Counterfactual Multi-Agent Policy Gradients

Jae Seong Hong MS/Ph.D Combined Student

Department of Biomedical Systems Informatics
College of Medicine
Yonsei University

The Thirty-Second AAAI Conference
on Artificial Intelligence (AAAI-18)

Counterfactual Multi-Agent Policy Gradients

Jakob N. Foerster*
University of Oxford, United Kingdom
jakob.foerster@cs.ox.ac.uk

Gregory Farquhar[†]
University of Oxford, United Kingdom
gregory.farquhar@cs.ox.ac.uk

Triantafyllos Afouras
University of Oxford, UK
afourast@robots.ox.ac.uk

Nantas Nardelli
University of Oxford, UK
nantas@robots.ox.ac.uk

Shimon Whiteson
University of Oxford, UK
shimon.whiteson@cs.ox.ac.uk

https://ojs.aaai.org › index.php › AAAI › article › view

Counterfactual Multi-Agent Policy Gradients

J Foerster 저술 · 2018 · 1315회 인용 — To this end, we propose a

Contents

1. Introduction & Background

Basic Backgrounds

Single Agent Policy Gradient

Expected Discounted Total Reward $J = \mathbb{E}_{\pi}[R_0]$

REINFORCE policy gradient $g = \mathbb{E}_{s_{0:\infty},u_{0:\infty}}[\sum_{t=0}^{T} R_t \nabla_{\theta^{\pi}} \log \pi(u_t|s_t)]$

Actor-critic approaches

Actor, i.e., the policy, is trained by following a gradient that depends on a critic, estimates a value function.

Advantage Function

$A(s_t, u_t) = R_t = Q(s_t, u_t) - b(s_t)$: Reward function with baseline to reduce variance

$b(s_t) = V(s_t)$: Common Choice

Temporal Difference (TD) error

$r_t + \gamma V(s_{t_1}) - V(s)$: Unbiased Estimate of $A(s_t, u_t)$

1. Introduction & Background

Basic Backgrounds

Reward Functions

$R_t = \sum_{l=0}^{\infty} \gamma^l r_{(t+l)}$: discounted return

$V^{\pi}(s_t) = \mathbb{E}_{s_{t+1:\infty}, u_{t:\infty}}[R_t | s_t]$: State-Value Function

$Q^{\pi}(s_t) = \mathbb{E}_{s_{t+1:\infty}, u_{t+1:\infty}}[R_t | s_t, u_t]$: Action-Value Function

$A^{\pi}(s_t, u_t) = Q^{\pi}(s_t, u_t) - V^{\pi}(s_t)$: Advantage Function

1. Introduction & Background

Basic Backgrounds

Train critics $\boldsymbol{f^c(\cdot, \theta^c)}$

Adapt TD(λ)

- Mixture of n-step returns $G_t^{(n)} = \sum_{l=1}^{n} \gamma^{l-1} r_{t+1} + \gamma^n f^c(\cdot_{t+n}, \theta^c)$
    - θ^c : critic parameter
        - Updated by minibatch gradient descent to minimize the loss

Loss Function

$$\mathcal{L}_t(\theta^c) = \left(y^{(\lambda)} - f^c(\cdot_t, \theta^c)\right)^2$$

$$y^{(\lambda)} = (1 - \lambda) \sum_{n=1}^{\infty} \lambda^{n-1} G_t^{(n)}$$

$G_t^{(n)}$: calculated with bootstrapped values estimated by a target network with parameters copied periodically from θ^c

1. Introduction & Background

Basic Backgrounds

Joint-Action

a : agent $(1, 2, \ldots n)$

$P(\mathbb{u}|s_t) = P(u^1|s^1) \cdot \ldots \cdot P(u^n|s^n)$: joint-action

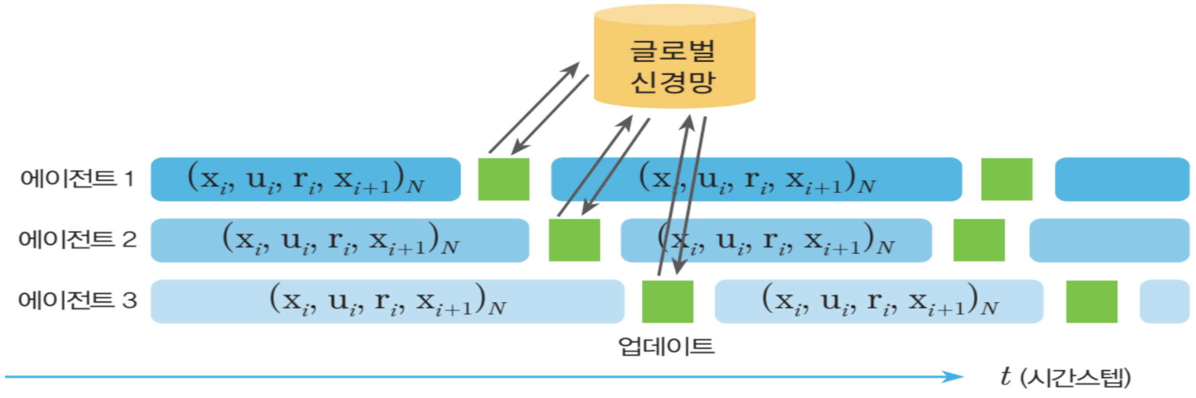


그림 5.6 비동기적 A2C

Credit Assignment Problem

Scalar Global reward

- Allocating which actions each agent has received rewards should be included in the learning process. (Makes confuse to figure out which agent is good for training)
- Allocating reward to each agent could be disturbed by the others

1. Introduction & Background

Notations

$$G = < S, U, P, r, Z, O, n, \gamma >$$

G : Stochastic Game
U : Action
P : State Transition Function
r : Reward Function, $r(s, u): S \times \mathbb{U}$
Z : Observations, agents draw observations $z \in Z$
O : Observation Function, $O(s, a): S \times A \rightarrow Z$
n : Number of Agents
γ : Discount Factor, $\gamma \in [0,1)$
a : Agents, $a \in A \equiv \{1,2, \ldots, n\}$
s : True State of Environment, $s \in S$
u^a : Action Chosen by Agent at Each Time Step, $u^a \in U$
$\mathbb{u}$: Joint Action, $\mathbb{u} \in \mathbb{U} \equiv U^n$
τ^a : Action Observation History of Agent a, $\tau^a \in T$
π : Stochastic Policy, $\pi^a(u^a | \tau^a): T \times U \rightarrow [0,1]$
u^{-a} : agents except a

1. Introduction & Background

Introduction

Reinforcement Learning Problems are naturally modelled as cooperative multi-agent systems.

- Autonomous Vehicles (Cao et al. 2013)
- Network Packet Delivery (Ye, Zhang, and Yang 2015)
- Distributed Logistics (Ying and Dayong 2005)

RL methods designed for single agents typically fare poorly on such tasks (autonomous vehicles …), since the joint action space of the agents grows exponentially with the number of agents

1. Introduction & Background

Centralized training of Decentralized policies

Centralized Training of Decentralized Policies
To cope with such complexity,
It is often necessary to resort to *decentralized policies*,
in which each agent selects its own action only on its local action-observation history.
∴ Agent may make use of RNN (LSTM, GRU, etc…)

Learning can take place in a simulator or a laboratory
in which extra state information is available and agents can communicate freely

Centralized
$\pi^C(\mathbb{u}|s_t) : \mathbb{U} \times S \rightarrow [0,1]$: One centralized policy with given state s_t

Decentralized
$\pi^a(u^a|s_t)$: local policy of agent a
$P(\mathbb{u}|s_t) = \Pi_a \pi^a(u^a|s_t)$: joint-action (product of prob. of each agent with given)

1. Introduction & Background

Limitations

Multi-agent *credit assignment*

- Joint Action generate only global rewards
- Each agent to deduce its own contribution to the team's success
- [Individual Reward Function] not generally available in cooperative settings and often fail

→ ***COunterfactual Multi-Agent*(COMA)** policy gradients & actor-critic

1. Introduction & Background

3 Main Ideas of COMA

1. **COMA uses a centralized critic**
   - **Critic**
     - Centralized
     - Only used during learning
     - Critic conditions on the joint action and all available state information

   - **Actor**
     - Decentralized
     - Needed during execution
     - Policy conditions only on its own action-observation history

1. Introduction & Background

3 Main Ideas of COMA

2. **Counterfactual baseline**
   - Inspired by ***difference rewards***
     - Use ***Aristocrat utility***

   => *Advantage function*

Computes separate baseline for each agent that relies on the centralized critic to reason about counterfactuals in which only that agent's action changes

1. Introduction & Background

3 Main Ideas of COMA

3. **Critic representation**
   - the counterfactual baseline to <u>be computed efficiently.</u>
   - In a single forward pass,
     it computes <u>the Q-values for all the different actions of a given agent</u>,
     conditioned on the actions of <u>all the other agents</u>

1. Introduction & Background

Evaluation

1. Introduction & Background

Evaluation

StarCraft unit micromanagement

- high stochasticity
- large state-action space
- delayed rewards

Previous works have made use of a centralized control policy

COMA

- Massively reduces each agent's field-of-view
- Removes access to macro-actions (combined move and attack)
- Significantly improve performance over other multi-agent actor-critic methods
- Almost SOTA performance of centralized controllers

2. Related Work

Tabular Data

- Busoniu, Babuska, and De Schutter 2008;
- Yang and Gu 2004
- VS ; We used CV data

DQN with independent Q-learning

- Tampuu et al. (2015) : two player pong
- Leibo et al. (2017) : Emergence of collaboration and defection in sequential social dilemmas
- VS ; We used Actor-Critic Method

Emergence of Communication between agents, learned by gradient descent

- centralized traninig (passing gradients between agents during training and sharing parameters)
  - Das et al. (2017)
  - Mordatch and Abbeel (2017)
  - Lazaridou, Peysakhovich, and Baroni (2016)
  - Foerster et al. (2016)
  - Sukhbaatar, Fergus, and others (2016)
- VS ; We used extra state Information during learning and addressed multi-agent credit assignment problem

2. Related Work

Actor-critic methods for decentralized execution with centralized training
- Gupta, Egorov, and Kochenderfer (2017)
- VS; <u>They used hand-crafted local rewards</u>

StarCraft micromanagement
- centralized controller
  - Access to full state, control of all units
  - VS; We used <u>Multi-agent</u>

- Greedy MDP, sequentially choose actions for agents given all previous actions
  - Usunier et al. (2016)

- Actor-critic method that relies on RNNs to exchange information between the agents.
  - Peng et al. (2017)

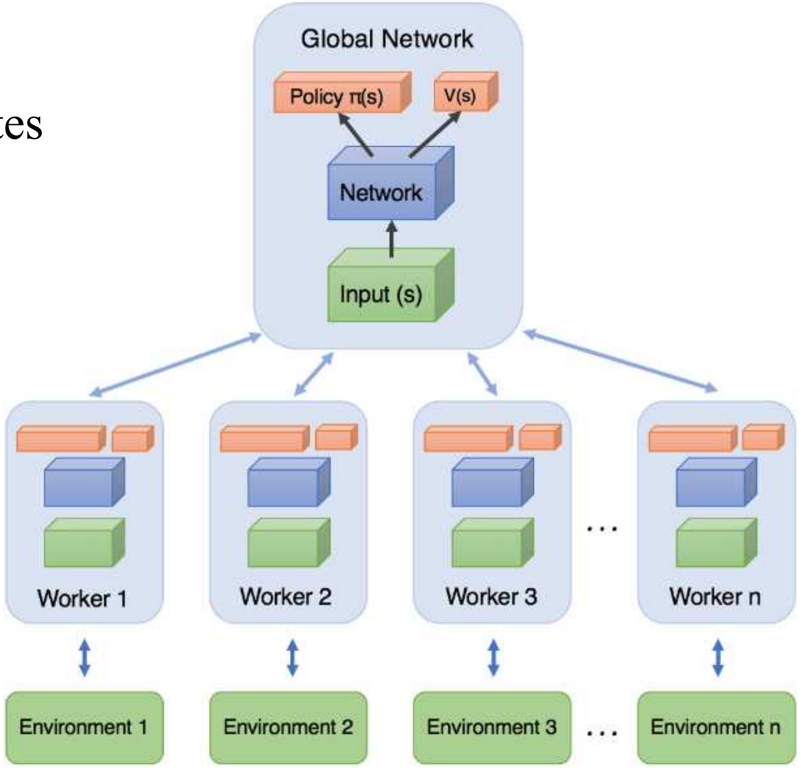
- Multi-Agent representation and decentralized policies with experience replay
  - Foerster et al. (2017)

3. Methods

Independent Actor-Critic (IAC)

Independent Q-Learing
apply policy gradients to multiple agents is learn each agent independently, with its own action-observation history.

Independent Actor-Critic (Baseline)
Actor-critic in place of Q-learning
- Agents can behave differently due to its different observations and hidden states
- Learning remains independent in the sense that each agent
- each actor $\pi(u^a|\tau^a)$ and each critic $Q(\tau^a, u^a)$ or $V(\tau^a)$
  only on the agent's own action-observation history τ^a

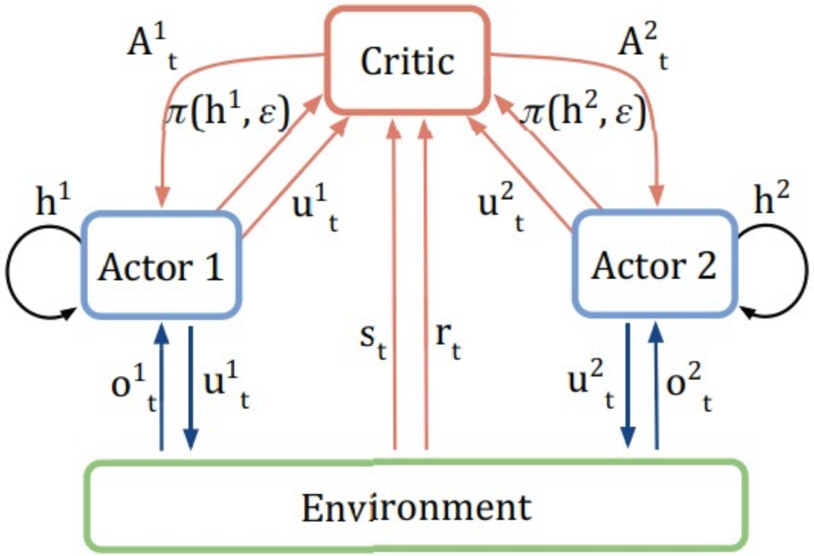


IAC-V
- Critic estimates $V(\tau^a)$, follows a gradient based TD error

IAC-Q
- Critic estimates $Q(\tau^a, u^a)$, follows gradient based advantage
- $\boldsymbol{A(\tau^a, u^a) = Q(\tau^a, u^a) - V(\tau^a), V(\tau^a) = \sum_{u^a} \pi(u^a|\tau^a) Q(\tau^a, u^a)}$

3. Methods

Counterfactual Multi-Agent Policy Gradients



(a)

A centralized critic

COMA

- Critic is used only during learning and the actor is needed during execution
- Critic use true global state s if available, or the joint action-observation histories τ
- Actor use own action-observation histories τ^a
- Centralized critic would be for each actor to follow a gradient based on the TD error

$$g = \nabla_{\theta^\pi} log\pi(u|\tau_t^a)(r + \gamma V(s_{t+1}) - V(s_t))$$

- This TD error considers only global rewards,
  the Gradient computed for each actor does not explicitly reason about how that particular agent's actions contribute to that global reward.

3. Methods

Counterfactual Multi-Agent Policy Gradients

Counterfactual Baseline

Difference rewards

- Shaped reward

$D^a = r(s, \mathbb{u}) - r\big(s, (\mathbb{u}^{-a}, c^a)\big)$

$r(s, \mathbb{u})$: global reward with joint action $\mathbb{u}$

$r\big(s, (\mathbb{u}^{-a}, c^a)\big)$: action of agent a is replaced with a *default action* c^a

∴ true global reward – reward that not depend on agent a's actions

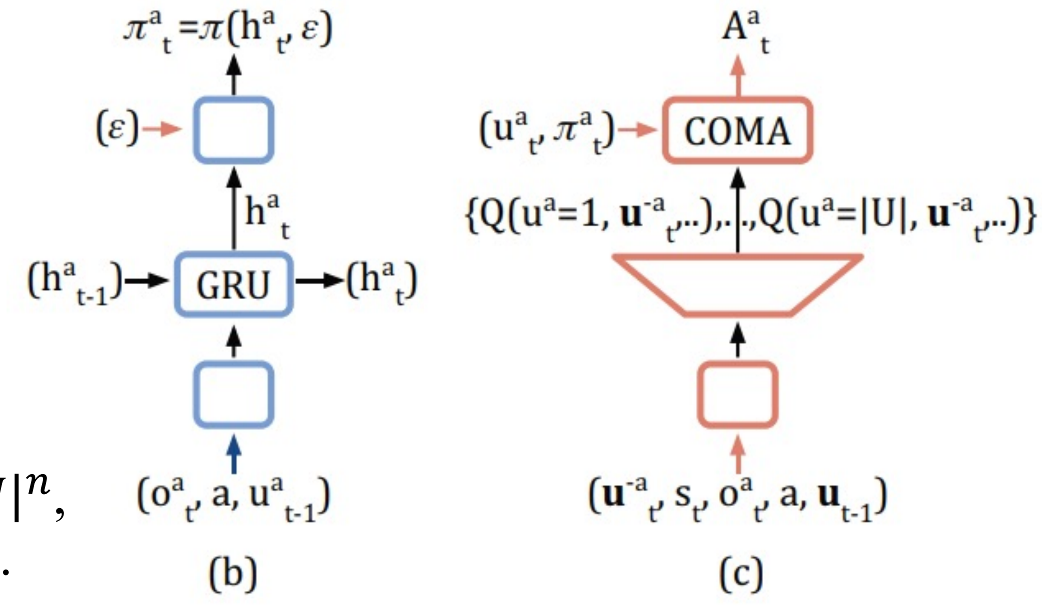
Aristocrat Utility

- avoids the problem of a recursive interdependence between the policy and utility function

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

3. Methods

Counterfactual Multi-Agent Policy Gradients



Single Forward Pass

The number of output nodes of such a network would equal $|U|^n$, the size of the joint action space, making it impractical to train.

COMA uses a critic representation that allows for efficient evaluation of the baseline.

The number of outputs is only $|U|$ instead of ($|U|^n$)
- Large input space that scales linearly in the number of agents and actions

3. Methods

Convergence Proof of COMA to a locally optimal policy

$$g_k = \mathbb{E}_\pi[\textstyle\sum_a \nabla_{\theta_k} \log\pi^a(u^a|\tau^a)A^a(s,u) \quad \text{TD(1)}$$

$$g = \mathbb{E}_\pi[\sum_a \nabla_\theta \log\pi^a(u^a|\tau^a)A^a(s,\mathbb{u})]$$

$$A^a(s,\mathbb{u}) = Q(s,\mathbb{u}) - b(s,\mathbb{u}^{-a})$$

$$\liminf_k ||\nabla J|| = 0 \quad w.p.\ 1.$$

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

$$g_b = -\mathbb{E}_\pi[\sum_a \nabla_{\theta_k} \log\pi^a(u^a|\tau^a)b^a(s,\mathbb{u}^{-a})]$$

3. Methods

Convergence Proof of COMA to a locally optimal policy

$$g_b = -\mathbb{E}_\pi\left[\sum_a \nabla_{\theta_k} \log\pi^a(u^a|\tau^a)b(s,\mathbf{u}^{-a})\right]$$

$$= -\sum_s d^\pi(s)\sum_a\sum_{\mathbf{u}^{-a}}\pi(\mathbf{u}^{-a}|\boldsymbol{\tau}-a)\cdot\sum_{u^a}\pi^a(u^a|\tau^a)\nabla_\theta\log\pi^a(u^a|\tau^a)b(s,\mathbf{u}^{-a})$$

$$= -\sum_s d^\pi(s)\sum_a\sum_{\mathbf{u}^{-a}}\pi(\mathbf{u}^{-a}|\boldsymbol{\tau}-a)\cdot\sum_{u^a}\nabla_\theta\pi^a(u^a|\tau^a)b(s,\mathbf{u}^{-a})$$

$$= -\sum_s d^\pi(s)\sum_a\sum_{\mathbf{u}^{-a}}\pi(\mathbf{u}^{-a}|\boldsymbol{\tau}-a)\cdot b(s,\mathbf{u}^{-a})\nabla_\theta 1$$

$$= 0$$

$d^\pi(s)$: discounted ergodic state distribution

+ Chain Rule

$\nabla_\theta \log\pi(s,a) = \dfrac{\nabla_\theta\pi_\theta(s,a)}{\pi(s,a)}$

$b(s,\mathbf{u}^{-a})$ is not relevant to u^a

$\sum_{u^a}\nabla_\theta\pi^a(u^a|\tau^a) = 1$ by definition

$\nabla_\theta 1 = 0$

∴ the per-agent baseline does not change the expected gradient and not affect the convergence of COMA

3. Methods

Reminder of the expected policy gradient

$$g_k = \mathbb{E}_\pi[\sum_a \nabla_{\theta_k} \log\pi^a(u^a|\tau^a)A^a(s,u)]$$

$$g = \mathbb{E}_\pi[\sum_a \nabla_\theta \log\pi^a(u^a|\tau^a)\underline{A^a(s,\mathbb{u})}]$$

$$A^a(s,\mathbb{u}) = Q(s,\mathbb{u}) - b(s,\mathbb{u}^{-a})$$

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

$$g = \overline{\mathbb{E}_\pi[\nabla_\theta \log\pi(\mathbb{u}|s)Q(s,\mathbb{u})]}$$

3. Methods

Reminder of the expected policy gradient

Single-agent actor-critic policy gradient g

$$g = \mathbb{E}_\pi[\nabla_\theta \log \pi(\mathbb{u}|s)Q(s,\mathbb{u})$$

$$= \mathbb{E}_\pi[\nabla_\theta \log \Pi_a \pi^a(u^a|\tau^a)Q(s,\mathbb{u})$$

$$= \mathbb{E}_\pi[\nabla_\theta \log \pi(\mathbb{u}|s)Q(s,\mathbb{u})$$

$$\pi(\mathbb{u}|s) = \Pi_a \pi^a(u^a\,|\tau^a)$$

Actor-critic following this gradient <u>converges to a local maximum</u> of the expected return J^π, given

1. Policy is differentiable
2. Update timescales for Q and π are sufficiently slow, and that π is updated sufficiently slower than Q
3. Q uses a representation compatible with π

4. Experimental Setup

Decentralized StarCraft Micromanagement

Low-level control of individual units' positioning and attack commands as the fight enemies.

Environment

Symmetric teams formed of

- 3 marines (3m)
- 5 marines (5m)
- 5 wraiths (5w)
- 2 dragoons with 3 zealots (2d_3z)

Enemy team is controlled by the StarCraft AI

4. Experimental Setup

Decentralized StarCraft Micromanagement

Discrete Actions

- Move[direction]
- Attack[enemy_id]
    - Originally, moves into attack range before firing using the game's built-in pathfinding route
    - However, Restricted field of view on the agents, equal to the firing range of ranged units' weapons
- Stop
- Noop
    - Invalid Action choice, such as attack to died enemy

4. Experimental Setup

Figure 2: Starting position with example local field of view for the 2d_3z map.

Decentralized StarCraft Micromanagement

Effect of Restricted field of view on the agents

1. Significant partial observability
2. Units can only attack when they are in range or enemies
3. Agents cannot distinguish between enemies who are dead and who are out of range that can issue invalid attack commands at such enemies, which results in no action being taken.

Increased the average size of the action space, increases the difficulty of both exploration and control.

Before this setting, <u>run forward and attack one enemy instruct achieves 98% win</u>, but 66% in this setting.

4. Experimental Setup

Decentralized StarCraft Micromanagement

Reward Setting

All agents receive the same global reward at each time step,

$$R = \sum(\text{damage on the opponent}) - \frac{1}{2}\sum(\text{damage taken}) + 10 * (\text{enemy killed}) + (\text{remaining health} + 200 \text{ if win})$$

4. Experimental Setup

State Features

Actor Input Features
- Local observations
- Distance
- Relative x
- Relative y
- Unit type
- Shield

	Local Field of View (FoV)							Full FoV, Central Control		
map	heur.	IAC-V	IAC-Q	cnt-V	cnt-QV	COMA mean	COMA best	heur.	DQN	GMEZO
3m	35	47 (3)	56 (6)	83 (3)	83 (5)	**87** (3)	98	74	-	-
5m	66	63 (2)	58 (3)	67 (5)	71 (9)	**81** (5)	95	98	99	100
5w	70	18 (5)	57 (5)	65 (3)	76 (1)	**82** (3)	98	82	70	74[3]
2d_3z	**63**	27 (9)	19 (21)	36 (6)	39 (5)	47 (5)	65	68	61	90

Critic Input Features
- Global state
  - X-y locations relative to the center of the map
  - Health points
  - Cooldown
- Local observations of agents
  - Same but egocentric distances relative to that agent

4. Experimental Setup

Architecture & Training

Actor

- 128-bit gated recurrent units (GRUs)
    - Use fc layers both to process the input and to produce output values
- Action probabilities are produced from the final layer, z
    - Bounded softmax distribution
        - Lower-bounds : $\epsilon/|U|$: $P(u) = (1-\epsilon)softmax(z)_u + \frac{\epsilon}{|U|}$
        - ϵ : linearly from 0.5 to 0.02 across 750 training episodes
    - TD(λ)
        - $\lambda = 0.8$ worked best

Critic

- Factored at the agent level and further exploit internal parameter sharing

4. Experimental Setup

Ablations

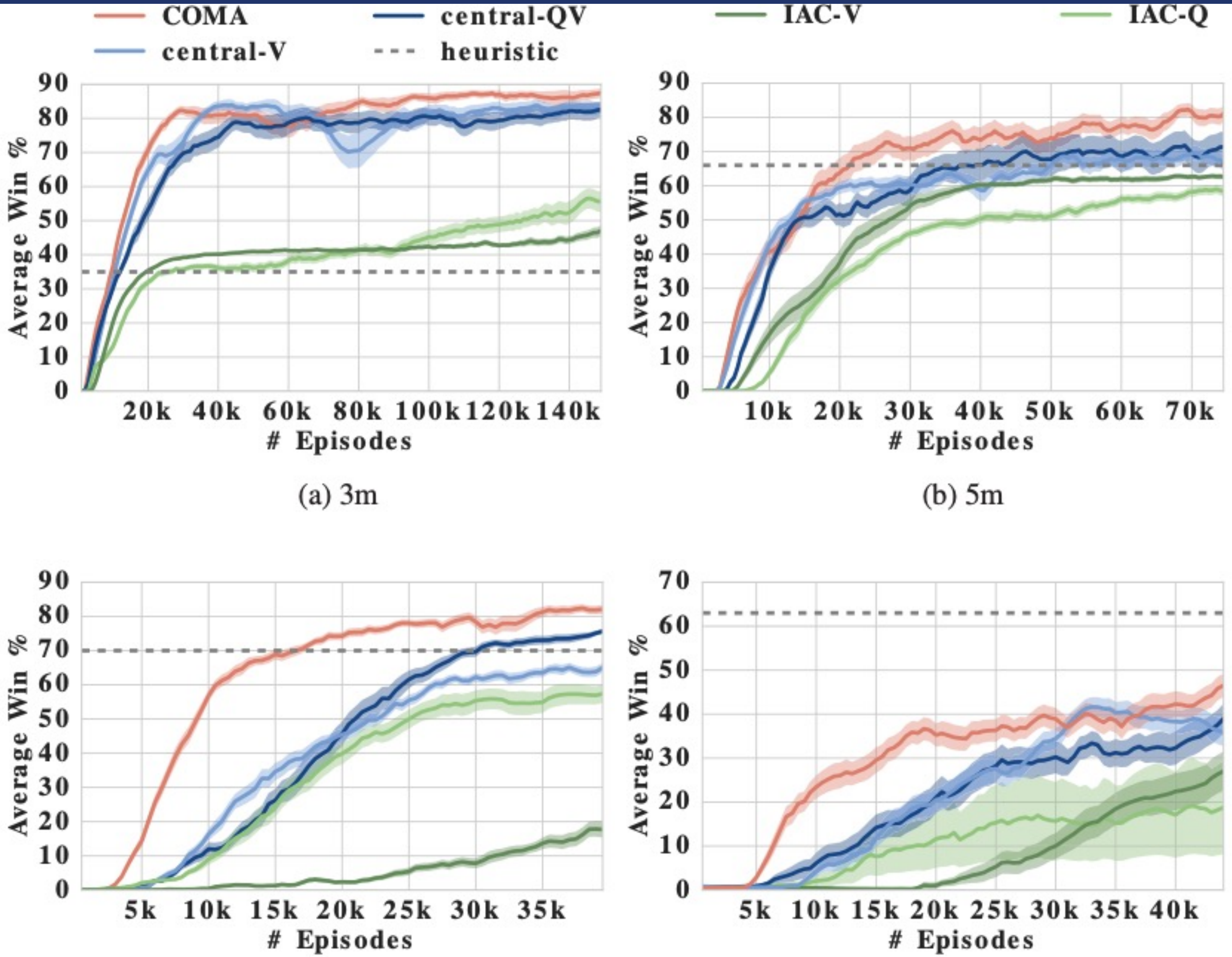
Ablation experiments to validate three key elements of COMA

1. Importance of centralizing the critic by comparing against two IAC variants, IAC-Q, IAC-V
   - IAC-Q : outputs $|U|$ Q-values, one for each action
   - IAC-V : outputs single state-value

2. Significance of learning Q instead of V
   - $central - V$ uses a central state for the critic, but learns $V(s)$
   - Uses the TD Error to estimate the advantage for policy gradient updates

3. Utility of counterfactual baseline
   - $central - QV$ learns both Q and V simultaneously and estimates the advantage as $Q - V$

5. Results



(a) 3m

(b) 5m

(c) 5w

(d) 2d_3z

6. Conclusions & Future Work

Conclusions

centralized critic in order to estimate a counterfactual advantage for decentralized policies in multi-agent RL

Multi-agent credit assignment by using a counterfactual baseline

- Marginalizes out a single agent's action
- Keeping the other agents' actions fixed

improves final performance and training speed

Future Work

Scenarios with large numbers of agents

Centralized critics are more difficult to train

Exploration is harder to coordinate

More sample-efficient variants such as self-driving cars

THANKS FOR LISTENING

Q&A