N), \end{aligned}$$



Normalized performance score between *M3DDPG* and *MADDPG* on 4 competitive environments.
M3DDPG learns better, more robust policies.

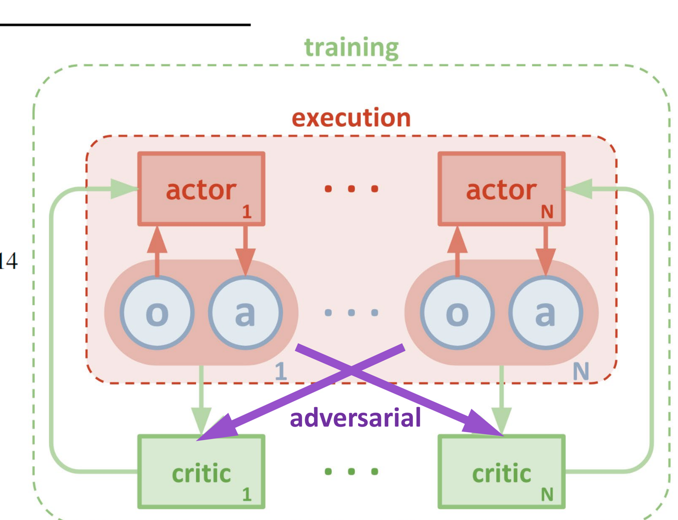
M3DDPG

Algorithm 1: Minimax Multi-Agent Deep Deterministic Policy Gradient (M3DDPG) for N agents

for episode $= 1$ to M **do**
  Initialize a random process $\mathcal{N}$ for action exploration, and receive initial state information $\mathbf{x}$
  for $t = 1$ to max-episode-length **do**
    for each agent i, select action $a_i = \boldsymbol{\mu}_{\theta_i}(o_i) + \mathcal{N}_t$ w.r.t. the current policy and exploration
    Execute actions $a = (a_1, \dots, a_N)$ and observe reward r and new state information $\mathbf{x}'$
    Store $(\mathbf{x}, a, r, \mathbf{x}')$ in replay buffer $\mathcal{D}$, and set $\mathbf{x} \leftarrow \mathbf{x}'$
    for agent $i = 1$ to N **do**
      Sample a random minibatch of S samples $(\mathbf{x}^k, a^k, r^k, \mathbf{x}'^k)$ from $\mathcal{D}$
      Set $y^k = r^k_i + \gamma Q^{\boldsymbol{\mu}'}_{\mathrm{M},i}(\mathbf{x}'^k, a'_1, \dots, a'_N)|_{a'_i=\boldsymbol{\mu}'_i(o^k_i), a'_{j\neq i}=\boldsymbol{\mu}'_j(o^k_j)+\hat{\epsilon}'_j}$ with $\hat{\epsilon}'_j$ defined in Eq. 14
      Update critic by minimizing the loss $\mathcal{L}(\theta_i) = \frac{1}{S}\sum_k \left(y^k - Q^{\boldsymbol{\mu}}_{\mathrm{M},i}(\mathbf{x}^k, a^k_1, \dots, a^k_N)\right)^2$
      Update actor using the sampled policy gradient with $\hat{\epsilon}_j$ defined in Eq. 13:

$$\nabla_{\theta_i} J \approx \frac{1}{S}\sum_k \nabla_{\theta_i}\boldsymbol{\mu}_i(o^k_i)\nabla_{a_i}Q^{\boldsymbol{\mu}}_{\mathrm{M},i}(\mathbf{x}^k, a^\star_1, \dots, a_i, \dots, a^\star_N)\big|_{a_i=\boldsymbol{\mu}_i(o^k_i), a^\star_{j\neq i}=a^k_j+\hat{\epsilon}_j}$$

    end for
    Update target network parameters for each agent i: $\theta'_i \leftarrow \tau\theta_i + (1-\tau)\theta'_i$
  end for
end for



The worst case performance – compete against *best response*
M3DDPG performs better with the worst opponents