Multi-agent System and Application

Xiangfeng Wang

Multi-Agent Artificial Intelligence Laboratory,
School of Computer Science and Technology,
East China Normal University

2021 Spring

Table of Contents

Challenges for Multi-agent Reinforcement Learning

M3DDPG

From Multi-Agent to Many-Agent RL

Mean Field Multi-Agent RL

Real World Applications and simulators

Key Factors for Successful MARL

Table of Contents

Challenges for Multi-agent Reinforcement Learning

M3DDPG

From Multi-Agent to Many-Agent RL

Mean Field Multi-Agent RL

Real World Applications and simulators

Key Factors for Successful MARL

Challenges for Multi-agent Reinforcement Learning

- Non-stationary issue
  - RL assumes a stationary environment (fixed MDP);
  - If directly apply single-agent RL, other agents becomes a part of the environment;
  - Non-stationery from each agent's own perspective.
- Unstable training
  - Neural networks can hardly converge;
  - Degenerate to bad local mode.
- Easier to overfit
  - Learned policy overfits its training partners;
  - Much worse when testing with a different opponent.
- Large scale problem
  - What will happen when agent number grows?
  - Learning will be much more difficult.

Recap: MADDPG algorithm

MADDPG, the first general purpose deep MARL algorithm for stabilizing training (Non-stationary issue and unstable training)

- Multi-Agent Deep Deterministic Policy Gradient (MADDPG, Lowe*, Wu*, et.al., NIPS2017)
- Use actor-critic framework
  - Decentralized actors (policies, π) to keep the **self-play framework**
  - Centralized critic (baseline, Q) to ease learning and reduce variance
- Applies to mixed competitive and cooperative environments

Recap: MADDPG algorithm

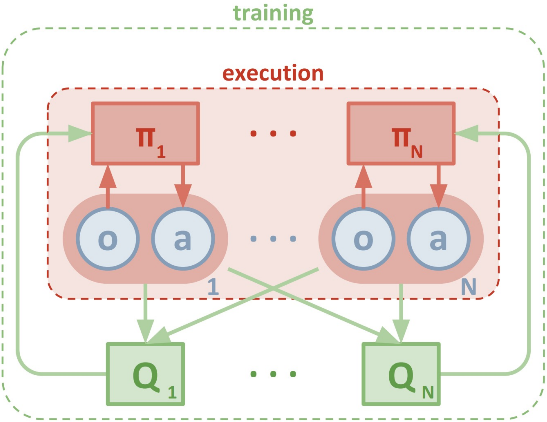


Figure 1: Overview of our multi-agent decentralized actor, centralized critic approach.

M3DDPG

- What about overfitting challenge?
  - M3DDPG
  - A extension and variant of MADDPG for **robust policies**

Table of Contents

Challenges for Multi-agent Reinforcement Learning

M3DDPG

From Multi-Agent to Many-Agent RL

Mean Field Multi-Agent RL

Real World Applications and simulators

Key Factors for Successful MARL

Overview of M3DDPG

Goal: learning robust policies in deep MARL

- Key ideas
  - Minimax objective: introduce minimax concept into deep MARL;
  - Multi-Agent Adversarial Learning (MAAL): use techniques from **adversarial training** for tractable and practical approximate computation.
- The algorithm: **M**iniMax **M**ulti-agent **D**eep **D**eterministic **P**olicy **G**radient (M3DDPG)
  - Simple, efficient and improved MADDPG

Minimax MARL Objective

- Minimax is a fundamental idea in zero-sum games;
- Minimax MARL: learning minimax deep policies.
- Classic Q-function:

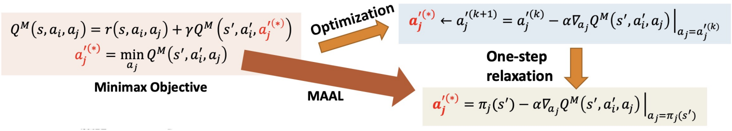
$$\begin{aligned} Q(s,a) =& (1-\alpha)Q(s,a) + \alpha(r + \gamma V(s')) \\ V(s') =& \max_{a'} Q(s',a') \end{aligned} \tag{1}$$

- Minmax Q-function:

$$\begin{aligned} Q(s,a,o) =& (1-\alpha)Q(s,a,o) + \alpha(r + \gamma V(s')) \\ V(s') =& \max_{\pi(s)} \min_{o} \sum_{a} Q(s,a,o)\pi_s(a) \end{aligned} \tag{2}$$

Multi-Agent Adversarial Learning

- The inner loop minimization is intractable and expensive to compute
  - No closed-form solution;
  - An expensive inner-loop gradient descent optimization process



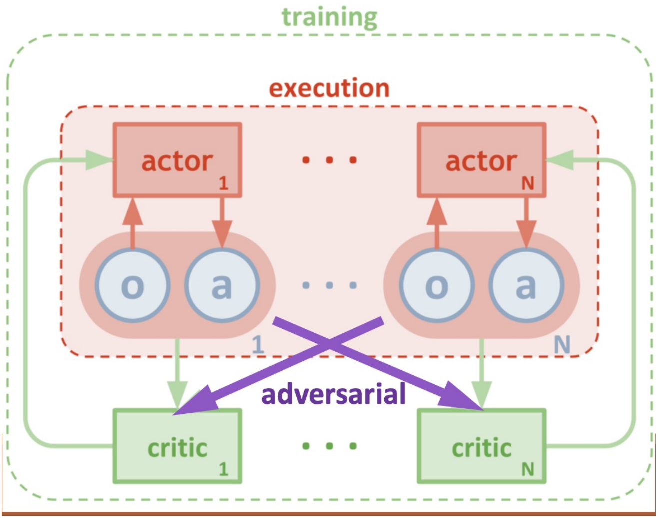
- Multi-agent Adversarial Learning (MAAL)
  - Key idea: replace the inner-loop minimization by a one-step gradient descent;
  - Motivated by recent successes of adversarial training (Goodfellow, et. al, ICLR 2014), and meta-learning (MAML, Finn, et. al., ICML 2017);
  - Fully differentiable, efficient approximation, effective in practice.

Overall Framework of M3DDPG

- M3DDPG:
  - Fast, efficient approximate algorithm for robust learning
  - Trade-off between objective and practical computation

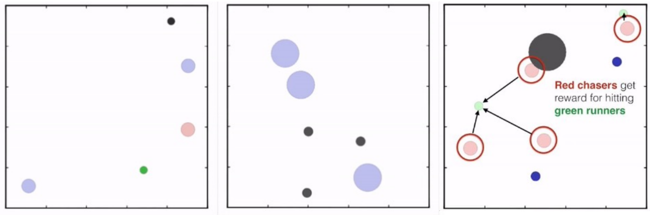
$$\nabla_{\theta_i} J(\theta_i) = \mathbb{E}_{\mathbf{x}, a \sim \mathcal{D}} \left[\begin{array}{l} \nabla_{\theta_i} \boldsymbol{\mu}_i(o_i) \nabla_{a_i} Q^{\boldsymbol{\mu}}_{\mathrm{M},i}(\mathbf{x}, a_1^\star, \ldots, a_i, \ldots a_N^\star) \Big| \\ \quad a_i = \boldsymbol{\mu}_i(o_i) \\ \quad a_j^\star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

Overall Framework of M3DDPG

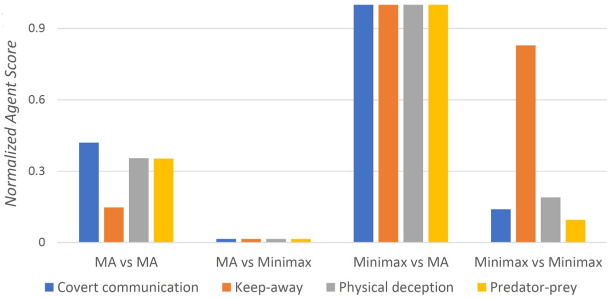


M3DDPG – Environment

- Environment and tasks:
  - The particle world environment from MADDPG (demo below)
  - Test 1: competition between M3DDPG and MADDPG
  - Test 2: performance against the best response



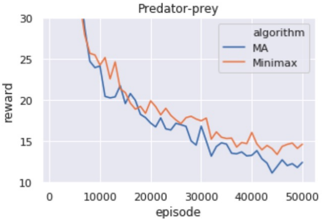
M3DDPG – Results



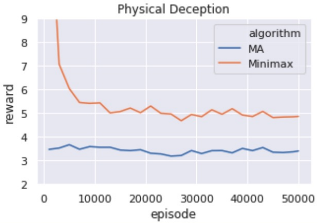
M3DDPG – Result

Worst Case Performance (BR)

- Evaluate the performance of M3DDPG and MADDPG against its best response policy
  - Fixed the policies learned by M3DDPG and MADDPG
  - Retrain a single opponent (reduce to single-agent RL setting, easy)



(a) a



(b) a

M3DDPG:Conclusion

- Take-home message
  - Minimax MARL + adversarial learning $\rightarrow$ M3DDPG (Minimax MADDPG);
  - A fully differentiable, fast, general algorithm for robust deep MARL;
  - Interpretation: adversarial training extension of MADDPG.

Table of Contents

Challenges for Multi-agent Reinforcement Learning

M3DDPG

From Multi-Agent to Many-Agent RL

Mean Field Multi-Agent RL

Real World Applications and simulators

Key Factors for Successful MARL

From Multi-Agent to Many-Agent RL

- What will happen when agent number grows?
  - Reward function of agent $r^j : S \times A^1 \times ... \times A^N \longrightarrow \mathbb{R}$
  - Transition probability $p : S \times A^1 \times ... \times A^N \longrightarrow \Omega(S)$
- Both reward function and state transition probability get exponentially larger
  - More difficult to model;
  - The environment is more dynamic and sensitive;
  - Need more exploration data;
  - More computational resources.

Idea: Taking Other Agents as A Whole

- In some many-body systems, the interaction between an agent and others can be approximated as that between the agent and the "mean agent" of others.

Table of Contents

Challenges for Multi-agent Reinforcement Learning

M3DDPG

From Multi-Agent to Many-Agent RL

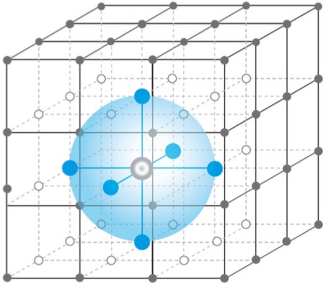
Mean Field Multi-Agent RL

Real World Applications and simulators

Key Factors for Successful MARL

Mean Field Multi-Agent RL

- Mean field approximation
  - Approximate the joint action value by factorizing the Q-function into pairwise interactions $Q^i(s,a) = \frac{1}{N^j}\sum_{k\in N(j)} Q^j(s,a^j,a^k)$



- Significantly reduces the global interactions among agents;
- Still preserves global interactions of any agent pair.

Action Representation

$Q^i(s,a) = \frac{1}{N^j}\sum_{k\in N(j)} Q^j(s,a^j,a^k)$

- Consider discrete action space
  - Action a^j of agent j is one-hot encoded as:

$$a^j \triangleq [a^j_1, ..., a^j_D] \text{ Only one element is 1} \tag{3}$$

  - The mean action based on the neighborhood of j is:

$$\bar{a}^j = \frac{1}{N^j}\sum_k a^k \tag{4}$$

  - Thus the action a^k of each neighbor k can be represented as:

$$a^k = \bar{a}^j = \delta a^{j,k}, \tag{5}$$

Here $\delta a^{j,k}$ is the residual and the residual sum is 0.

Mean Field Approximation

- A 2-order Taylor expansion on Q-function

$$\boxed{a^k = \bar{a}^j + \delta a^{j,k}}$$

$$
\begin{aligned}
Q^j(s,\boldsymbol{a}) &= \frac{1}{N^j}\sum_k Q^j(s,a^j,a^k) \\
&= \frac{1}{N^j}\sum_k \left[Q^j(s,a^j,\bar{a}^j) + \nabla_{\bar{a}^j}Q^j(s,a^j,\bar{a}^j)\cdot \delta a^{j,k} + \frac{1}{2}\delta a^{j,k}\cdot \nabla^2_{\tilde{a}^{j,k}}Q^j(s,a^j,\tilde{a}^{j,k})\cdot \delta a^{j,k}\right] \\
&= Q^j(s,a^j,\bar{a}^j) + \nabla_{\bar{a}^j}Q^j(s,a^j,\bar{a}^j)\cdot \frac{1}{N^j}\sum_k \delta a^{j,k} + \frac{1}{2N^j}\sum_k \delta a^{j,k}\cdot \nabla^2_{\tilde{a}^{j,k}}Q^j(s,a^j,\tilde{a}^{j,k})\cdot \delta a^{j,k} \\
&= Q^j(s,a^j,\bar{a}^j) + \frac{1}{2N^j}\sum_k R^j_{s,a^j}(a^k) \\
&\approx Q^j(s,a^j,\bar{a}^j)
\end{aligned}
$$

External random signal for agent *j*

Q-function model the interaction between the agent's action and the mean action

$$R^j_{s,a^j}(a^k) \triangleq \delta a^{j,k}\cdot \nabla^2_{\tilde{a}^{j,k}}Q^j(s,a^j,\tilde{a}^{j,k})\cdot \delta a^{j,k}$$

$$\tilde{a}^{j,k} = \bar{a}^j + \epsilon^{j,k}\delta a^{j,k}$$

Mean Field Q-Learning

- A softmax MF-Q policy:

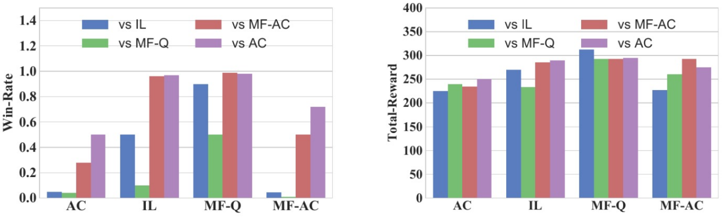
$$\pi_t^j(a^j|s,\bar{a}^j) = \frac{exp(\beta Q_t^j(s,a^j,\bar{a}^j))}{\sum_{a^{j\prime}\in A^j} exp(\beta Q_t^j(s,a^{j\prime},\bar{a}^j))} \tag{6}$$

- Given an experience $s, a, r, s', \bar{a}$ sampled from replay buffer
  - Sample the next action a_-^j from $Q_{\phi_-^j}$
  - Set $y^j = r^j + \gamma Q_{\phi_-^j}(s', a_-^j, \bar{a}^j)$
  - Update Q function with the loss function $\zeta(\phi^j) = (y^j - Q_{\phi^j}(s', a^j, \bar{a}^j))$

MF-Q Convergence

- Theorem: In a finite-state stochastic game, the Q values computed by the update rule of MF-Q converges to the Nash Q-value
  - under certain assumptions of reward function, policy form and game equilibrium

Experiment Performance Battle



(a) Average wining rate. (b) Average total reward.

- For 64 vs 64 battle, MF-Q works the best among all compared models
- MF-AC may not work that well particularly when the agent number is large

Table of Contents

Challenges for Multi-agent Reinforcement Learning

M3DDPG

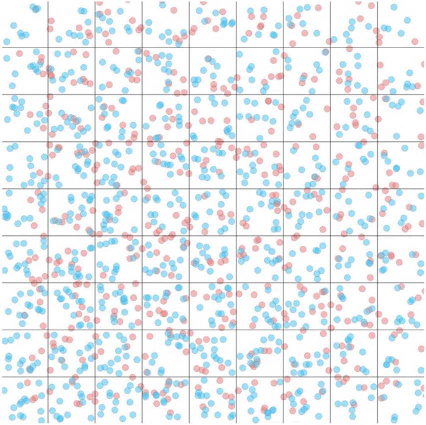
From Multi-Agent to Many-Agent RL

Mean Field Multi-Agent RL

Real World Applications and simulators

Key Factors for Successful MARL

Online Taxi Order Dispatch



Blue points: orders

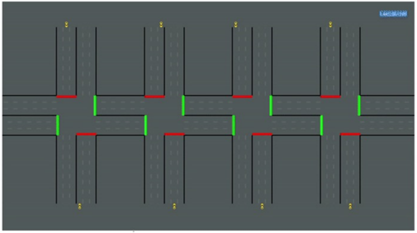
Red points: taxis

Traffic Signal Control

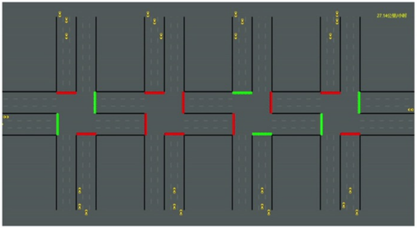
,

- A special version of UI
- Same flow
- Difference between

Bad Traffic Signal Plan



Good Traffic Plan



Simulators of MARL

- Traffic Signal Control: CityFlow
  https://github.com/cityflow-project/CityFlow
- Discrete game world: MAgent
  https://github.com/geek-ai/MAgent
- Another game world: Discrete game world (Your final project)
  https://github.com/MultiAgentLearning/playground

Table of Contents

Challenges for Multi-agent Reinforcement Learning

M3DDPG

From Multi-Agent to Many-Agent RL

Mean Field Multi-Agent RL

Real World Applications and simulators

Key Factors for Successful MARL

Key Factors for Successful MARL

- Computation: High computational resource for reinforcement learning;
- Data: a huge amount of data for training the models;
- Environment: a low-cost environment for RL agents to interact with
- Algorithm: an effective MARL algorithm for learning good policies and learn coordination among agents.

Discussions

Thank You