

# Multi-Agent Reinforcement Learning

Roberto Capobianco



SAPIENZA  
UNIVERSITÀ DI ROMA

# Recap

# Imitation Learning

— — —

Given an MDP, what happens if we cannot get access to a reward?

We can learn by imitation of an expert!

1. Collect expert demonstrations

$$D = \{s_i^*, a_i^*\}_{i=1}^M \sim d^{\pi^*}$$

For simplicity, let's assume expert is a (nearly) optimal policy  $\pi^*$

# Imitation Learning

— — —

Given an MDP, what happens if we cannot get access to a reward?

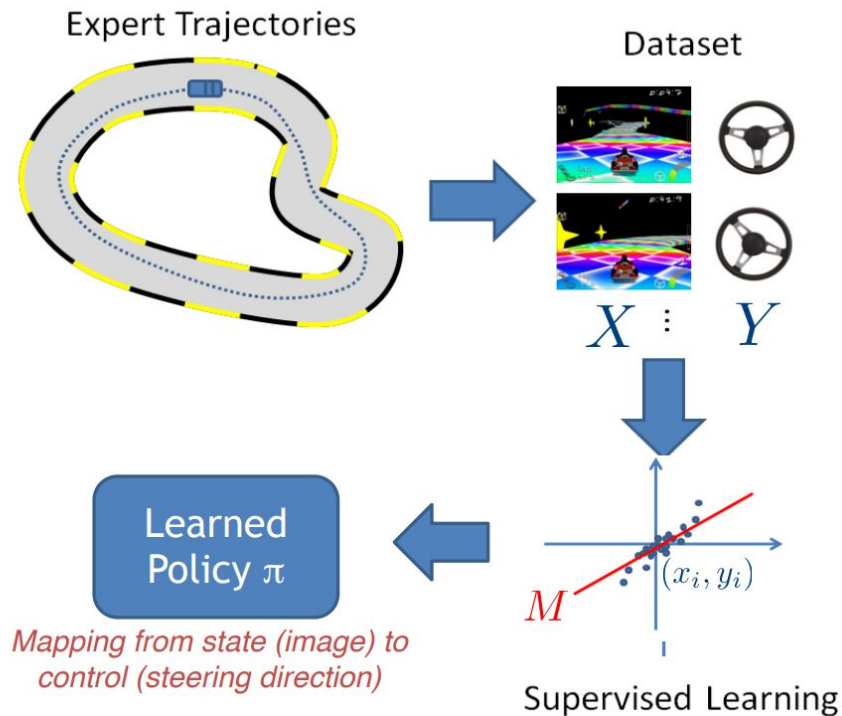
We can learn by imitation of an expert!

1. Collect expert demonstrations
2. Use a machine learning algorithm to learn to map states to actions
3. Generate a policy

For simplicity, let's assume expert is a (nearly) optimal policy  $\pi^*$

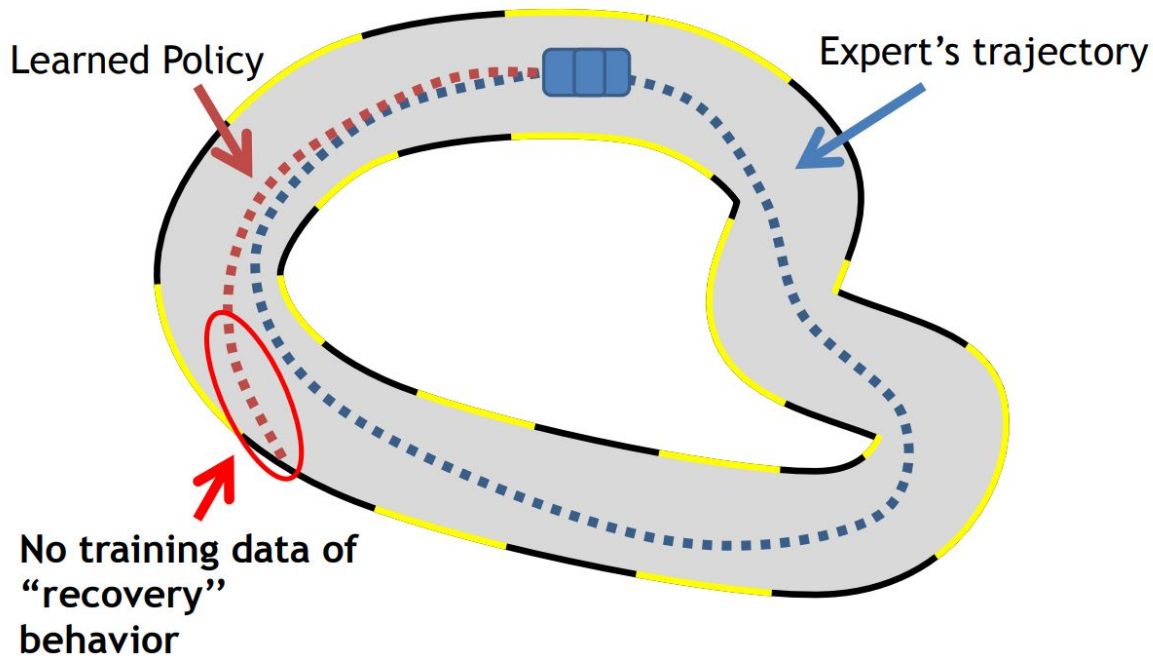
# Behavior Cloning

— — —



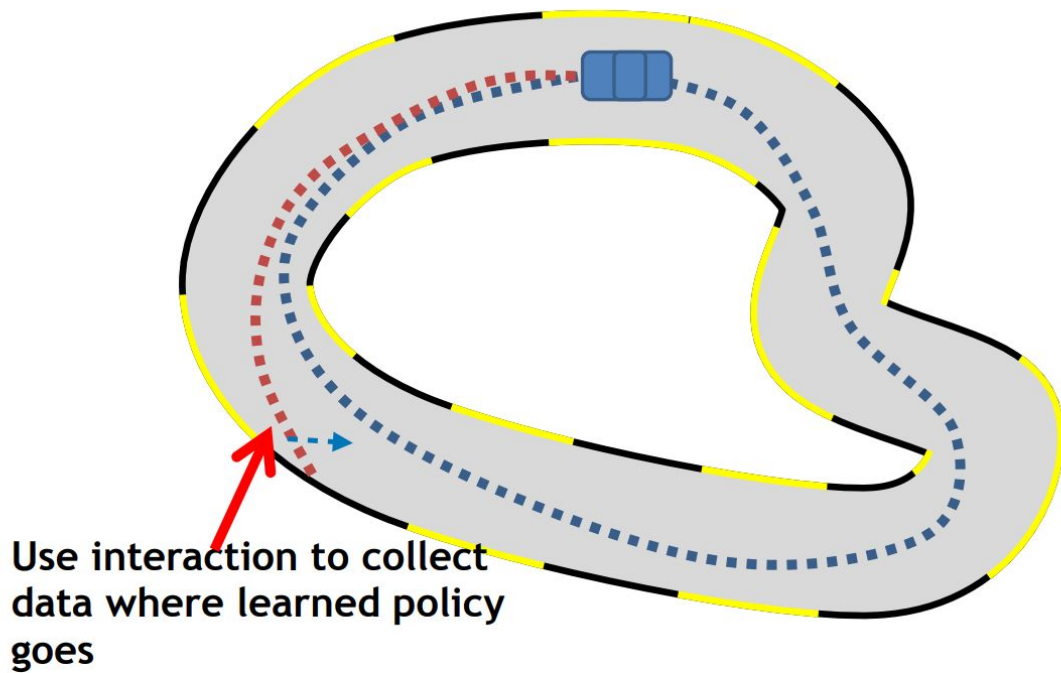
# Behavior Cloning: Distribution Shift

— — —

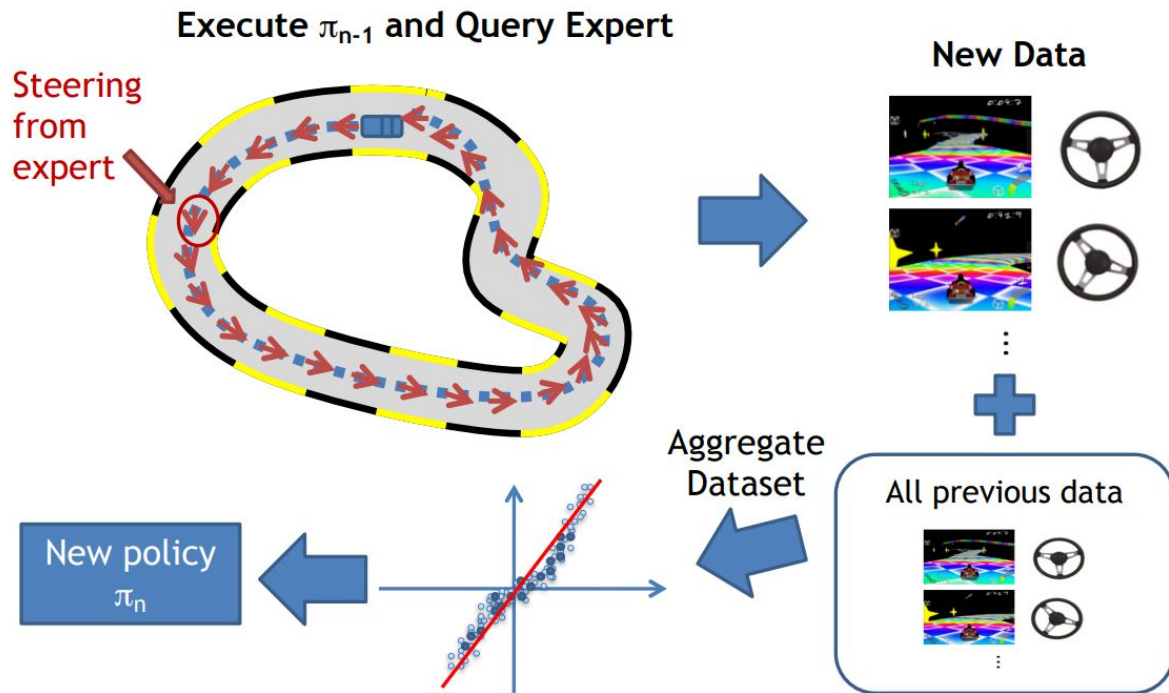


# Interactive Imitation Learning

— — —



# DAgger: Iterations (n-th)





# Inverse Reinforcement Learning

— — —

Given an MDP, what happens if we cannot get access to a reward?

We can learn through an optimal expert that minimizes the true cost!

**Assume transition function is known and we have our dataset  $D$**

**Also assume the true reward/cost is linear in some features  $\phi(s,a)$**

# Maximum Entropy IRL: Algorithm

— — —

Initialize  $w^0 \in \mathbb{R}^d$

For  $t = 0 \rightarrow T - 1$

This is like an RL problem w/ cost  
 $c(s, a) := (w^t)^\top \phi(s, a)$ , but w/ an additional  $\ln \pi(a | s)$

$$\pi^t = \arg \min_{\pi} \mathbb{E}_{s, a \sim d_{\mu}^{\pi}} [(w^t)^\top \phi(s, a) + \ln \pi(a | s)]$$

Uses soft-value iteration  
(# best response:  $\pi^t = \arg \min_{\pi} \ell(\pi, w^t)$ )

$$w^{t+1} = w^t + \eta \left( \mathbb{E}_{s, a \sim d_{\mu}^{\pi^t}} \phi(s, a) - \mathbb{E}_{s, a \sim d_{\mu}^{\pi^*}} \phi(s, a) \right)$$

Return  $\bar{\pi} = \text{Uniform}(\pi^0, \dots, \pi^{T-1})$

(# gradient update:  $w^{t+1} = w^t + \eta \nabla_w \ell(\pi^t, w^t)$ )



# End Recap



SAPIENZA  
UNIVERSITÀ DI ROMA

# Multi-Agent Reinforcement Learning Problem

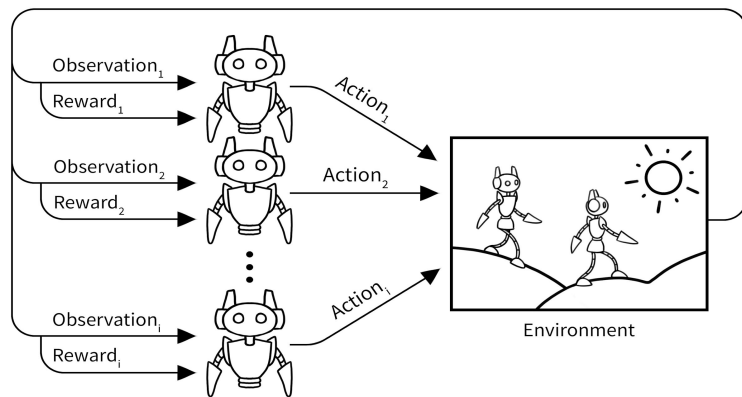
— — —

- Partially observable Markov (or stochastic) Games (POMGs)
- Multi-agent extensions of MDPs of  $N$  agents

A Partially Observable Markov Game is a tuple

$$G = \langle S, \mathcal{A}, \mathcal{T}, \mathcal{R}, \mathcal{Q}, \mathcal{O}, \gamma, N \rangle$$

- $S$  is a finite set of states
- $\mathcal{A}$  is a collection of sets of actions,  $\mathcal{A} = \{\mathcal{A}_1, \dots, \mathcal{A}_N\}$
- $\mathcal{T}$  is a state transition function
- $\mathcal{R}$  is a reward function
- $\mathcal{Q}$  is a collection of private observation functions  $\mathcal{Q} = \{Q_1, \dots, Q_N\}$
- $\mathcal{O}$  is a collection of private observations  $\mathcal{O} = \{\mathcal{O}_1, \dots, \mathcal{O}_N\}$
- $\gamma$  is a discount factor
- $N$  is the number of agents



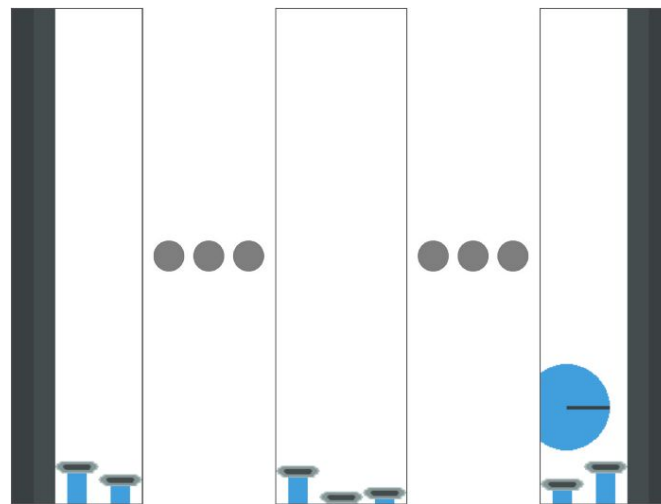
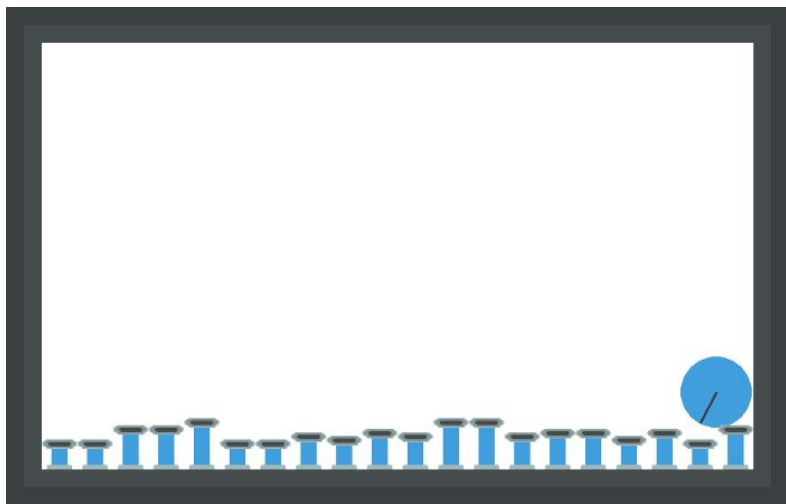
Copyright Justin Terry 2021



# MARL Example

— — —

Pistonball, a cooperative environment from PettingZoo and its observation space



# MARL Challenges (1)

— — —

- Partial observability
  - Agents do not have full access to the true state
  - Each agent receives a private partial observation correlated with the state
  - And chooses an action according to a policy conditioned on its own private observation



# MARL Challenges (2)

— — —

- Non-stationarity
  - Environment moves into the next state according to actions of all agents
  - It is non-stationary from the viewpoint of any agent
- Credit assignment
  - Which agent is responsible for the received reward?



# MARL Challenges (3)

— — —

- Markov assumption is violated (P0 + NS)
- Transitions in the experience replay become invalid (NS)
- High variance problem exacerbates (CA)
- Sample inefficiency exacerbates (P0 + NS + CA)

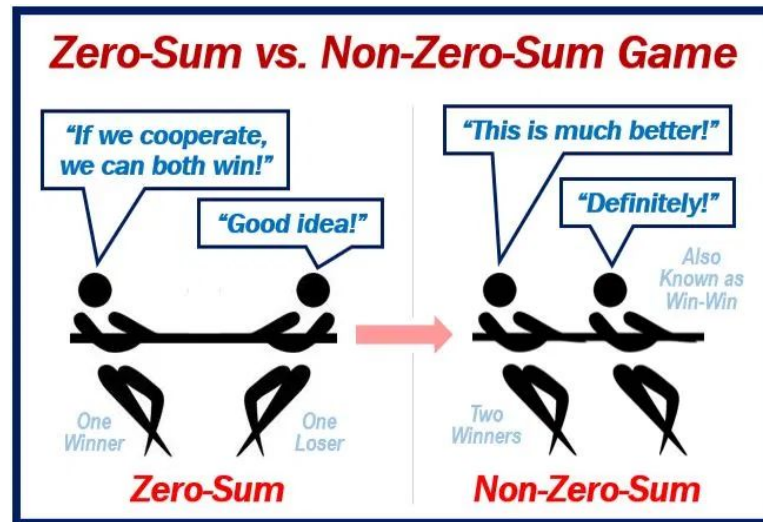




# Settings

— — —

- Cooperative
  - Reward function is the same for all agents working together to optimise the performance of a larger system
  - E.g., urban traffic control, air traffic control and electricity generator scheduling
- Competitive
  - Any win for one agent implies a loss for another (zero-sum)
  - E.g., Backgammon, Go and Starcraft 2
- Mixed
  - No restriction is imposed on the reward function definitions
  - May incorporate elements of both cooperation and competition
  - E.g., RoboCup soccer



# Overall Approaches

— — —

- Independent Learners:

- Ignore all the problems and train/execute agents independently in decentralised manner
- Over-optimistic

- Joint Learners:

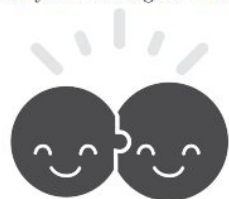
- Use all available information and train/execute agents as a single Meta-agent in centralised manner
- Scalability: input and action size is multiplied by  $N$  for each one of  $N$  agents (grows exponentially)

# Learning to Cooperate

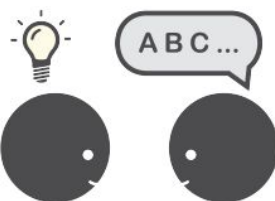
Hernandez-Leal, Pablo, Bilal Kartal, and Matthew E. Taylor. "A survey and critique of multiagent deep reinforcement learning." *Autonomous Agents and Multi-Agent Systems* 33.6 (2019): 750-797.



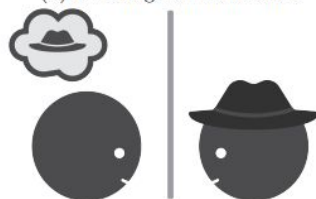
(a) Analysis of emergent behaviors



(c) Learning cooperation



(b) Learning communication



(d) Agents modeling agents

Table 3: These papers aim to *learn cooperation*. Learning type is either value-based (VB) or policy gradient (PG). Setting where experiments were performed: cooperative (CO), competitive (CMP) or mixed. A more detailed description is given in Section 3.5.

Algorithm	Summary	Learning	Setting
Lerer and Peysakhovich [176]	Propose DRL agents able to cooperate in social dilemmas.	VB	Mixed
MD-MADDPG [165]	Use of a shared memory as a means to multiagent communication.	PG	CO
MADDPG-MD [177]	Extend dropout technique to robustify communication when applied in multiagent scenarios with direct communication.	PG	CO
RIAL [162]	Use a single network (parameter sharing) to train agents that take environmental and communication actions.	VB	CO
DIAL [162]	Use gradient sharing during learning and communication actions during execution.	VB	CO
DCH/PSRO [172]	Policies can overfit to opponents: better compute approximate best responses to a mixture of policies.	VB	CO & CMP
Fingerprints [168]	Deal with ER problems in MDRL by conditioning the value function on a fingerprint that disambiguates the age of the sampled data.	VB	CO
Lenient-DQN [35]	Achieve cooperation by leniency, optimism in the value function by forgiving suboptimal (low-rewards) actions.	VB	CO
Hysteretic-DRQN [166]	Achieve cooperation by using two learning rates, depending on the updated values together with multitask learning via policy distillation.	VB	CO
WDDQN [178]	Achieve cooperation by leniency, weighted double estimators, and a modified prioritized experience replay buffer.	VB	CO
FTW [179]	Agents act in a mixed environment (composed of teammates and opponents), it proposes a two-level architecture and population-based learning.	PG	Mixed
VDN [180]	Decompose the team action-value function into pieces across agents, where the pieces can be easily added.	VB	Mixed
QMIX [181]	Decompose the team action-value function together with a mixing network that can recombine them.	VB	Mixed
COMA [167]	Use a centralized critic and a counter-factual advantage function based on solving the multiagent credit assignment.	PG	Mixed
PS-DQN, PS-TRPO, PS-A3C [182]	Propose parameter sharing for learning cooperative tasks.	VB, PG	CO
MADDPG [63]	Use an actor-critic approach where the critic is augmented with information from other agents, the actions of all agents.	PG	Mixed



# Fingerprints

— — —

Foerster, Jakob, et al. "Stabilising experience replay for deep multi-agent reinforcement learning." *International conference on machine learning*. PMLR, 2017. - Code: [https://github.com/cts198859/deepri\\_network](https://github.com/cts198859/deepri_network)

- **Problem:**

- Independent Q-Learning (DQN) agents
- Experience replay needs to be fixed
  - Dynamics that generated the data in the ER no longer reflects current dynamics: experience is obsolete

- **Idea:** disambiguate the age of the sampled data from the replay memory



# Fingerprints - Multi-Agent Importance Sampling

— — —

Foerster, Jakob, et al. "Stabilising experience replay for deep multi-agent reinforcement learning." *International conference on machine learning*. PMLR, 2017. - Code: [https://github.com/cts198859/deepri\\_network](https://github.com/cts198859/deepri_network)

- **Off-environment** learning (similar to IS in off-policy)
- We know how the environment (as a result of the policies) is changing and we account for that in the ER by storing

$$\langle s, u_a, r, \pi[\mathbf{u}_{-a}|s], s' \rangle^{(t_c)}$$

joint quantities over agents  
other than  $a$

- We compute a ratio depending on  $\pi_i$  at storage time and  $\pi_i$  at sampling time

# Fingerprints - Multi-Agent Importance Sampling

— — —

Foerster, Jakob, et al. "Stabilising experience replay for deep multi-agent reinforcement learning." *International conference on machine learning*. PMLR, 2017. - Code: [https://github.com/cts198859/deepri\\_network](https://github.com/cts198859/deepri_network)

We compute a ratio depending on  $\pi$  at storage time and  $\pi$  at sampling time

$$\sum_{i=1}^b \frac{\pi_{-a}^{t_r}(\mathbf{u}_{-a}|s)}{\pi_{-a}^{t_i}(\mathbf{u}_{-a}|s)} [(y_i^{DQN} - Q(s, u; \theta))^2]$$

replay time  
collection time



# Fingerprints - Multi-Agent Fingerprints

— — —

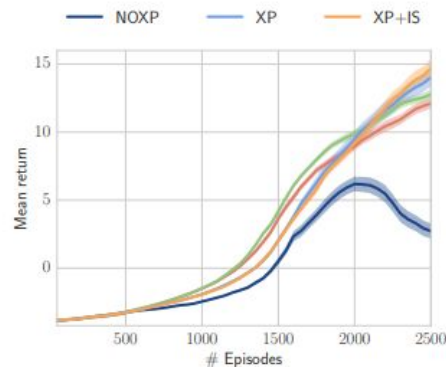
Foerster, Jakob, et al. "Stabilising experience replay for deep multi-agent reinforcement learning." *International conference on machine learning*. PMLR, 2017. - Code: [https://github.com/cts198859/deepri\\_network](https://github.com/cts198859/deepri_network)

- Importance ratios have high variance
- Agent's observations should disambiguate where along the evolution of policies the current training sample originated:
  - Epsilon is correlated with performance (for exploration)
  - Training iteration  $e$  is also relevant
  - Use that as a compact 'fingerprint' representation  $\{O(s), \epsilon, e\}$

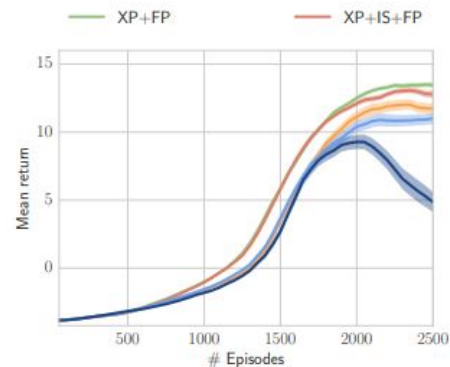


# Fingerprints - Results

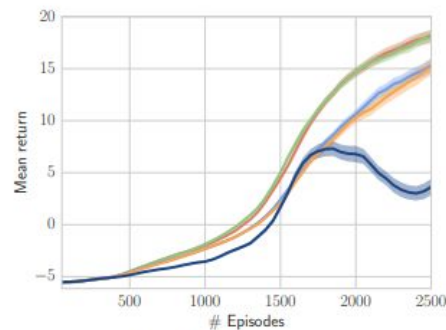
— — —



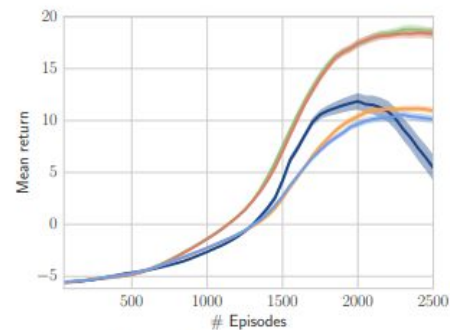
(a) 3v3 with recurrent networks



(b) 3v3 with feed-forward networks



(c) 5v5 with recurrent networks



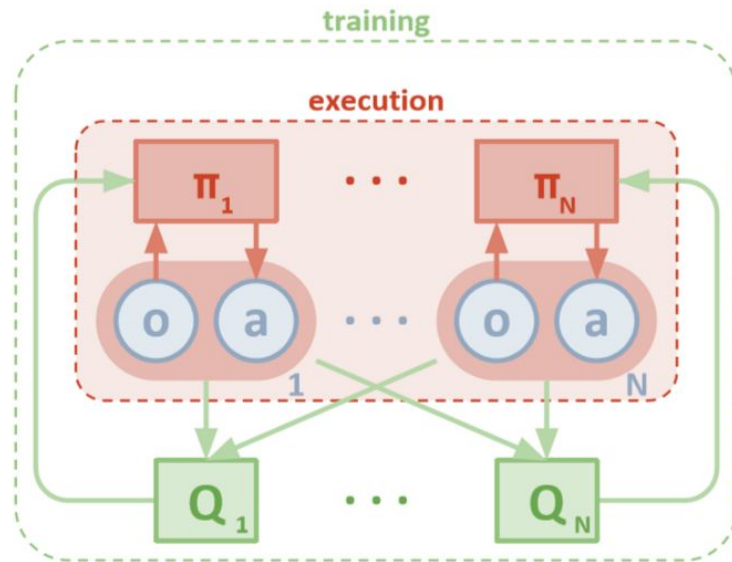
(d) 5v5 with feed-forward networks



# Multi-Agent DDPG

Lowe, Ryan, et al. "Multi-agent actor-critic for mixed cooperative-competitive environments." *arXiv preprint arXiv:1706.02275* (2017). - Code: <https://github.com/openai/maddpg>

- Every agent has:
  - an observation space and continuous action space
  - an actor-network using local observations
  - a target actor-network with identical functionality for training stability
  - a critic-network that uses joint states action pairs
- Q-networks can be shared
- Reduce the variance by removing the non-stationarity caused by the concurrently learning agents



# Multi-Agent DDPG - Learning

— — —

Lowe, Ryan, et al. "Multi-agent actor-critic for mixed cooperative-competitive environments." *arXiv preprint arXiv:1706.02275* (2017). - Code: <https://github.com/openai/maddpg>

- Experience replay stores:  $(\mathbf{x}, \mathbf{x}', a_1, \dots, a_N, r_1, \dots, r_N)$
- Q-networks learning

$$\mathcal{L}(\theta_i) = \mathbb{E}_{\mathbf{x}, a, r, \mathbf{x}'} [(Q_i^{\mu}(\mathbf{x}, a_1, \dots, a_N) - y)^2], \quad y = r_i + \gamma Q_i^{\mu'}(\mathbf{x}', a'_1, \dots, a'_N) \big|_{a'_j = \mu'_j(o_j)}$$

- Policy learning

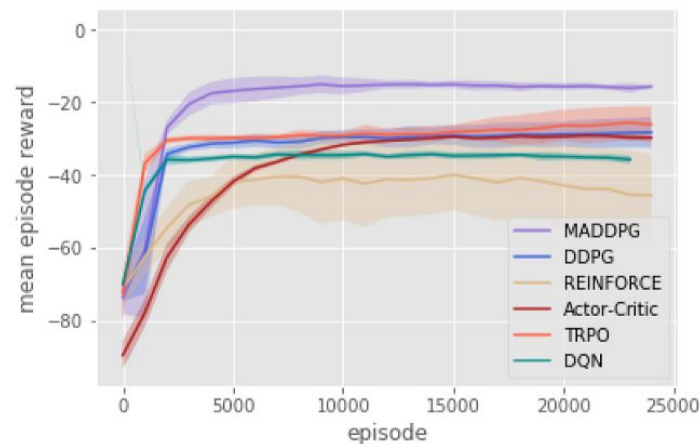
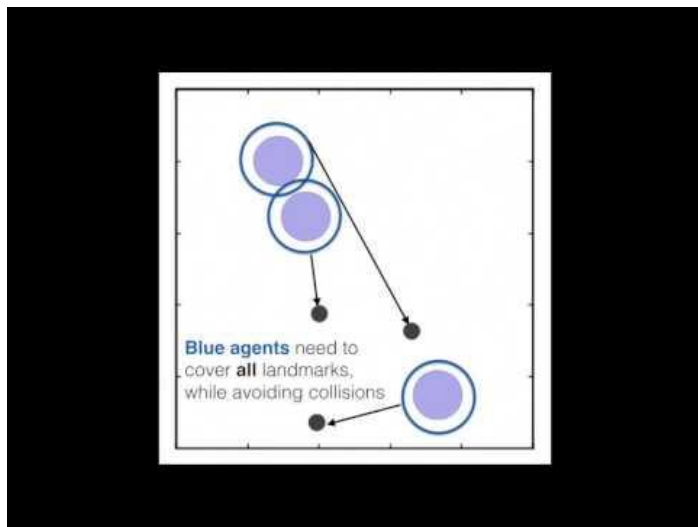
$$\nabla_{\theta_i} J(\mu_i) = \mathbb{E}_{\mathbf{x}, a \sim \mathcal{D}} [\nabla_{\theta_i} \mu_i(a_i | o_i) \nabla_{a_i} Q_i^{\mu}(\mathbf{x}, a_1, \dots, a_N) \big|_{a_i = \mu_i(o_i)}]$$



# Multi-Agent DDPG - Results

— — —

Lowe, Ryan, et al. "Multi-agent actor-critic for mixed cooperative-competitive environments." *arXiv preprint arXiv:1706.02275* (2017). - Code: <https://github.com/openai/maddpg>



# COMA

— — —

Foerster, Jakob, et al. "Counterfactual multi-agent policy gradients." *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 32. No. 1. 2018. -  
Code: <https://github.com/TonghanWang/NDQ>

- Designed for the fully centralized setting
  - Does not suffer non-stationarity but scalability
  - Estimates Q-values for the joint action conditioned on the central state
- Attempts to solve the multi-agent credit assignment problem

# COMA

— — —

Foerster, Jakob, et al. "Counterfactual multi-agent policy gradients." *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 32. No. 1. 2018. -  
Code: <https://github.com/TonghanWang/NDQ>

- Compute a counterfactual baseline
  - Marginalize over the actions of the agent
  - Keep the rest of the other agents' actions fixed
- Compute advantage function and compare the current Q value to the counterfactual

*...what would it be if I did this instead of that...*

$$A^a(s, \mathbf{u}) = Q(s, \mathbf{u}) - \sum_{u'^a} \pi^a(u'^a | \tau^a) Q(s, (\mathbf{u}^{-a}, u'^a))$$

Any action by agent a that  
improves such term also improves  
the true global reward



# COMA - Results

— — —

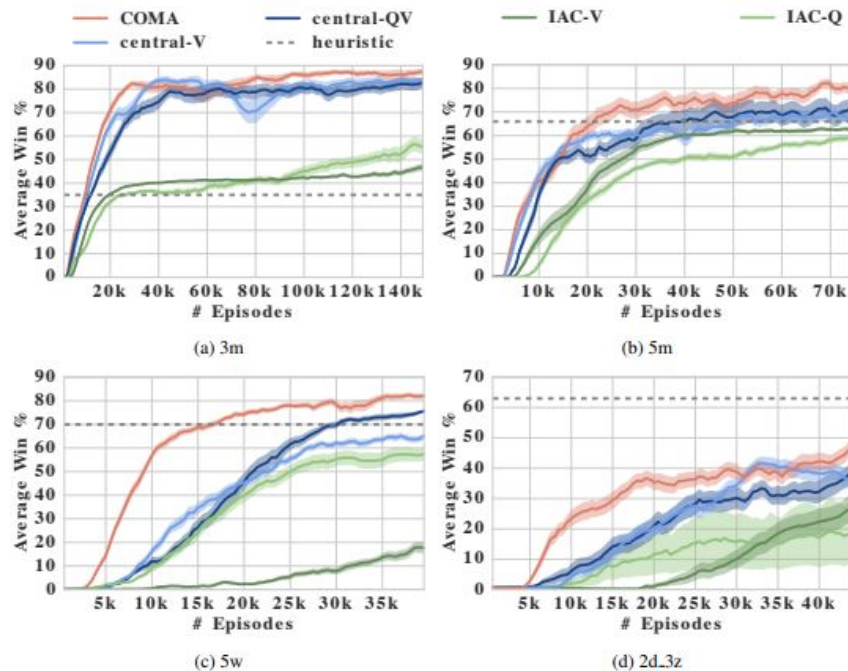


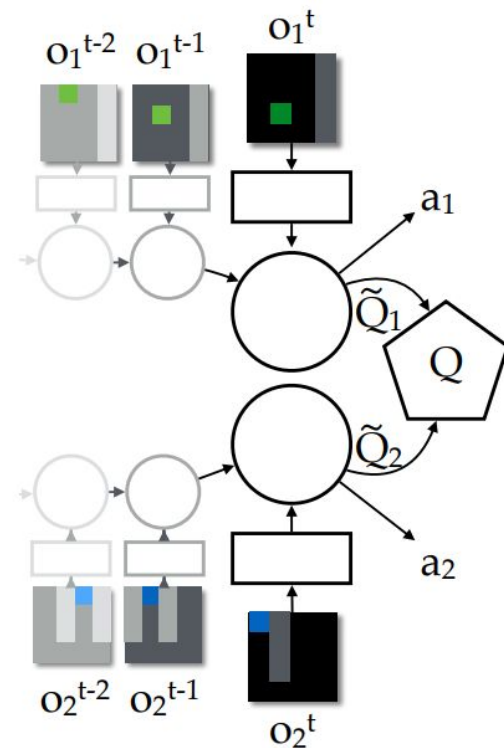
Figure 3: Win rates for COMA and competing algorithms on four different scenarios. COMA outperforms all baseline methods. Centralised critics also clearly outperform their decentralised counterparts. The legend at the top applies across all plots.

# Value Decomposition Networks

— — —  
Suneag, Peter, et al. "Value-decomposition networks for cooperative multi-agent learning." *arXiv preprint arXiv:1706.05296* (2017). - <https://github.com/TonghanWang/NDQ>

- Decomposes a team value function into an additive decomposition of the individual value functions
- Learns a joint action-value function  $Q_{tot}(\tau, a)$  represented by the sum of individual value functions  $Q_i(\tau_i, a_i; \theta_i)$

$$Q_{tot}(\tau, a) = \sum_{i=1}^N Q_i(\tau^i, a^i; \theta^i)$$



# Emergent Behaviors

Table 1: These papers analyze *emergent behaviors* in MDRL. Learning type is either value-based (VB) or policy gradient (PG). Setting where experiments were performed: cooperative (CO), competitive (CMP) or mixed. A detailed description is given in Section 3.3.

Work	Summary	Learning	Setting
Tampuu et al. [155]	Train DQN agents to play Pong.	VB	CO&CMP
Leibo et al. [156]	Train DQN agents to play sequential social dilemmas.	VB	Mixed
Lerer and Peysakhovich [176]	Propose DRL agents able to cooperate in social dilemmas.	VB	Mixed
Leibo et al. [159]	Propose Malthusian reinforcement learning which extends self-play to population dynamics.	VB	Mixed
Bansal et al. [158]	Train PPO agents in competitive MuJoCo scenarios.	PG	CMP
Raghu et al. [157]	Train PPO, A3C, and DQN agents in attacker-defender games.	VB, PG	CMP
Lazaridou et al. [161]	Train agents represented with NN to learn a communication language.	PG	CO
Mordatch and Abbeel [160]	Learn communication with an end-to-end differentiable model to train with backpropagation.	PG	CO



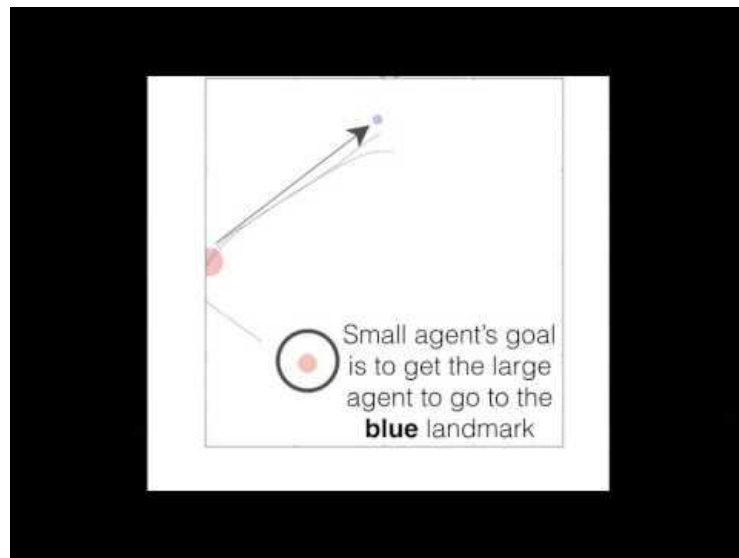


# Learn Communication

— — —

Mordatch, Igor, and Pieter Abbeel. "Emergence of grounded compositional language in multi-agent populations." *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 32. No. 1. 2018.

- **Goal:** design environment where language is naturally emergent:
  - Actions are both motion events and the emission of symbols at each step
  - Reward is specified as a joint reward function over the entire set of agents (must cooperate)
  - Goals are hidden state - not every agent knows about their own goals
  - Agents can communicate
  - No special meaning is assigned to each symbol



# Modeling Agents

Table 4: These papers consider *agents modeling agents*. Learning type is either value-based (VB) or policy gradient (PG). Setting where experiments were performed: cooperative (CO), competitive (CMP) or mixed. A more detailed description is given in Section 3.6.

Algorithm	Summary	Learning	Setting
MADDPG [63]	Use an actor-critic approach where the critic is augmented with information from other agents, the actions of all agents.	PG	Mixed
DRON [169]	Have a network to infer the opponent behavior together with the standard DQN architecture.	VB	Mixed
DPIQN, DPIRQN [171]	Learn policy features from raw observations that represent high-level opponent behaviors via auxiliary tasks.	VB	Mixed
SOM [170]	Assume the reward function depends on a hidden goal of both agents and then use an agent's own policy to infer the goal of the other agent.	PG	Mixed
NFSP [173]	Compute approximate Nash equilibria via self-play and two neural networks.	VB	CMP
PSRO/DCH [172]	Policies can overfit to opponents: better compute approximate best responses to a mixture of policies.	PG	CO & CMP
M3DDPG [183]	Extend MADDPG with minimax objective to robustify the learned policy.	PG	Mixed
LOLA [64]	Use a learning rule where the agent accounts for the parameter update of other agents to maximize its own reward.	PG	Mixed
ToMnet [174]	Use an architecture for end-to-end learning and inference of diverse opponent types.	PG	Mixed
Deep Bayes-ToMoP [175]	Best respond to opponents using Bayesian policy reuse, theory of mind, and deep networks.	VB	CMP
Deep BPR+[184]	Bayesian policy reuse and policy distillation to quickly best respond to opponents.	VB	CO & CMP

# LOLA



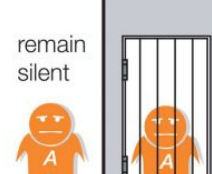

Foerster, Jakob N., et al. "Learning with opponent-learning awareness." *arXiv preprint arXiv:1709.04326* (2017). - Code: <https://github.com/alshedivat/lola/>

- Includes an extra-term
  - Describes how one's value changes depending on the other

$$\left( \frac{\partial V^1(\theta_i^1, \theta_i^2)}{\partial \theta_i^2} \right)^T \frac{\partial^2 V^2(\theta_i^1, \theta_i^2)}{\partial \theta_i^1 \partial \theta_i^2} \cdot \delta \eta,$$

- Opponent's value can be estimated from samples

Prisoners' dilemma

		prisoner B	
		confess (Defect)	remain silent (Cooperate)
prisoner A	confess (Defect)	 2 years   2 years	 0 years   3 years
	remain silent (Cooperate)	 3 years   0 years	 1 year   1 year

© 2010 Encyclopædia Britannica, Inc.



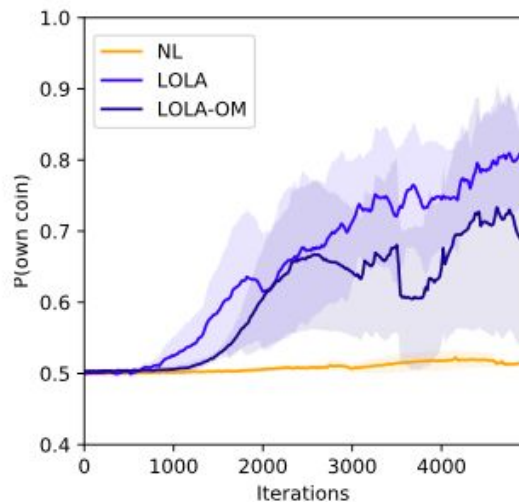
SAPIENZA  
UNIVERSITÀ DI ROMA

# LOLA - Results

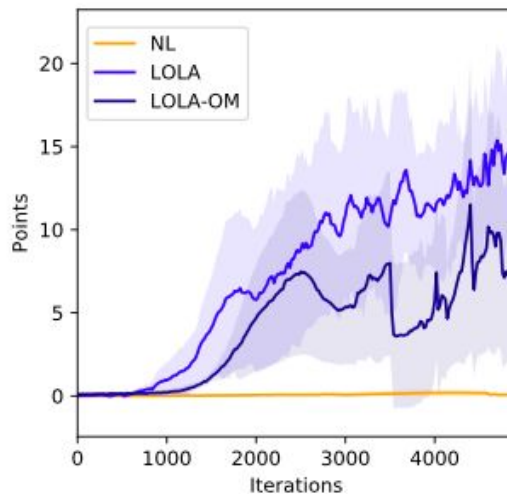
— — —

[https://cdn.openai.com/LOLA/LOLA\\_annotated5.mp4](https://cdn.openai.com/LOLA/LOLA_annotated5.mp4)

[https://cdn.openai.com/LOLA/Naive\\_learner2.mp4](https://cdn.openai.com/LOLA/Naive_learner2.mp4)



(a)



(b)



# Imperfect Information Games

— — —

- Imperfect information:
  - Players are unaware of other players' actions
  - They know who the other players are
  - They know possible what the possible strategies/actions are as well as payoffs
  - E.g., card games like Poker



# Approaches to Imperfect Information Games

— — —

- **Deepstack** [Moravcik, Matej; Schmid, Martin; Burch, Neil; Lisy, Viliam; Morrill, Dustin; Bard, Nolan; Davis, Trevor; Waugh, Kevin; Johanson, Michael; Bowling, Michael (2017). "Deepstack: Expert-level artificial intelligence in heads-up no-limit poker" (PDF). *Science*. 356 (6337): 508–513. arXiv:1701.01724]
- **Libratus** [Brown, Noam, and Tuomas Sandholm. "Superhuman AI for heads-up no-limit poker: Libratus beats top professionals." *Science* 359.6374 (2018): 418–424.]
- **Pluribus** [Brown, Noam, and Tuomas Sandholm. "Superhuman AI for multiplayer poker." *Science* 365.6456 (2019): 885–890.]



# Libratus

— — —



SAPIENZA  
UNIVERSITÀ DI ROMA