45th IEEE International Conference on Acoustics, Speech and Signal Processing
ICASSP 2020

Stabilizing Multi-agent Deep Reinforcement Learning by Implicitly Estimating Other Agents' Behaviors

Yue Jin[1] Shuangqing Wei[2] Jian Yuan[1] Xudong Zhang[1] Chao Wang[1]

[1] Department of Electronic Engineering, Tsinghua University

[2] School of Electrical Engineering and Computer Science, Louisiana State University

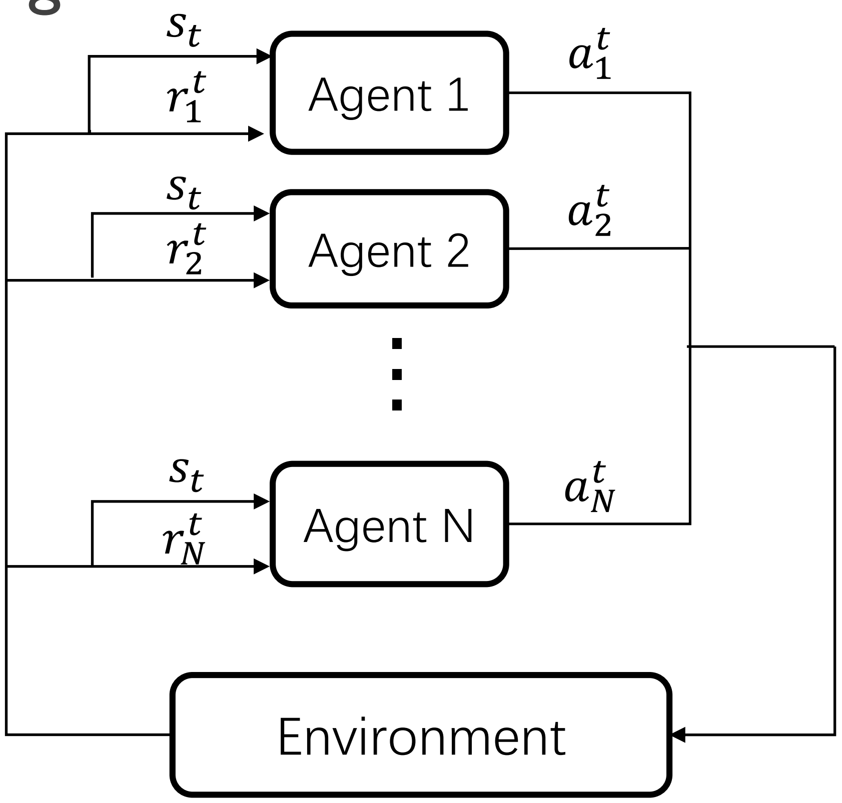
Outline

- Background
- Method
- Experiments
- Conclusion

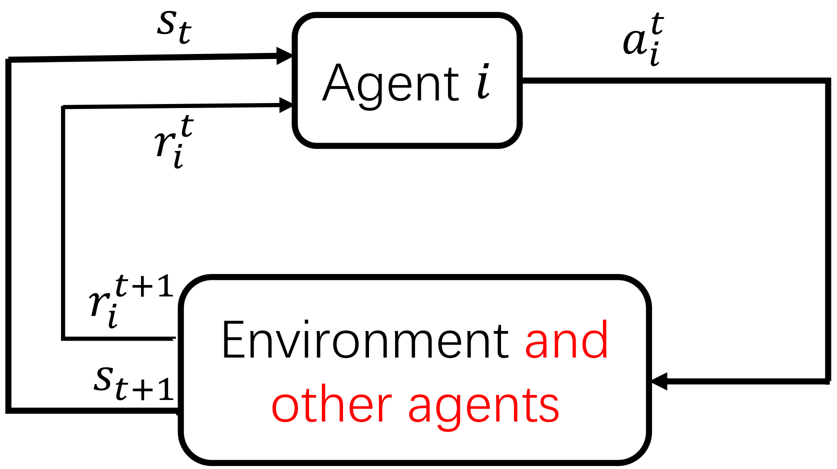
Markov Game

- Markov game describes a process where multiple agents make decisions in a random environment
- A Markov game with N agents includes:
  - A set of states: $s \in S$, Joint action: $a_1, \cdots, a_N$
  - Transition probability function: $p(s' | s, a_1, \cdots, a_N)$
  - Reward function of each agent: $r_i(s, a_1, \cdots, a_N)$
- Objective
  - To find the optimal policies: $\{\pi_i^*\}_{i=1}^N$ that can maximize each agent's cumulative discounted rewards:

$$E[\sum_{\tau=0}^{T} \gamma^\tau r_i(s^{t+\tau}, a_1^{t+\tau}, \cdots, a_N^{t+\tau})]$$



Independent deep Q-learning



- Independent learner and controller
- Discrete action space
- Action-value function of agent i:

$$Q_i^{\pi_i}(s, a_i) = E[\sum_{\tau=0}^{T} \gamma^\tau r_i^{t+\tau} | s^t = s, a_i^t = a_i, \pi_i]$$

- Optimal action-value function: $Q_i^*(s, a_i) = max_{\pi_i} Q_i^{\pi_i}(s, a_i)$
- Each agent learns a greedy policy: $a_i^t = argmax_{a_i} Q_i^*(s^t, a_i)$

Independent deep Q-learning

- Learn the optimal action-value function based on the Bellman optimality equation:

y_i^t, target value for learning $Q_i^*(s^t, a_i^t)$.

Loss function: $L(\theta_i) = E_{s^t, a_i^t}[(y_i^t - Q_i(s^t, a_i^t; \theta_i))^2]$

$$Q_i^*(s^t, a_i^t) = \boxed{\sum_{s^{t+1}} p(s^{t+1}|s^t, a_i^t)\left[r_i^{t+1} + \gamma \max_{a_i^{t+1}} Q^*(s^{t+1}, a_i^{t+1})\right]}$$

Experience replay method:
Store transitions $(s^t, a_i^t, r_i^{t+1}, s^{t+1})$ in a replay buffer.
Randomly sample a minibatch to calculate the gradient of the loss function.

Independent deep Q-learning

- Learn the optimal action-value function based on the Bellman optimality equation:

y_i^t, target value for learning $Q_i^*(s^t, a_i^t)$.
Loss function:$L(\theta_i) = E_{s^t, a_i^t}[(y_i^t - Q_i(s^t, a_i^t; \theta_i))^2]$

$$Q_i^*(s^t, a_i^t) = \boxed{\sum_{s^{t+1}} p(s^{t+1}|s^t, a_i^t)\left[r_i^{t+1} + \gamma \max_{a_i^{t+1}} Q^*(s^{t+1}, a_i^{t+1})\right]}$$

$$= \sum_{s^{t+1}} \sum_{\boldsymbol{a}_{-i}^t} p(s^{t+1}|s^t, a_i^t, \boldsymbol{a}_{-i}^t)\boxed{p(\boldsymbol{a}_{-i}^t|s^t)}\left[r_i^{t+1} + \gamma \max_{a_i^{t+1}} Q_i^*(s^{t+1}, a_i^{t+1})\right]$$

Non-stationary component

Fingerprint-based method

- Augmenting the input of the action-value function with low-dimensional fingerprints correlated with the changes of other agents' policies.

$$Q_i^*(s^t, a_i^t) = \sum_{s^{t+1}} \sum_{\boldsymbol{a}_{-i}^t} p(s^{t+1}|s^t, a_i^t, \boldsymbol{a}_{-i}^t) p(\boldsymbol{a}_{-i}^t|s^t) \left[r_i^{t+1} + \gamma \max_{a_i^{t+1}} Q_i^*(s^{t+1}, a_i^{t+1}) \right]$$

$s^t, \boldsymbol{x}$

$s^t, \boldsymbol{x}$

- Training iteration number
- The rate of exploration (the value of ε in ε-greedy policy)

- Essentially, owing to being augmented with the fingerprint, state-action pairs stored in the past training iterations are outdated in the current training iteration. Using them to update parameters of the action-value function is meaningless and time-wasting.

Stabilizing MARL

- Take $\boldsymbol{a}_{-i}$ into account to evaluate a modified action-value function as:

  $Q_i^{\pi_i}(\boxed{s, \boldsymbol{a}_{-i},} a_i)$

- Bellman optimality equation: $Q_i^*(s^t, \boldsymbol{a}_{-i}^t, a_i^t) = \sum_{s^{t+1}, \boldsymbol{a}_{-i}^{t+1}} p(s^{t+1}, \boldsymbol{a}_{-i}^{t+1} | s^t, \boldsymbol{a}_{-i}^t, a_i^t)$

  $$\left[r_i^{t+1} + \gamma \max_{a_i^{t+1}} Q_i^*(s^{t+1}, \boldsymbol{a}_{-i}^{t+1}, a_i^{t+1}) \right]$$

- Define an action estimation as:

  $\widehat{\boldsymbol{a}_{-i}^t} = f(s_{-i}^t, s_{-i}^{t+1})$

  - Other agents' states at adjacent time steps can partially reveal their actions
  - f: a function to be learned
  - Stochastic environments $p(s_{-i}^{t+1} | s_{-i}^t, \boldsymbol{a}_{-i}^t)$
  - Assume that $f(s_{-i}^t,\ s_{-i}^{t+1})$ is an unbiased estimate of $\boldsymbol{a}_{-i}^t$

$$\boldsymbol{a}_{-i}^t = E_{s_{-i}^{t+1} | s_{-i}^t, a_{-i}^t} f(s_{-i}^t, s_{-i}^{t+1})$$

Stabilizing MARL

- Derive the Bellman optimality equation to learn the modified action-value function

$$Q_i^*(s^t, \boldsymbol{a}_{-i}^t, a_i^t) = \sum_{s^{t+1}, \boldsymbol{a}_{-i}^{t+1}} p(s^{t+1}, \boldsymbol{a}_{-i}^{t+1} | s^t, \boldsymbol{a}_{-i}^t, a_i^t) \left[r_i^{t+1} + \gamma \max_{a_i^{t+1}} Q_i^*(s^{t+1}, \boldsymbol{a}_{-i}^{t+1}, a_i^{t+1}) \right]$$

$$Q_i^*(s^t, \mathbb{E}_{s_{-i}^{t+1} | s_{-i}^t, \boldsymbol{a}_{-i}^t} f(s_{-i}^t, s_{-i}^{t+1}), a_i^t) \approx \sum_{s^{t+1}} p(s^{t+1} | s^t, \boldsymbol{a}_{-i}^t, a_i^t) \left[r_i^{t+1} + \gamma \max_{a_i^{t+1}} Q_i^*(s^{t+1}, f(s_{-i}^t, s_{-i}^{t+1}), a_i^{t+1}) \right]$$

Use the last action estimate: $\widehat{\boldsymbol{a}_{-i}^t} = f(s_{-i}^t, s_{-i}^{t+1})$

- The optimal policy has continuity property
- Looking only one-step ahead can result in more stable learning

Stabilizing MARL

$$Q_i^*(s^t, \underline{\mathbb{E}_{s_{-i}^{t+1}|s_{-i}^t, \boldsymbol{a}_{-i}^t}} f(s_{-i}^t, s_{-i}^{t+1}), a_i^t) \approx \sum_{s^{t+1}} p(s^{t+1}|s^t, \boldsymbol{a}_{-i}^t, a_i^t) \left[r_i^{t+1} + \gamma \max_{a_i^{t+1}} Q_i^*(s^{t+1}, f(s_{-i}^t, s_{-i}^{t+1}), a_i^{t+1}) \right]$$

- Assume that
  - Given s_{-i}^t and a_{-i}^t, $f(s_{-i}^t, s_{-i}^{t+1})$ does not change much
  - Q_i^* is locally linear

$$\mathbb{E}_{s_{-i}^{t+1}|s_{-i}^t, \boldsymbol{a}_{-i}^t} Q_i^*(s^t, f(s_{-i}^t, s_{-i}^{t+1}), a_i^t) \approx \mathbb{E}_{s^{t+1}|s^t, a_i^t, \boldsymbol{a}_{-i}^t} \left[r_i^{t+1} + \gamma \max_{a_i^{t+1}} Q_i^*(s^{t+1}, f(s_{-i}^t, s_{-i}^{t+1}), a_i^{t+1}) \right]$$

- Learn a composite function incorporating the action estimation function:

$$G_i^{\pi_i}(s, s_{-i}, s'_{-i}, a_i) = Q_i^{\pi_i}(s, f(s_{-i}, s'_{-i}), a_i)$$

Stabilizing MARL

- Learn G-function:
  - The approximate Bellman optimality equation for G_i:

$$\mathbb{E}_{s^{t+1}_{-i} | s^t_{-i}, \boldsymbol{a}^t_{-i}} G_i^*(s^t, s^t_{-i}, s^{t+1}_{-i}, a_i^t) \approx$$

$$\mathbb{E}_{s^{t+1} | s^t, a_i^t, \boldsymbol{a}^t_{-i}} \left[\boxed{r_i^{t+1} + \gamma \max_{a_i^{t+1}} G_i^*(s^{t+1}, s^t_{-i}, s^{t+1}_{-i}, a_i^{t+1})} \right]$$

  - Loss function:

$$L(\theta_i) = \mathbb{E}_{s^t, s^{t+1}_{-i}, a_i^t} \left[\left(y_i^t - G_i(s^t, s^t_{-i}, s^{t+1}_{-i}, a_i^t; \theta_i) \right)^2 \right]$$

- Make decisions:

$$a_i^t = \arg\max_{a_i} G_i^*(s^t, s^{t-1}_{-i}, s^t_{-i}, a_i)$$

Simulation settings

- Multi-agent cooperative navigation problem
  - Agents need to cooperate through motions to reach a set of targets with the minimum time consumption
  - Randomly generate positions of targets and agents in every episode
  - Different numbers of targets and agents (N =2; 3; 4; 5; 6)

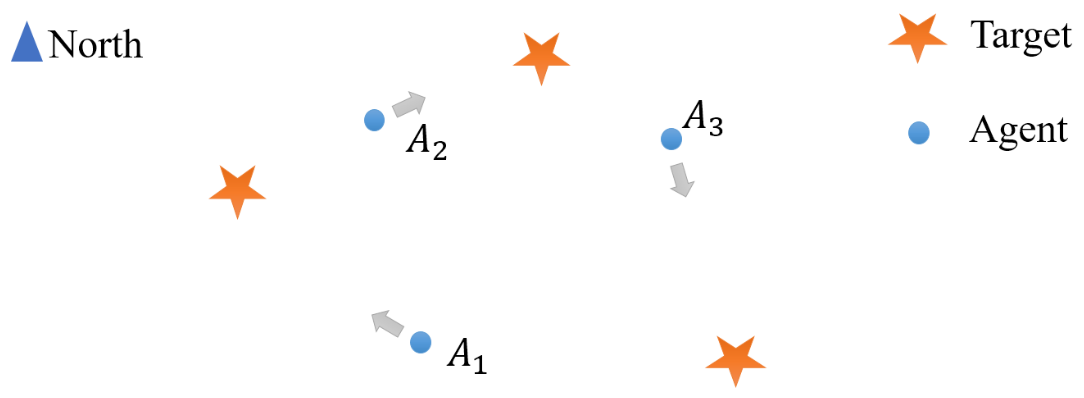


Fig. 1: Illustration of the cooperative navigation task involving three agents

Observation: relative position coordinates of targets and other agents

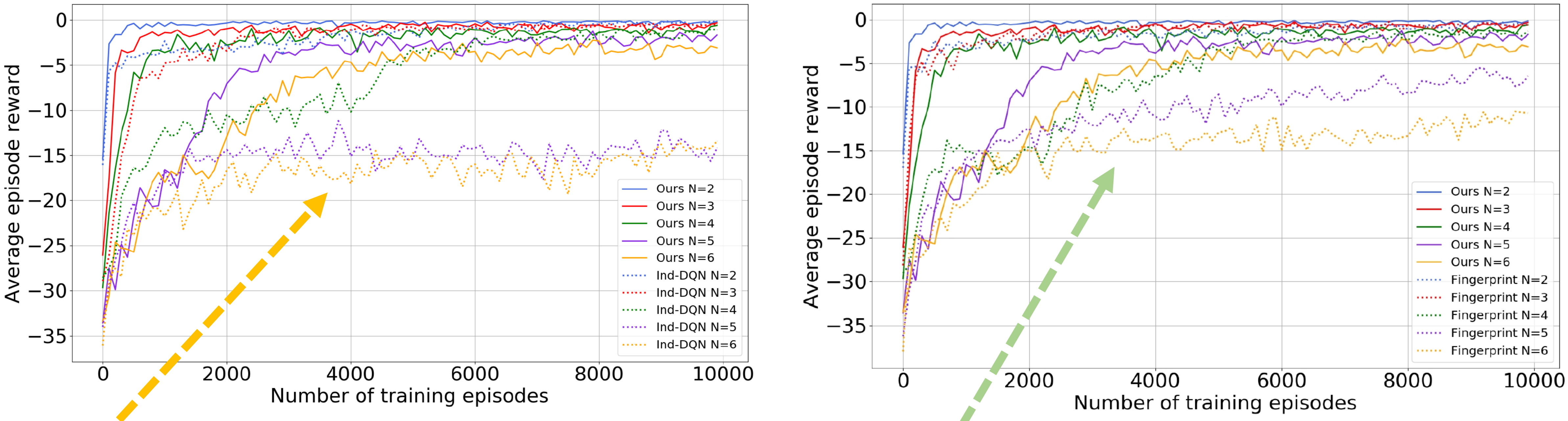
Action: select a target to head for

Assuming a constant speed.

Results

- Training performance

Our method shows better convergence performance
- Faster convergence speed
- Higher reward gain



Independent deep Q-learning

Fingerprint-based method

Results

- Testing performance
  - One thousand randomly generated tasks

Success:
Agents successfully arrive at different targets

Maximum navigation time:
The time cost by the agent who is the last one to arrive at a target

N	Success rate		
	Ours	Ind-DQN	Fingerprint
2	99.9%	99.6%	99.6%
3	98.2%	97.7%	98.3%
4	97.8%	94.7%	96.3%
5	96.1%	4.0%	66.1%
6	91.2%	2.6%	18.3%

Normalized average maximum time		
Ours	Ind-DQN	Fingerprint
0.517	0.516	0.521
0.537	0.541	0.548
0.560	0.589	0.563
0.581	0.603	0.585
0.589	0.613	0.596

4. CONCLUSION

We present
a novel method to stabilize multi-agent DRL, which learns a modified action-value function incorporating implicit estimate of other agents' actions to stabilize agents' policy learning and improve learning efficiency.

We prove that
by incorporating the estimation function into the action-value function,
each agent can learn a policy in an approximate stationary environment.

Empirical results show that
compared with independent deep Q-learning and the fingerprint-based method,
our method significantly improves the convergence speed and policy performance.

Thank you for your attention!