Actor-Attention-Critic for Multi-Agent Reinforcement Learning

Shariq Iqbal, Fei Sha

ICML(International Conference on Machine Learning) 2019

Outline

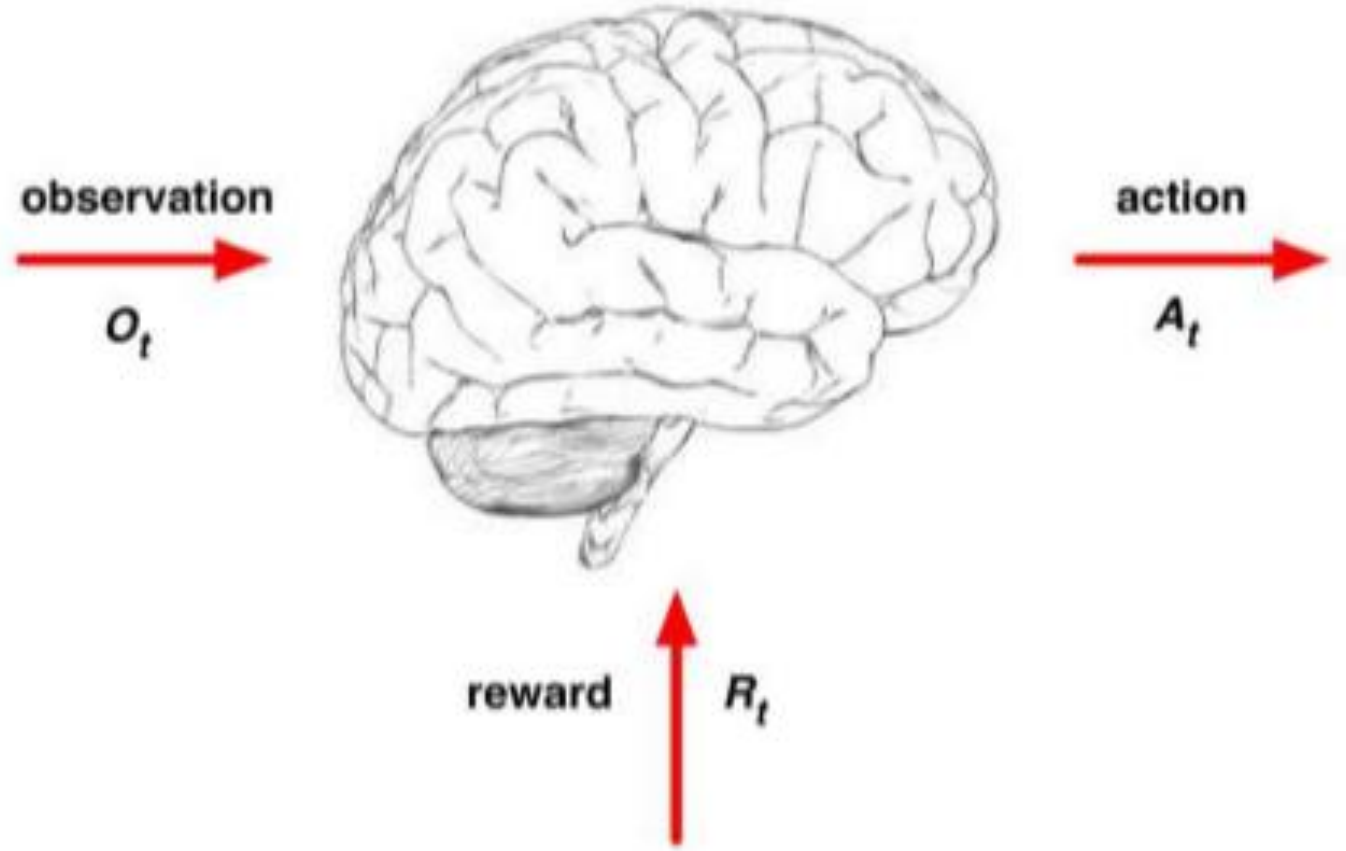
- Background
  - Reinforcement Learning
- Motivation
  - Multi-agent Reinforcement Learning
- Existing Problem
- Our Proposed Approach
- Experiment Result
- Conclusion

Outline

- Background
  - Reinforcement Learning
- Motivation
  - Multi-agent Reinforcement Learning
- Existing Problem
- Our Proposed Approach
- Experiment Result
- Conclusion

Reinforcement Learning

Agent and Environment



Reinforcement Learning

History and State

- The history is the sequence of observations, actions, rewards
- $H_t = O_1, R_1, A_1, \ldots, A_{t-1}, O_t, R_t$
- State is the information used to determine what happens next
- state is a function of the history: $S_t = f(H_t)$

Reinforcement Learning

Markov State

- A state St is Markov if and only if :

$$P[S_{t+1}|S_t] = P[S_{t+1}|S_1, \dots, S_t]$$

- Once the state is known, the history may be thrown away

Important components of an RL agent

- An RL agent may include one or more of these components:
  - Policy: agent's behavior
  - Value function: how good is each state and/or action
  - Model: agent's evaluation of the environment

- An agent aims to learn a policy that maximizes their expected discounted returns.

Outline

- Background
  - Reinforcement Learning
- Motivation
  - Multi-agent Reinforcement Learning
- Existing Problem
- Our Proposed Approach
- Experiment Result
- Conclusion

Motivation

Multi-agent Reinforcement Learning

- Reinforcement learning in multi-agent scenarios is important for real-world applications but presents challenges beyond those seen in single agent settings

- While most reinforcement learning paradigms focus on single agents acting in a static environment (or against themselves in the case of Go), real-world agents often compete or cooperate with other agents in a dynamically shifting environment.

- Any single agent's environment is dynamic and nonstationary due to other agents' changing policies. As such, standard algorithms developed for stationary Markov decision processes fail.

Outline

- Background
  - Reinforcement Learning
- Motivation
  - Multi-agent Reinforcement Learning
- Existing Problem
- Our Proposed Approach
- Experiment Result
- Conclusion

Existing Problem

Two simple methods:

- Training each agent independently to maximize their individual reward, while treating other agents as part of the environment
  - ◆environment is neither stationary nor Markovian
- All agents can be collectively modeled as *a single-agent* whose action space is the joint action space of all agents.
  - ◆ not scalable as the size of action space increases exponentially with respect to the number of agents

- ➢ In order to learn effectively in multi-agent environments, agents must not only learn the dynamics of their environment, but also those of the other learning agents present.
- ➢ Algorithms for multi-agent reinforcement learning are still far from being scalable (to larger numbers of agents) and being generically applicable to environments and tasks that are cooperative (sharing a global reward), competitive, or mixed.

Outline

- Background
  - Reinforcement Learning
- Motivation
  - Multi-agent Reinforcement Learning
- Existing Problem
- Our Proposed Approach
- Experiment Result
- Conclusion

Proposed Approach

Actor-Critic with Attention Mechanism

- The main idea is to learn a centralized critic with an attention mechanism.
  - ✓ The intuition behind our idea comes from the fact that, in many real-world environments, it is beneficial for agents to know what other agents it should pay attention to.

- Our attention critic is able to dynamically select which agents to attend to at each time point during training, improving performance in multi-agent domains with complex interactions.

- It is also applicable to cooperative, competitive, and mixed environments, exceeding the capability of prior work that focuses only on cooperative environments

Proposed Approach

Notation and Background

- We consider the framework of Markov Games (Littman, 1994), which is a multi-agent extension of Markov Decision.
  - S: a set of states
  - $A_1, \ldots, A_N$: action sets for each of N agents
  - $T: S \times \mathrm{A}_1 \times \ldots \times \mathrm{A}_\mathrm{N} \rightarrow \mathrm{P}(\mathrm{S})$
  - $R_i: S \times A_1 \times \cdots \times A_N \rightarrow R$
  - $\pi_i : O_i \rightarrow P(A_i)$
  - $J_i(\pi_i) = E_{a_1 \sim \pi_1, \ldots, a_N \sim \pi_N, s \sim T}[\sum_{t=0}^{\infty} \gamma^t r_{it}(s_t, a_{1t}, \ldots, a_{Nt})]$
  - The agents aim to learn a policy that maximizes their expected discounted returns, where $\gamma \in [0,1]$ is the discount factor that determines how much the policy favors immediate reward over long-term gain.

Proposed Approach

Policy Gradients

Policy Gradients Policy gradient techniques (Sutton et al., 2000; Williams, 1992) aim to estimate the gradient of an agent's expected returns with respect to the parameters of its policy. This gradient estimate takes the following form:

$$\nabla_\theta J(\pi_\theta) = \nabla_\theta \log(\pi_\theta(a_t|s_t)) \sum_{t'=t}^{\infty} \gamma^{t'-t} r_{t'}(s_{t'}, a_{t'}) \quad (1)$$

Proposed Approach

Actor-Critic and Soft Actor-Critic

Actor-Critic and Soft Actor-Critic The term $\sum_{t'=t}^{\infty} \gamma^{t'-t} r_{t'}(s_{t'}, a_{t'})$ in the policy gradient estimator leads to high variance, as these returns can vary drastically between episodes. Actor-critic methods (Konda & Tsitsiklis, 2000) aim to ameliorate this issue by using a function approximation of the expected returns, and replacing the original return term in the policy gradient estimator with this function. One specific instance of actor-critic methods learns a function to estimate expected discounted returns, given a state and action, $Q_\psi(s_t, a_t) = \mathbb{E}[\sum_{t'=t}^{\infty} \gamma^{t'-t} r_{t'}(s_{t'}, a_{t'})]$, learned through off-policy temporal-difference learning by minimizing the regression loss:

$$\begin{aligned} \mathcal{L}_Q(\psi) &= \mathbb{E}_{(s,a,r,s')\sim D}\left[(Q_\psi(s,a) - y)^2\right] \\ \text{where } y &= r(s,a) + \gamma \mathbb{E}_{a'\sim\pi(s')}\left[Q_{\bar{\psi}}(s', a')\right] \end{aligned} \tag{2}$$

where $Q_{\bar{\psi}}$ is the target Q-value function, which is simply an exponential moving average of the past Q-functions and D is a replay buffer that stores past experiences.

Proposed Approach

Actor-Critic and Soft Actor-Critic

To encourage exploration and avoid converging to non-optimal deterministic policies, recent approaches of maximum entropy reinforcement learning learn a soft value function by modifying the policy gradient to incorporate an entropy term (Haarnoja et al., 2018):

$$\nabla_\theta J(\pi_\theta) = \mathbb{E}_{s\sim D, a\sim\pi}[\nabla_\theta \log(\pi_\theta(a|s))(-\alpha \log(\pi_\theta(a|s)) + Q_\psi(s,a) - b(s))] \quad (3)$$

where $b(s)$ is a state-dependent baseline (for the Q-value function). The loss function for temporal-difference learning of the value function is also revised accordingly with a new target:

$$y = r(s,a) + \gamma \mathbb{E}_{a'\sim\pi(s')}[Q_{\bar{\psi}}(s',a') - \alpha \log(\pi_{\bar{\theta}}(a'|s'))] \quad (4)$$

While an estimate of the value function $V_\phi(s)$ can be used a baseline, we provide an alternative that further reduces variance and addresses credit assignment in the multi-agent setting in section 3.2.

Proposed Approach

Multi-Actor-Attention-Critic(MAAC)

- The main idea behind our multi-agent learning approach is to learn the critic for each agent by selectively paying attention to information from other agents.
- This is the same paradigm of training critics centrally (to overcome the challenge of non-stationary non-Markovian environments) and executing learned policies distributedly.

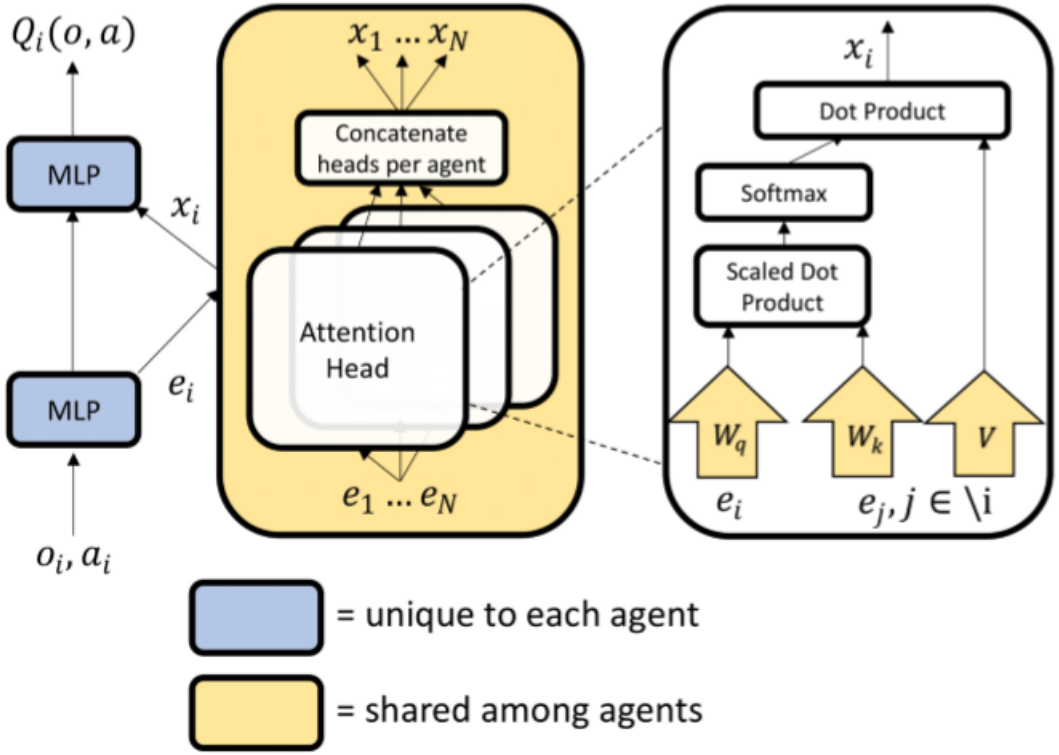


Figure 1. Calculating $Q_i^{\psi}(o,a)$ with attention for agent i. Each agent encodes its observations and actions, sends it to the central attention mechanism, and receives a weighted sum of other agents encodings (each tranformed by the matrix V)

Proposed Approach

Attention

The attention mechanism functions in a manner similar to a differentiable key-value memory model (Graves et al., 2014; Oh et al., 2016). Intuitively, each agent queries the other agents for information about their observations and actions and incorporates that information into the estimate of its value function. This paradigm was chosen, in contrast to other attention-based approaches, as it doesn't make any assumptions about the temporal or spatial locality of the inputs, as opposed to approaches taken in the natural language processing and computer vision fields.

Proposed Approach

Q-value function

$Q_i^{\psi}(o, a)$ is a function of agent i's observation and action, as well as other agents' contributions:

$$Q_i^{\psi}(o, a) = f_i(g_i(o_i, a_i), x_i) \tag{5}$$

where f_i is a two-layer multi-layer perceptron (MLP), while g_i is a one-layer MLP embedding function. The contribution from other agents, x_i, is a weighted sum of each agent's value:

$$x_i = \sum_{j \neq i} \alpha_j v_j = \sum_{j \neq i} \alpha_j h(V g_j(o_j, a_j))$$

where the value, v_j is a function of agent j's embedding, encoded with an embedding function and then linearly transformed by a shared matrix V. h is an element-wise nonlinearity (we have used leaky ReLU).

Proposed Approach

Attention Weight

The attention weight α_j compares the embedding e_j with $e_i = g_i(o_i, a_i)$, using a bilinear mapping (ie, the query-key system) and passes the similarity value between these two embeddings into a softmax

$$\alpha_j \propto \exp(e_j^{\mathrm{T}} W_k^{\mathrm{T}} W_q e_i) \tag{6}$$

where W_q transforms e_i into a "query" and W_k transforms e_j into a "key". The matching is then scaled by the dimensionality of these two matrices to prevent vanishing gradients (Vaswani et al., 2017).

In our experiments, we have used multiple attention heads (Vaswani et al., 2017). In this case, each head, using a separate set of parameters (W_k, W_q, V), gives rise to an aggregated contribution from all other agents to the agent i and we simply concatenate the contributions from all heads as a single vector. Crucially, each head can focus on a different weighted mixture of agents.

Proposed Approach

Learning with Attentive Critics

All critics are updated together to minimize a joint regression loss function, due to the parameter sharing:

$$\mathcal{L}_Q(\psi) = \sum_{i=1}^{N} \mathbb{E}_{(o,a,r,o')\sim D}\left[(Q_i^{\psi}(o,a) - y_i)^2\right], \text{ where}$$

$$y_i = r_i + \gamma \mathbb{E}_{a'\sim\pi_{\bar{\theta}}(o')}[Q_i^{\bar{\psi}}(o',a') - \alpha \log(\pi_{\bar{\theta}_i}(a_i'|o_i'))] \tag{7}$$

where $\bar{\psi}$ and $\bar{\theta}$ are the parameters of the target critics and target policies respectively.

Proposed Approach

Update policy

α is the temperature parameter determining the balance between maximizing entropy and rewards. The individual policies are updated by ascent with the following gradient:

$$\nabla_{\theta_i} J(\pi_\theta) = \mathbb{E}_{o\sim D, a\sim\pi}[\nabla_{\theta_i} \log(\pi_{\theta_i}(a_i|o_i))(-\alpha \log(\pi_{\theta_i}(a_i|o_i)) + Q_i^{\psi}(o,a) - b(o,a_{\setminus i}))] \tag{8}$$

where $b(o, a_{\setminus i})$ is the multi-agent baseline used to calculate the advantage function decribed in the following section. Note that we are sampling all actions, a, from all agents' current policies in order to calculate the gradient estimate for agent i

Proposed Approach

Multi-Agent Advantage Function

Multi-Agent Advantage Function As shown in Foerster et al. (2018), an advantage function using a baseline that only marginalizes out the actions of the given agent from $Q_i^{\psi}(o, a)$, can help solve the multi-agent credit assignment problem. In other words, by comparing the value of a specific action to the value of the average action for the agent, with all other agents fixed, we can learn whether said action will cause an increase in expected return or whether any increase in reward is attributed to the actions of other agents. The form of this advantage function is shown below:

$$\begin{aligned} A_i(o, a) &= Q_i^{\psi}(o, a) - b(o, a_{\setminus i})), \text{ where} \\ b(o, a_{\setminus i})) &= \mathbb{E}_{a_i \sim \pi_i(o_i)} \left[Q_i^{\psi}(o, (a_i, a_{\setminus i})) \right] \end{aligned} \tag{9}$$

Proposed Approach

Multi-Agent Advantage Function

Concretely, in the case of discrete policies, we can calculate our baseline in a single forward pass by outputting the expected return $Q_i(o, (a_i, a_{\setminus i}))$ for every possible action, $a_i \in A_i$, that agent i can take. We can then calculate the expectation exactly:

$$\mathbb{E}_{a_i \sim \pi_i(o_i)} \left[Q_i^{\psi}(o, (a_i, a_{\setminus i})) \right] = \sum_{a_i' \in A_i} \pi(a_i'|o_i) Q_i(o, (a_i', a_{\setminus i})) \tag{10}$$

In order to do so, we must remove a_i from the input of Q_i, and output a value for every action. We add an observation-encoder, $e_i = g_i^o(o_i)$, for each agent, using these encodings in place of the $e_i = g_i(o_i, a_i)$ described above, and modify f_i such that it outputs a value for each possible action, rather than the single input action. In the case of continuous policies, we can either estimate the above expectation by sampling from agent i's policy, or by learning a separate value head that only takes other agents' actions as input.

Outline

- Background
  - Reinforcement Learning
- Motivation
  - Multi-agent Reinforcement Learning
- Existing Problem
- Our Proposed Approach
- Experiment Result
- Conclusion

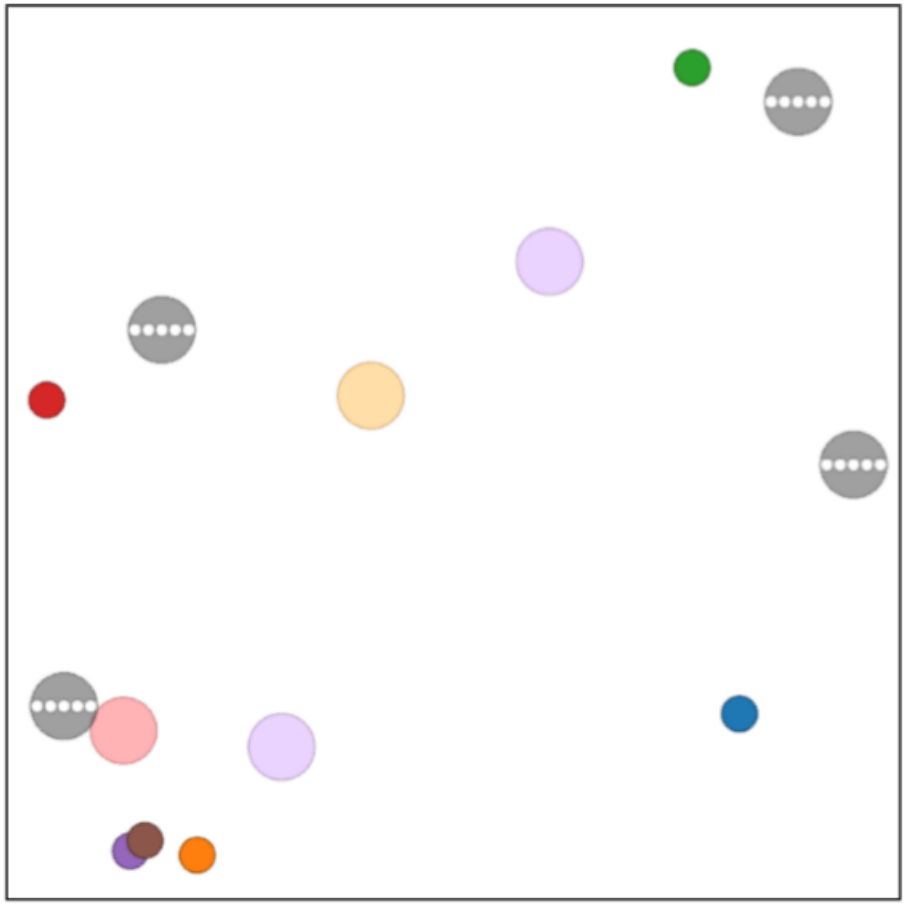
Experiment

Cooperative Treasure Collection

(a) Cooperative Treasure Collection. The small grey agents are "hunters" who collect the colored treasure, and deposit them with the correctly colored large "bank" agents.

Experiment

Rover Tower



(b) Rover-Tower. Each grey "Tower" is paired with a "Rover" and a destination (color of rover corresponds to its destination). Their goal is to communicate with the "Rover" such that it moves toward the destination.

Experiment

Baseline

Table 1. Comparison of various methods for multi-agent RL

	Base Algorithm	How to incorporate other agents	Number of Critics	Multi-task Learning of Critics	Multi-Agent Advantage
MAAC (ours)	SAC‡	Attention	N	✓	✓
MAAC (Uniform) (ours)	SAC	Uniform Atttention	N	✓	✓
COMA*	Actor-Critic (On-Policy)	Global State + Action Concatenation	1		✓
MADDPG†	DDPG**	Observation and Action Concatenation	N		
COMA+SAC	SAC	Global State + Action Concatenation	1		✓
MADDPG+SAC	SAC	Observation and Action Concatenation	N		✓

Heading Explanation *How to incorporate other agents*: method by which the centralized critic(s) incorporates observations and/or actions from other agents (MADDPG: concatenating all information together. COMA: a global state instead of concatenating observations; however, when the global state is not available, all observations must be included.) *Number of Critics*: number of separate networks used for predicting Q_i for all N agents. *Multi-task Learning of Critics*: all agents' estimates of Q_i share information in intermediate layers, benefiting from multi-task learning. *Multi-Agent Advantage*: cf. Sec 3.2 for details.
Citations: *(Foerster et al., 2018), †(Lowe et al., 2017), ‡(Haarnoja et al., 2018), **(Lillicrap et al., 2016)

Experiment

Result

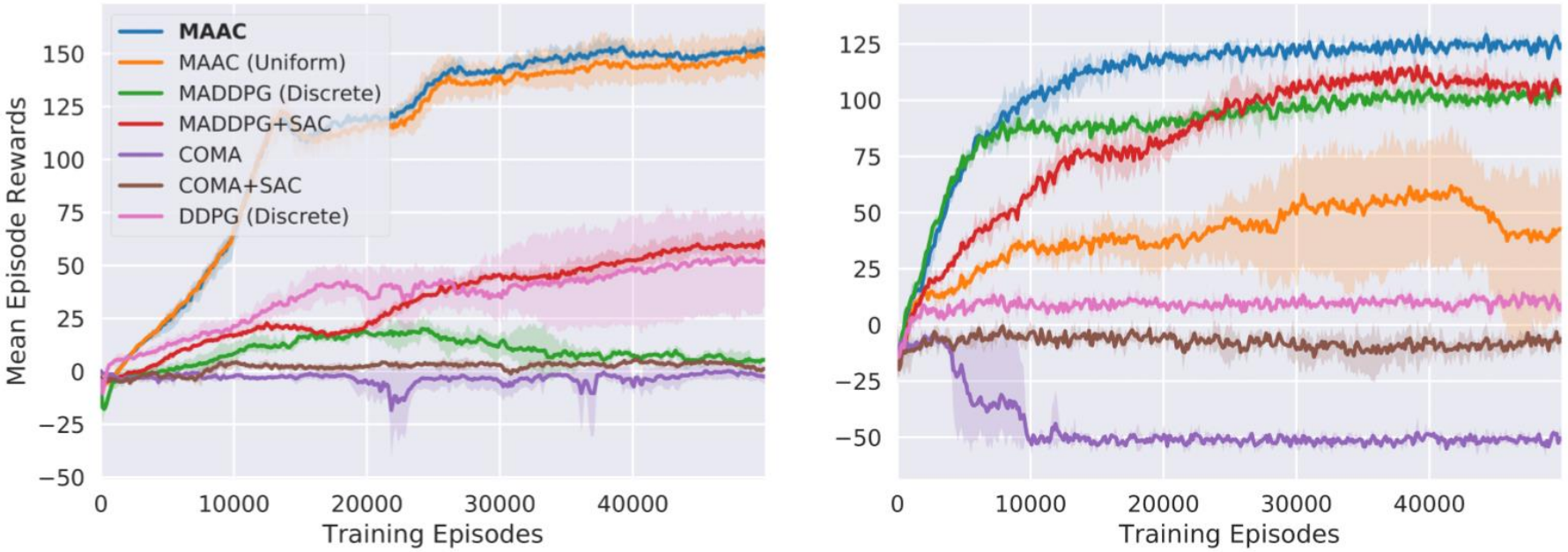


Figure 3. (Left) Average Rewards on Cooperative Treasure Collection. (Right) Average Rewards on Rover-Tower. Our model (MAAC) is competitive in both environments. Error bars are a 95% confidence interval across 6 runs.

Table 2. Average rewards per episode on Cooperative Navigation

MAAC	MAAC (Uniform)	MADDPG+SAC	COMA+SAC
-1.74 $\pm$ 0.05	-1.76 $\pm$ 0.05	-2.09 $\pm$ 0.12	-1.89 $\pm$ 0.07

Experiment

Scalability

Table 3. MAAC improvement over MADDPG+SAC in CTC

# Agents	4	8	12
% Improvement	17	98	208

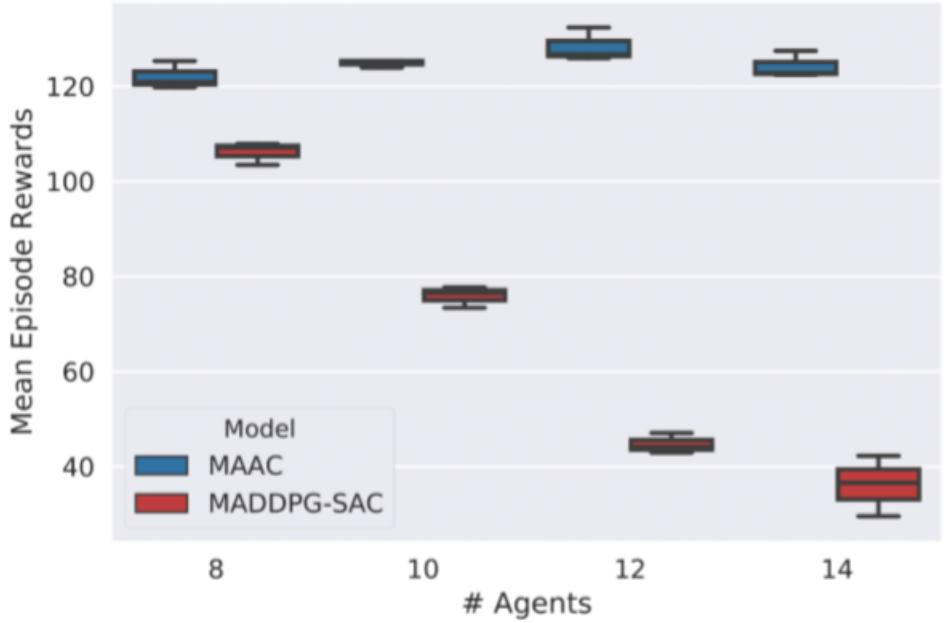


Figure 4. Scalability in the Rover-Tower task. Note that the performance of MAAC does not deteriorate as agents are added.

Experiment

Visualizing Attention

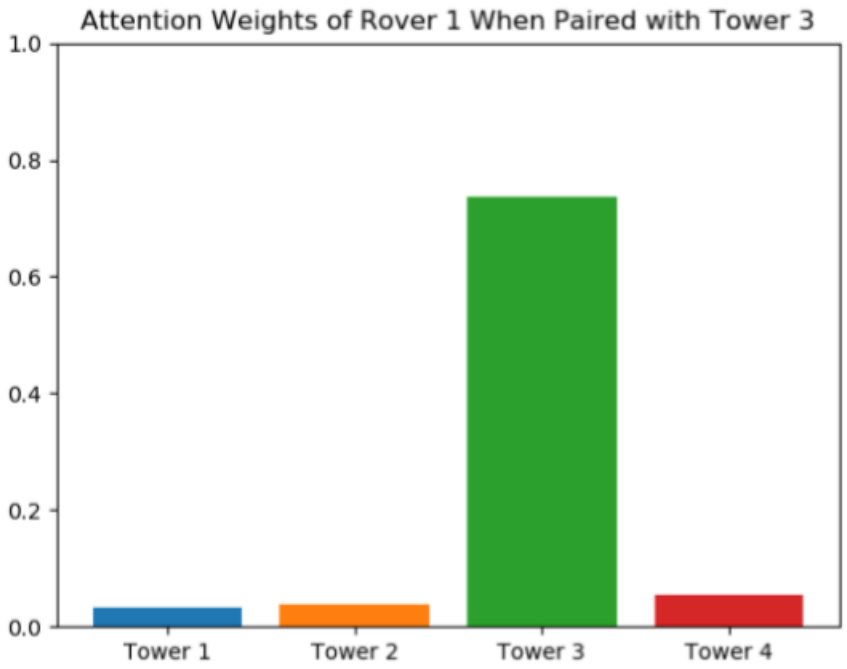


Figure 5. Attention weights over all Towers for a Rover in Rover-Tower task. As expected, the Rover learns to attend to the correct tower, despite receiving no explicit signal to do so.

Outline

- Background
  - Reinforcement Learning
- Motivation
  - Multi-agent Reinforcement Learning
- Existing Problem
- Our Proposed Approach
- Experiment Result
- Conclusion

Conclusion

We propose an algorithm for training decentralized policies in multi-agent settings. The key idea is to utilize attention in order to select relevant information for estimating critics. We analyze the performance of the proposed approach with respect to the number of agents, different configurations of rewards, and the span of relevant observational information. Empirical results are promising and we intend to extend to highly complicated and dynamic environments.

Thank you !