Distributed Critics using Counterfactual Value Decomposition in Multi-Agent Reinforcement Learning

Simon Vanneste
University of Antwerp - imec
IDLab - Faculty of Applied Engineering
Antwerp, Belgium
simon.vanneste@uantwerp.be

Astrid Vanneste
University of Antwerp - imec
IDLab - Faculty of Applied Engineering
Antwerp, Belgium
astrid.vanneste@uantwerp.be

Tom De Schepper
University of Antwerp - imec
IDLab - Department of Computer Science
Antwerp, Belgium
tom.deschepper@uantwerp.be

Siegfried Mercelis
University of Antwerp - imec
IDLab - Faculty of Applied Engineering
Antwerp, Belgium
siegfried.mercelis@uantwerp.be

Peter Hellinckx
University of Antwerp
Faculty of Applied Engineering
Antwerp, Belgium
peter.hellinckx@uantwerp.be

Kevin Mets
University of Antwerp - imec
IDLab - Faculty of Applied Engineering
Antwerp, Belgium
kevin.mets@uantwerp.be

ABSTRACT

In cooperative multi-agent reinforcement learning, the credit assignment limits the ability of the agents to learn a policy. Many state-of-the-art methods use a centralised critic to overcome this credit assignment problem. However, the disadvantage of using a centralised critic is that this limits the scalability of the multi-agent systems following the centralised training and decentralised execution paradigm. The state-of-the-art has attempted to overcome this limitation by using factorisation methods. Unfortunately, these factorisation methods are not usable in every jointly observable environment. This paper presents the Counterfactual Value Decomposition Critics (CVDC) method that follows the decentralised training with free critic communication and a decentralised execution paradigm. The CVDC method uses the insight that any Q-function is decomposable into a set of agent-specific Q-functions. This property is combined with counterfactual reasoning to create a set of decomposed communicating critics, which is usable within every jointly observable environment. The agent-specific critic is then used to train the local policy of an agent without the need for any centralised training structure. We evaluate and compare the CVDC method with other state-of-the-art baselines in a set of environments from the Multi Particle Environments. The results show that our method outperforms the baseline algorithms in training time and obtained return even when parameter sharing is disabled.

KEYWORDS

Value Decomposition, Credit Assignment, Multi-Agent, Reinforcement Learning, Communication Learning

1 INTRODUCTION

Single-agent reinforcement learning (RL) [3, 13, 14, 26, 29, 33] is a popular research domain that has received a lot of attention in recent years. In this research domain, a single agent learns a policy based on an environment-specific reward signal. Multi-agent reinforcement learning (MARL) [1, 2, 5, 12] extends this by training a set of policies in a multi-agent environment.

One of the core challenges of MARL is that the environment appears non-stationary from the perspective of a single agent. This is because, from the viewpoint of a single agent, the other agents are modelled as part of the environment. As a result, any changes in the agents' policies as a direct consequence of exploration or training will cause the environment to appear non-stationary.

This paper focuses on cooperative MARL, in which agents must collaborate to achieve a common goal. The setting where multiple actors work together applies to many real-world applications like industrial robotics or smart grid in energy applications. A shared team reward encourages agents to collaborate towards this common goal by learning their policy based on the shared team reward. However, this causes an additional challenge in which it is difficult for the agent to determine whether its actions contributed to the shared team reward. This challenge is called the credit-assignment problem.

A strategy to tackle the credit-assignment problem is to use the centralised training and decentralised execution (CTDE) paradigm [4, 5, 9, 12] which uses global information during training while still allowing for decentralised policy deployment after training. The CTDE paradigm allows us to train a centralised critic [5, 12] that can be used to train the policies of the various agents using the actor-critic approach [13]. However, the non-stationarity of the environment hinders the training of the centralised critic. A centralised critic is more sensitive to the non-stationarity problem because the centralised critic depends on the actions of every agent.

An alternative to CTDE is the decentralised training and decentralised execution (DTDE) paradigm, where the agents train without any centralised structure or global information. Unfortunately, applications with a large number of agents are also very challenging due to the credit-assignment problem. A hybrid approach between CTDE and DTDE is DTDE with a free critic communication channel where limited communication between the critics is allowed during training. The communication between the critics allows the agents to share observations and actions during training. These local observations of the other agents are necessary to be able to

Proc. of the Adaptive and Learning Agents Workshop (ALA 2023), Cruz, Hayes, Wang, Yates (eds.), May 9-10, 2023, Online, https://ala2023.github.io/. 2023.

train the agent-specific critic in all possible decentralised Markov decision process (Dec-MDP). In a Dec-MDP, the joint observation of the agents defines the state of the environment, which the agent-specific critics need to evaluate the agents' policy.

In this work, we use the DTDE with a free critic communication channel (DTDE-FCC) paradigm. As far as we are aware, this is a novel extension of the DTDE paradigm where critics can send information to each other during training without any cost associated with sending these messages. This paradigm acts as a stepping stone to allow future research to limit the amount of information sent between the critics. It is important to note that with the DTDE-FCC paradigm, it is still possible to train a global Q-function that depends on the environment state and the action of every agent. However, we focus on training a set of decentralised advantage functions instead of learning a global Q-function or global advantage function for every agent. Each agent uses its agent-specific critic to evaluate the behaviour of the policy, reducing the credit assignment problem. The advantage of creating a set of decentralised advantage functions compared to training a centralised variant is that we construct a set of decomposed advantage functions where not every action or observation of the other agents influences the decomposed advantage function. This reduction in influence from the other agents reduces the non-stationarity problem the critics encounter during training. Additionally, the free critic communication channel allows us to send actions and observations to the other agents, which lets us train these critics in any Dec-MDP (as shown in Section 2 and 4). These advantages are demonstrated with the Counterfactual Value Decomposition Critics (CVDC) method. To summarise, the goals of the CVDC methods are shown in the following list.

(1) Train decentralised critics using a free critic communication channel
(2) Train a set of critics within any Dec-MDP
(3) Tackle the credit assignment problem
(4) Reduce the non-stationarity from the point of view of the agent-specific critic

CVDC achieves this by using the novel insight that any Q-function is decomposable into a set of agent-specific Q-functions using the joint observation and the joint action of all the agents. Our theoretical contributions (see Section 5) show that we can use this insight combined with counterfactual reasoning from the COMA method [5] to train a set of decomposed critics. In Section 4, we discuss our CVDC method in more detail.

2 RELATED WORK

The credit-assignment problem [2, 5, 12, 30] is omnipresent within the cooperative multi-agent reinforcement learning domain because a shared team reward is required to learn cooperating policies. A centralised critic under the CTDE paradigm is a popular approach to address the credit-assignment problem. Foerster et al. [5] presented the counterfactual multi-agent (COMA) policy gradients method that trains a centralised critic and uses counterfactual reasoning to extract an agent-specific advantage function. We discuss this method in more detail in Section 3.3. The COMA method is extended to allow for counterfactual communication learning by Vanneste et al. [30] in the multi-agent counterfactual communication (MACC) learning method. These methods use counterfactual actions to create an agent-specific advantage compared to other methods that use counterfactual states or actions to achieve additional objectives (e.g. Counterfactual explanations [18]).

Lowe et al. [12] presents an alternative method to learn a set of policies using a centralised critic in the multi-agent deep deterministic policy gradient (MADDPG) method. MADDPG trains the policies by backpropagating through the critic and actors following the DDPG method [24]. Sheikh and Bölöni [23] explored another approach by extending MADDPG to handle agent-specific and team rewards. This approach combines MADDPG for the team reward and DDPG for the agent-specific reward. However, this method requires an environment that can provide both a team and agent-specific reward. Next, Yang et al. [35] presented the Learning to Incentivize Others (LIO) method in which agents learn an incentive function that is added to the reward function of the other agents to influence them. The incentive function is trained to maximise the extrinsic reward of the agents. However, training a centralised Q-function can be very challenging as the number of agents increases due to the non-stationarity of the changing policy of the agents. Castellini et al. [2] described the difference reward policy gradients (DRPG) method. In this method, the centralised critic learns the global reward function (which remains stationary during training) instead of learning a global Q-function.

The centralised critic approaches [2, 5, 12, 30] are challenging to train and very sensitive to the used hyperparameters when the number of agents increases. Additionally, by creating a global Q-function that depends on the joint action of all the agents, the non-stationarity of Q-function (caused by the learning agents) can make the training of the centralised critic unstable. Decomposition methods try to tackle these problems by splitting the global Q-function into a set of agent-specific Q-functions. Sunehag et al. [25] presented value decomposition networks (VDN) which assume that the global Q-function can be composed of the sum of agent-specific Q-functions $Q(o, u) = \sum_a Q^a(o^a, u^a)$. This assumption allows this method to train the agent-specific Q-functions by backpropagating gradients through the global Q-function, constructed as the sum of the agent-specific Q-functions. These agent-specific Q-functions are directly usable as a policy after the training process. The multi-agent decomposed policy gradient method (DOP) [32] is an actor-critic variant of VDN in which the decomposed critic is used to train the policy of an individual agent.

Rashid et al. [20] extend this method by using a mixing network instead of summing the agent-specific Q-values within the QMIX method. This mixing network will mix the agent-specific Q-values monotonically based on the complete environment state. In later work, QMIX is extended in the Weighted QMIX method [19] in which suboptimal actions are down-weighted during the training of the global Q-function. Additionally, the Attentive-Imaginative QMIX method [7] extends QMIX by using an attention-based mixing network. An actor-critic variant of QMIX is the LICA method [36], in which a set of policies are directly decomposed using a mixing network. Instead of mixing the agent-specific Q-values, LICA use the output distribution of the actors in combination with the mixing network to create a Q-value estimate.

Table 1: An overview of which methods can learn an advantage or Q-function for different MDP classes. When a method cannot learn an advantage or Q-function for every MDP within a specific class, the number of plus signs denotes the MDP set size where the technique can learn a Q-function.

	MMDP	Dec-MDP	Dec-POMDP
Centralized Critic FC [5]	✓	✓	✗ (++)
VDN [25], DOP [32]	✓	✗	✗
QMIX [20], LICA [36]	✓	✗ (+)	✗ (+)
CVDC (this work)	✓	✓	✗ (++)

Our work differs from other state-of-the-art methods by creating a set of decentralised critics to train agent-specific policies. We achieve this by training the critics using a counterfactual decomposition method. The CVDC method can tolerate the non-stationarity problem because the agent-specific critics can learn how much actions from the other agents need to be taken into account. This method reduces the effect of the non-stationarity as not every changing policy impacts every critic. Additionally, the state-of-the-art decomposition methods limit the information available to the agent-specific Q-function to decompose the global Q-function to a subset of Dec-MDPs.

Table 1 compares CVDC and a centralised critic with a fully connected (FC) neural network architecture for different decomposition methods and different MDPs combinations. This table shows that the VDN and QMIX methods are not able to learn within every Dec-MDP. This limitation is caused by the agent-specific Q-functions not having access to the global state but only to the local observations, which is not always sufficient in a jointly observable MDP. The centralised FC critic or CVDC have access to the joint observation and joint action, which allows them to learn an advantage or Q-function within any Dec-MDP.

3 BACKGROUND

3.1 Cooperative Multi-Agent Systems

A cooperative multi-agent system can be described as a decentralised POMDP (Dec-POMDP)[17] which is defined by the tuple $\mathbb{M} = \langle \mathbb{D}, \mathbb{S}, \mathbb{U}, P, \mathbb{O}, O, R, h, P_0 \rangle$. In this tuple, $\mathbb{D}$ is the set of n agents in the Dec-POMDP in which the current agent is denoted by the superscript a and the other agents are denoted by the superscript $-a$. Additionally, the subscript t is used to indicate a certain time step. The global state $s_t \in \mathbb{S}$ of the environment is sampled from the transition function $P(s_{t+1}|s_t, u_t) : \mathbb{S} \times \mathbb{U} \times \mathbb{S} \rightarrow [0, 1]$. The state at time step $t = 0$ is defined by the initial state distribution P_0. The joint action $u_t \in \mathbb{U}$ is a tuple of the individual actions $u_t = \langle u_t^0, u_t^1, ..., u_t^n \rangle$ from the n different agents. Every individual agent a can perform an action $u_t^a \in \mathbb{U}^a$ in which $\mathbb{U}^a$ is the set of actions that are available to agent a. These individual actions are selected by the policy of the agent $\pi^a(u_t^a|o_t^a)$ based on the local observation $o_t^a \in \mathbb{O}^a$. The set of observations $\mathbb{O}^a$ is the set of all possible observations for agent a. The tuple of joint observations for all agents o_t is defined by $o_t \in \mathbb{O}$ which is drawn from the observation function $O(s_t, u_t) : \mathbb{S} \times \mathbb{U} \rightarrow \mathbb{O}$. The horizon h is

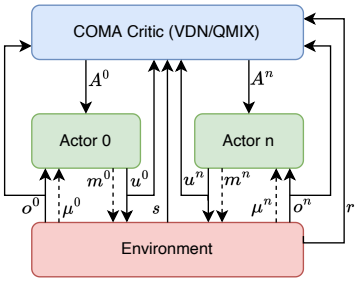


Figure 1: The COMA or MACC method using a centralised critic. The optional messages that are used in the MACC method are shown with a dotted line.

the number of time steps the agents are interacting with the environment. The policy π^a of agent a at time step t is trained with the team reward signal r_t which is defined by the reward function $R(s_t, u_t) : \mathbb{S} \times \mathbb{U} \rightarrow \mathbb{R}$. The agents adapt their policy to maximise the discounted expected reward $G(s_t, u_t) = \sum_{k=t}^{h} \gamma^{k-t} R(s_k, u_k)$ with a discount factor $\gamma \in [0, 1]$. A jointly observable Dec-POMDP is a decentralised Markov decision process (Dec-MDP) [17] in which the joint observation of the agents defines the state of the environment. However, an individual agent still only has a partial view of the state. The Dec-POMDP in which the individual agents observe the global state is called a multiagent Markov decision process (MMDP) [17].

3.2 Policy Gradient

In this section, we discuss the single-agent policy gradient which is extended to a multi-agent variant in the following section. Policy gradient methods directly optimise the parameters of the agents' policy using gradient ascent in which the gradient approximation is denoted by g (see Equation 1).

$$g = \mathbb{E}\Big[\sum_{t=0}^{\infty} \Psi_t \nabla_\theta \ln \pi_\theta(u_t|s_t) \Big] \tag{1}$$

In this paper, we use the actor-critic approach [10, 22, 27] where Ψ_t is defined as $Q(s_t, u_t)$ or $A(s_t, u_t) = Q(s_t, u_t) - b(s_t)$ in order to learn the policy of the actor. In many cases the baseline $b(s_t)$ is defined as the utility of a state $V(s_t)$. The value function, that is used to train the actor, is learned by the critic.

3.3 Counterfactual Training

The counterfactual multi-agent (COMA) policy gradients method [5] uses the actor-critic architecture in which the centralised critic learns a global Q-function $Q(s_t, u_t)$ using the global state s_t of the environment and the joint actions u_t of the different agents. The value for Ψ_t^a, to train the policy for agent a using policy gradient (see Section 3.2), is defined as the agent-specific advantage function $A^a(s_t, u_t)$. This agent-specific advantage function is created using the global Q-function as shown in Equation 2 by iterating over all

possible counterfactual actions $u_t'^a$.

$$A^a(s_t, u_t) = Q(s_t, u_t) - \sum_{u_t'^a} \pi^a(u_t'^a | o_t^a) Q(s_t, \langle u_t'^a, u_t^{-a} \rangle)$$
$$= Q(s_t, u_t) - \mathbb{E}_{\pi^a}[Q(s_t, u_t)] \tag{2}$$

An extension to COMA is the Multi-Agent Counterfactual Communication (MACC) learning method [30] which learns an additional communication policy to send discrete messages m to other agents. Using these received messages μ, the actors of the other agents will generate a new action and message. MACC uses counterfactual reasoning about alternative messages and their impact on the policy distribution of the other agents to train their communication policy. The global COMA/MACC architecture is shown Figure 1.

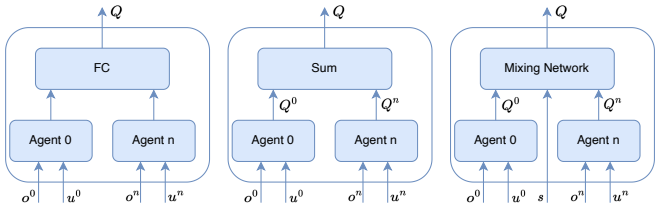
3.4 Centralised Critic Architectures

The COMA/MACC method uses a global Q-function to create an agent-specific advantage. However, when the number of agents increases, learning the global Q-function becomes infeasible without a specialised neural network architecture. The first architecture uses a fully connected (FC) neural network (see Figure 2a) in which the preprocessing sections of the neural network are agent specific. The advantage of this architecture is that the network parameters between identical agents are shared within the global Q-function. We define identical agents as agents that share neural network parameters following the CTDE paradigm. The subsequent architecture is based on the Value Decomposition Networks (VDN) method [25] (see Figure 2b). The VDN method uses the sum of agent-specific Q-functions to create the global Q-function. This idea within a centralised critic allows us to share the neural network parameters between identical agents. However, the agent-specific Q-functions only use the local observation and action of the agent to predict the Q-function. This property limits the ability of VDN to learn the global Q-function within certain MDPs (as discussed in Section 2 and 4). The final architecture uses a set of agent-specific Q-functions mixed monotonically based on the global state in a mixing network, as described in the QMIX method [20]. Here, we can again use parameter sharing between the different agent-specific Q-function between identical agents. This QMIX critic neural network architecture is shown in Figure 2c. In the experiments of this work, the COMA/MACC method, with these different architectures, is used as a set of baselines.

4 METHODS

This section introduces our novel Counterfactual Value Decomposition Critics (CVDC) method. The cornerstone of this method is the insight that any global Q-function $Q^{tot}(o_t, u_t)$ in a Dec-MDP can be defined as the sum of the agent-specific Q-functions $Q^a(o_t, u_t)$ for agent a, as shown in Equation 3.

$$Q(o_t, u_t) \triangleq \sum_a Q^a(o_t, u_t) \tag{3}$$



(a) FC architecture **(b) VDN architecture** **(c) QMIX architecture**

Figure 2: The baseline neural network architectures for the centralised critic.

This property may seem trivial at first glance, although it is only valid when the agent-specific Q-functions take the joint observation and joint action as input. This requirement allows the agent-specific Q-function to learn a scaled version of the global Q-function $Q^a(o_t, u_t) = \frac{1}{n} Q(o_t, u_t)$, which means that any global Q-function can be composed of the summed agent-specific Q-functions. However, many Q-functions can be decomposed much more efficiently (e.g. when not every agent influences every other agent-specific Q-function), which we further discuss in Section 4.1. Alternatively, when the agent-specific Q-functions do not use the joint observation of the other agents, the agent-specific Q-function model needs to learn an expectation of the joint observation of the other agents as shown in Equation 4. This approach works for any MMDP but not for every Dec-MDP. In a Dec-MDP, the full state is no longer uniquely defined based on the agents' observation which prevents the convergence of the agent-specific Q-function model.

$$Q^a(o_t^a, u_t) = \mathbb{E}_{o_t^{-a}}\left[Q^a(o_t, u_t) \middle| P(o_t^{-a} | o_t^a) \right] \tag{4}$$

Similarly, when the joint action of the other agents is not used as an input for the agent-specific Q-function, the model needs to learn an expectation of the actions of the other agents (see Equation 5). However, in a multi-agent reinforcement learning system, the probability of the actions of the other agents will change due to their changing policy during training. This changing expectation will make the environment appear non-stationary causing learning instabilities and preventing the agent-specific Q-function models from converging. These properties of the joint action of the other agents are independent of the type of MDP.

$$Q^a(o_t, u_t^a) = \mathbb{E}_{u_t^{-a}}\left[Q^a(o_t, u_t) \middle| P(u_t^{-a} | o_t^a) \right] \tag{5}$$

So to learn an agent-specific Q-function in a Dec-MDP and limit the non-stationarity of the environment, the agent-specific Q-functions require the joint action and observation of the agents.

4.1 Counterfactual Value Decomposition Critics

In this section, we will use the insight from Equation 3 to create our Counterfactual Value Decomposition Critics (CVDC) method. This method aims to learn a policy within any Dec-MDP, tackle the credit assignment problem, reduce the non-stationarity (from the viewpoint of the critics) and train the critics decentralised using a free critic communication channel.

The CVDC architecture (see Figure 3) adapts the COMA [5] or MACC [30] architecture (see Figure 1) by replacing the centralised

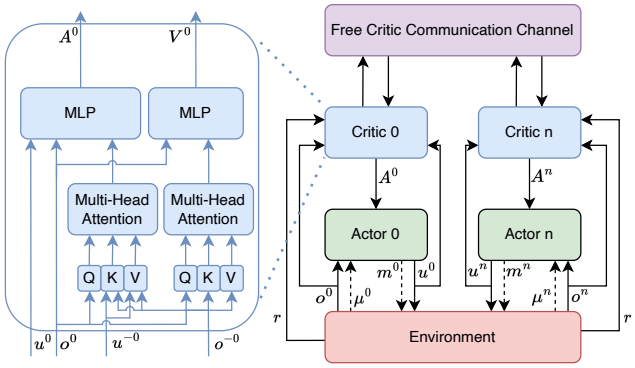


Figure 3: The CVDC method with decentralised critics. The optional messages that are used in the MACC method are shown with a dotted line.

critic with a set of decentralised critics. Every critic learns a personalised advantage and utility function (see Section 4.1.1 and 4.1.2) using a free communication channel in which observations, actions, advantages and utilities are shared between the critics. The agent trains its policy (see Section 4.2) by using the advantage function of the agents' critic.

4.1.1 Advantage Decomposition. The decomposed agent-specific advantage function $A^a(s_t, u_t)$, used by CVDC, adapts the COMA [5] definition of the agent-specific advantage function without the need for a global Q-function. We achieve this adaptation by combining it with the property that the global advantage is the sum of the agent-specific advantage functions $A(o_t, u_t) \triangleq \sum_a A^a(o_t, u_t)$ which can be derived from Equation 3. In Equation 6, we show the resulting agent-specific advantage function where the advantage of the other agents is defined as $A^{-a}(s_t, u_t) = \sum_{a' \neq a} A^{a'}(s_t, u_t)$. It is important to note that the agent-specific advantage function uses the joint observation and actions to reduce the non-stationarity problem and to allow us to train on every Dec-MDP. The theoretical contribution used in creating this equation is discussed in Section 5.

$$A^a(s_t, u_t) = A(s_t, u_t) - \mathbb{E}_{\pi^a}[A^{-a}(s_t, u_t)] \tag{6}$$

The intuition behind this equation is that the critics learn a customised agent-specific advantage function to train the agents' policy. This effect of using the expected advantage of the other agents $\mathbb{E}_{\pi^a}[A^{-a}(s_t, u_t)]$ is shown in Equation 7.

$$\begin{aligned} A^a(s_t, u_t) &= A(s_t, u_t) - \mathbb{E}_{\pi^a}[A^{-a}(s_t, u_t)] \\ A^a(s_t, u_t) &= A^a(s_t, u_t) + A^{-a}(s_t, u_t) - \mathbb{E}_{\pi^a}[A^{-a}(s_t, u_t)] \\ 0 &= A^{-a}(s_t, u_t) - \mathbb{E}_{\pi^a}[A^{-a}(s_t, u_t)] \\ A^{-a}(s_t, u_t) &= \mathbb{E}_{\pi^a}[A^{-a}(s_t, u_t)] \end{aligned} \tag{7}$$

Here we see that Equation 6 is only correct when the expected advantage of the other agents is equal to the advantage of the other agents. This equality is only valid when the advantage of the other agents is independent of the policy of the current agent. So when learning the agent-specific advantage functions, the optimisers attempt to make the agent-specific advantage functions as independent as possible from the current actions and policies of the other agents. This novel property allows us to decompose the global advantage function into a set of agent-specific advantage functions that can be used directly to train the agents' policy.

However, we cannot directly use the definition of the agent-specific advantage function as stated in Equation 6 because $A(s_t, u_t)$ is used in the function definition, which causes a circular reference. Nevertheless, we can create a target advantage approximation $\hat{A}^a(s_t, u_t)$ using a combination of the reward, advantages of the other agents and the utility as shown in Equation 8.

$$\begin{aligned} \hat{A}^a(s_t, u_t) &= A_{\theta^-}(s_t, u_t) - \mathbb{E}_{\pi^a}[A^{-a}_{\theta^-}(s_t, u_t)] \\ &= Q_{\theta^-}(s_t, u_t) - V_{\theta^-}(s_t) - \mathbb{E}_{\pi^a}[A^{-a}_{\theta^-}(s_t, u_t)] \\ &= r_t + \gamma Q_{\theta^-}(s_{t+1}, u_{t+1}) - V_{\theta^-}(s_t) - \mathbb{E}_{\pi^a}[A^{-a}_{\theta^-}(s_t, u_t)] \\ &= r_t + \gamma(A_{\theta^-}(s_{t+1}, u_{t+1}) + V_{\theta^-}(s_{t+1})) \\ &\quad - V_{\theta^-}(s_t) - \mathbb{E}_{\pi^a}[A^{-a}_{\theta^-}(s_t, u_t)] \end{aligned} \tag{8}$$

We use target networks [15] (noted as θ^- in Equation 8) to calculate the target advantage value to improve the training stability. Finally, this definition of the agent-specific advantage function is used to create an advantage loss function, as shown in Equation 9.

$$\mathcal{L}(\theta^a_A) = \left(A^a_\theta(s_t, u_t) - \hat{A}^a(s_t, u_t)\right)^2 \tag{9}$$

4.1.2 Utility Decomposition. The agent-specific advantage function requires us to learn a global utility function, as shown in Equation 8. However, the goal of our approach is to create a set of decentralised critics which do not require us to learn any global function. So to create the global utility from a set of agent-specific utility functions, we use the property that the global utility can be defined as the sum of agent-specific utility functions $V(s_t, u_t) = \sum_a V^a(s_t, u_t)$. This property is then used to create the target approximation of the agent-specific utility function $\hat{V}^a(s_t)$, as shown in Equation 10. The utility of the other agents is defined as the sum of the utility of the other agents $V^{-a}(s_t, u_t) = \sum_{a' \neq a} V^{a'}(s_t, u_t)$, which is similar to the notation used in the advantage definition.

$$\begin{aligned} \hat{V}^a(s_t) &= V_{\theta^-}(s_t) - V^{-a}_{\theta^-}(s_t) \\ &= (r_t + \gamma V_{\theta^-}(s_{t+1})) - V^{-a}_{\theta^-}(s_t) \end{aligned} \tag{10}$$

In order to stabilise the learning, we used target networks (noted as θ^-) of the global utility of the next state and the utility of the other agents. This definition of the utility approximation can then be used in the agent-specific utility loss function, as shown in Equation 11.

$$\mathcal{L}(\theta^a_V) = \left(V^a_\theta(s_t) - \hat{V}^a(s_t)\right)^2 \tag{11}$$

4.1.3 Model Architecture. The agent-specific advantage and utility functions for our CVDC method are represented by a neural network (see Figure 3) which both uses the multi-head attention [31] architecture for the observation and actions of the other agents. The keys and values for the multi-head attention network are created based on the observations o^{-a} and actions u^{-a} of every agent apart from the current agent by a fully connected network. The query for the multi-head attention network is directly created from the observation o^a of the current agent to let the multi-head attention network filter the observations and actions of the other agents based on this observation. The output of the multi-head attention is

concatenated with the observation o^a and action u^a of the current agent in a fully connected network. The attention part of our neural network architecture allows the agent to handle a larger number of agents or a variable number of agents and represent this in a fixed representation size. This kind of neural network architecture is similar to the work of Iqbal and Sha [8], where a multi-head attention mechanism is used to encode the actions and observations of the other agents.

4.2 Actors

In this section, we describe the actors' training and exploration. The learned agent-specific advantage function $A^a_\theta(s_t, u_t)$ is a learned approximation of the true agent-specific advantage $A^a(s_t, u_t)$. In order to improve stability, we define Ψ^a_t as defined in Equation 12 by subtracting a baseline [34] from the advantage.

$$\Psi^a_t = A^a_\theta(s_t, u_t) - \mathbb{E}_{\pi^a}[A^a_\theta(s_t, u_t)] \tag{12}$$

The resulting Ψ^a_t is used to train the policy of the agent using the policy gradient method as defined in Section 3.2. In the baseline actors, we use the COMA definition of Ψ^a_t (see Section 3.3).

In the experiments for CVDC and the baselines, we learn with a large number of agents, which requires a considerable amount of exploration to find a set of suitable policies. To make this exploration explicit, we use the off-policy actor-critic [3] learning rule as it allows for a different behaviour policy $\pi^a_b(u^a|o^a)$ from the agent policy $\pi^a(u^a|o^a)$ (see Equation 13).

$$g = \mathbb{E}\Big[\Psi_t \nabla_{\theta^a} \frac{\pi^a(u^a|o^a)}{\pi^a_b(u^a|o^a)} \ln \pi^a(u^a|o^a)\Big] \tag{13}$$

The behaviour policy explicitly enforces exploration by balancing between following the agent policy and uniformly exploring the different actions by a decaying weight ϵ from 1 to 0 (see Equation 14).

$$\pi^a_b(u^a|o^a) = \epsilon \frac{1}{|U|} + (1-\epsilon)\pi^a(u^a|o^a) \tag{14}$$

This exploration strategy is based on the work of Liu et al. [11], where off-policy methods can use a separate exploration policy from the target policy. However, we use a uniform distribution exploration policy instead of training an exploration policy. We do this off-policy actor-critic learning and exploration for both CVDC and the baseline methods to allow us to compare the performance of the methods critic with identical actors. The advantage of making the exploration explicit is that the different methods achieve better results while requiring less hyperparameter tuning because the epsilon-decay parameters are more intuitive.

In conclusion, the agents in our methodology learn an agent-specific advantage function $A^a(s_t, u_t)$, agent-specific utility function $V^a(s_t)$ and an agent-specific policy $\pi^a(u^a|o^a)$ by sharing observations, actions, advantages and values between the different agents. The advantage and utility functions of the different agents interact with each other to learn a valid policy for every agent in the multi-agent system. It is important to note that this method does not rely on any centralised structure to learn these policies but a set of agent-specific critics to reduce the non-stationarity problem, tackle the credit assignment problem and is able to train within every Dec-MDP.

5 THEORETICAL CONTRIBUTIONS

In this section, our theoretical contributions are described. These contributions allow us to create a novel agent-specific advantage definition. The first step is to show that the expected value of the agent-specific advantage function is zero (see Lemma 5.1).

Lemma 5.1. $\mathbb{E}_{\pi^a}\left[A^a(s_t, u_t)\right] = 0$

Proof.

$$\begin{aligned}
A^a(s_t, u_t) &= Q(s_t, u_t) - \mathbb{E}_{\pi^a}[Q(s_t, u_t)] \\
\mathbb{E}_{\pi^a}[A^a(s_t, u_t)] &= \mathbb{E}_{\pi^a}[Q(s_t, u_t) - \mathbb{E}_{\pi^a}[Q(s_t, u_t)]] \\
&= \mathbb{E}_{\pi^a}[Q(s_t, u_t)] - \mathbb{E}_{\pi^a}[\mathbb{E}_{\pi^a}[Q(s_t, u_t)]] \\
&= \mathbb{E}_{\pi^a}[Q(s_t, u_t)] - \mathbb{E}_{\pi^a}[Q(s_t, u_t)] \\
&= 0
\end{aligned} \tag{15}$$

□

The next step is to rework the original COMA [5] definition of the agent-specific advantage function. We do this by combining the COMA definition with the property that the global advantage is constructed from a set of agent-specific advantages to create the novel agent-specific advantage definition.

Theorem 5.2. *The agent-specific advantage function $A^a(s_t, u_t)$ is equal to the difference between the global advantage and the expected advantage of the other agents over the policy of the current agent.*

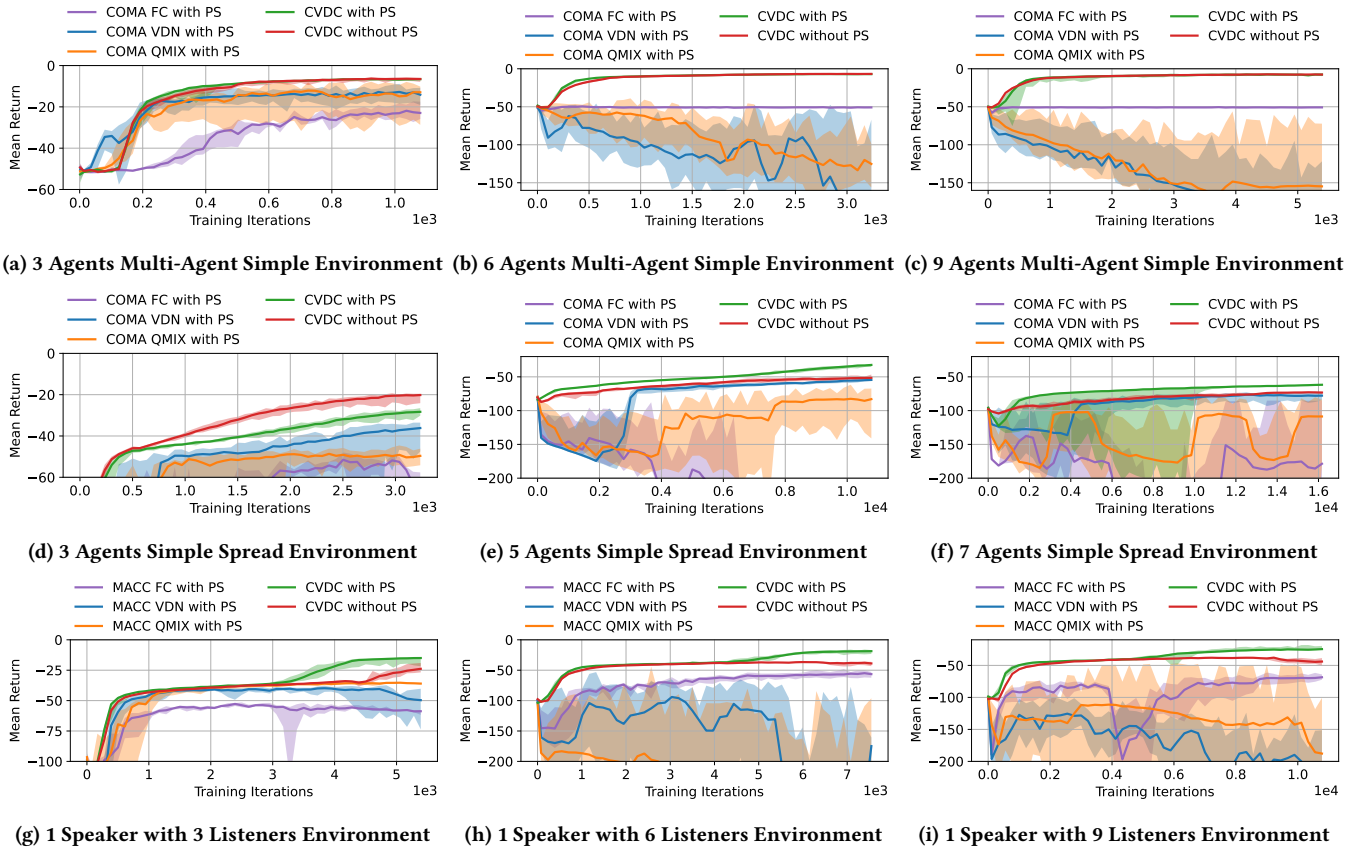
Proof.

$$\begin{aligned}
A^a(s_t, u_t) &= Q(s_t, u_t) - \mathbb{E}_{\pi^a}[Q(s_t, u_t)] \\
&= A(s_t, u_t) + V(s_t) - \mathbb{E}_{\pi^a}[A(s_t, u_t) + V(s_t)] \\
&= A(s_t, u_t) + V(s_t) - V(s_t) - \mathbb{E}_{\pi^a}[A(s_t, u_t)] \\
&= A(s_t, u_t) - \mathbb{E}_{\pi^a}[A^a(s_t, u_t) + A^{-a}(s_t, u_t)] \\
&= A(s_t, u_t) - \mathbb{E}_{\pi^a}[A^a(s_t, u_t)] - \mathbb{E}_{\pi^a}[A^{-a}(s_t, u_t)] \\
&\quad \text{(use Lemma 5.1)} \\
&= A(s_t, u_t) - \mathbb{E}_{\pi^a}[A^{-a}(s_t, u_t)]
\end{aligned} \tag{16}$$

□

6 EXPERIMENTS

In this Section, we describe the experiments aiming to compare our CVDC method with the different baselines. We approach these experiments as an ablation study. So the architecture and hyperparameters are identical between the different methods apart from the methods' architecture and hyperparameters for the critic. The three baselines use the COMA or MACC method to train the actors but only differ in neural network architecture. These architectures are a fully connected (FC) architecture, VDN [25] architecture and a QMIX [20] architecture. The baseline architectures are described in more detail in Section 3.4. The hyper-parameters for the different critics are found empirically for each method. However, the parameters for the actor are the same for the different critic methods for a given environment to only compare the difference in critic performance. The different baseline methods are trained using parameter sharing (PS) because they follow the CTDE paradigm, as discussed in the COMA [5] method. So to compare CVDC equally with the different baseline methods, we also used parameter sharing in the experiments. However, this means that CVDC no longer follows



(a) 3 Agents Multi-Agent Simple Environment (b) 6 Agents Multi-Agent Simple Environment (c) 9 Agents Multi-Agent Simple Environment

(d) 3 Agents Simple Spread Environment (e) 5 Agents Simple Spread Environment (f) 7 Agents Simple Spread Environment

(g) 1 Speaker with 3 Listeners Environment (h) 1 Speaker with 6 Listeners Environment (i) 1 Speaker with 9 Listeners Environment

Figure 4: Results for different experiments in the Particle Environment

the DTDE-FCC paradigm but also uses the CTDE paradigm. So to also evaluate CVDC in the DTDE-FCC paradigm, we also train the CVDC method with all parameter sharing disabled for the different critics and actors.

We evaluate these different methods and configurations in three different environments from the Multi Particle Environment (MPE) [12, 16], which Gorsane et al. [6] showed is one of the most popular MARL environments. The scenarios from the MPE are available in the PettingZoo project [28]. Since the number of agents (N) impacts the extent of the credit-assignment problem, we test every method on multiple values for N.

The first environment is a multi-agent version of the simple environment from the MPE. In the multi-agent simple environment, N agents need to navigate to their goal landmark in a 2D world based on the relative position of this landmark. The team reward is the average distance between the agents and their corresponding goal landmark. We selected this environment because the agent-specific value for the agents does not depend on actions or observations from other agents. These actions and observations are independent because the team reward is the sum of the agent-specific rewards, which makes it easy for the agents to factorise the global value.

The second environment is the simple-spread environment from the MPE. In this environment, the agents do not have a target landmark, but the N agents have to cover N landmarks. The agents do this while observing the relative position of the landmarks and the other agents. In this environment, the reward function is the sum of the minimum distance between all the landmarks and the agents. The environment is challenging compared to the multi-agent simple environment because the agent-specific value function is dependent on the other agents' actions making the non-stationarity problem difficult. This property makes the environment suitable to validate the different methods on their ability to handle an environment where the non-stationarity problem is pronounced.

The last environment is an adapted version of the speaker-listener environment from the MPE to support n listeners with a single speaker. The listener agents have to go towards their target landmark without knowing which of the landmarks is their target. The landmarks are uniquely identifiable because they have a specific colour. This target colour is identical between the different listener agents and is available within the observation of the speaker agent. The speaker aims to encode this information into a message and send it to the listener agents. The agents train with a shared team reward, which is defined as the average distance between the listeners and their landmarks. The speaker-listener environment is a jointly observable environment (Dec-MDP) where the agent-specific value function depends directly on the observation of another agent, making the training of a critic challenging. In this environment, the agents can achieve a mean return of -35

without communication by moving to the equidistant between the landmarks. The agents can obtain a mean return of -20 by learning a communication protocol. The speaker-listener environment requires the agents to communicate to achieve their goal, so the different baseline algorithms use the MACC method to learn this communication protocol.

7 RESULTS

The results of the experiments, as discussed in Section 6, are shown in Figure 4. Every method and environment combination is evaluated over five independent runs with a different random seed. These results are combined using the interquartile mean and the 90% confidence interval, using the Bootstrap method and bootstrapping 25 times. We used the interquartile mean instead of an Arithmetic mean to minimize the impact of the outliers.

The results show that our CVDC method with parameter sharing (PS) outperforms the baseline methods for all the evaluated environments. This performance difference is caused because we selected a set of environments that expose different problems the critics can encounter. The Multi-Agent Simple environment focuses on the scaling performance of the methods in an environment where the agents are easily decomposable. The Simple Spread environment is difficult for the agents because the optimal policy of an agent is impacted by the other agents' policies, which makes the environment appear non-stationary. Finally, the Speaker Listener environment is a Dec-MDP which requires the critics to combine information from different agents. These results show that the CVDC method can handle environments with a larger number of agents, minimize the non-stationarity problem and train within a Dec-MDP.

The fully connected critic architecture shows that it experiences scaling problems for environments with more agents because it suffers heavily from the non-stationarity problem. However, in a Dec-MDP environment (Speaker Listener), the FC architecture outperforms the other baselines for a larger number of agents. The FC architecture can achieve this performance because it has no problem combining information from different agents.

Next, when we increase the number of agents in the Multi-Agent Simple environment, the VDN and QMIX methods experience learning failures because of the combination of the credit-assignment problem and the non-stationarity problem. The VDN method achieves good results in the Simple Spread environment, which shows that this method can tolerate some amount of non-stationary. However, the QMIX method can not reach the same performance due to training instabilities when using more agents because the global state becomes more complex, making the mixing of the Q-values problematic. The VDN and QMIX methods do not work within a larger Dec-MDP environment because these methods cannot use observations of the other agents as effectively as a fully connected network, as shown in the Speaker Listener experiments.

Finally, we evaluated the CVDC method without parameter sharing for the critics and actors for the different agents. Doing so allows us to evaluate the CVDC method using the DTDE-FCC paradigm. The results show that CVDC without PS can outperform the other baseline methods. Even when comparing CVDC without PS to CVDC with PS, we see very comparable results. Only in the Speaker Listener environment is the CVDC method without PS unable to accomplish the same results as CVDC with PS. The PS is important in communication learning because when the receiving agents react differently to a received message, the sender has problems finding a communication protocol that works for the receiving agents. This difference is caused because, in this environment, the agents need to learn a communication protocol, which benefits from parameter sharing to learn a global communication protocol.

8 CONCLUSION AND FUTURE WORK

In this work, we introduced our novel CVDC method for value decomposition. This method aims to reduce the credit assignment problem, reduce the critics' non-stationarity problem and learn within a Dec-MDP. This while also training in a decentralised training and decentralised execution using a free critic communication channel paradigm in contrast to the state-of-the-art methods that use the centralised training and decentralised execution paradigm. Our CVDC method uses the insight that any Q-function is decomposable into a set of agent-specific Q-functions. The theoretical results show that this insight can be used to calculate the target advantage based on the state-of-the-art COMA method. These target advantages allow us to train a set of decomposed agent-specific critic networks that are as independent of the other agents as possible and can directly be used to learn the agent's policy. The experiments show that CVDC significantly outperforms the VDN and the QMIX methods. The state-of-the-art baseline methods suffer from the non-stationarity problem and credit assignment problem, which causes significant learning instabilities when using a large number of agents. The Speaker Listener experiments show that the architecture of MACC CVDC can use the observations from the other agents effectively to estimate the advantage when learning to communicate. In contrast to the baseline methods that do not succeed in using the observation information from the other agents effectively since their policies fail to learn to communicate. Additionally, we evaluated the CVDC method without parameter sharing to validate its performance under the decentralised training and decentralised execution using a free critic communication channel paradigm. The CVDC method without parameter sharing can achieve similar results to CVDC with parameter sharing in different environments apart from the Speaker Listener environment. These results from the Speaker Listener environment are caused by the problems when learning a communication protocol when the different agents are not synchronised using parameter sharing. However, the results show that CVDC without parameter sharing can outperform every baseline in the evaluated environments. In future work, we want to apply this method to different environments like the StarCraft Multi-Agent Challenge (SMAC) [21] and focus on limiting the amount of information shared between the critics.

ACKNOWLEDGMENTS

This work was supported by the Research Foundation Flanders (FWO) under Grant Number 1S94120N and Grant Number 1S12121N.

REFERENCES

[1] L. Busoniu, R. Babuska, and B. De Schutter. 2008. A Comprehensive Survey of Multiagent Reinforcement Learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 38, 2 (2008), 156–172.

[2] Jacopo Castellini, Sam Devlin, Frans A Oliehoek, and Rahul Savani. 2021. Difference Rewards Policy Gradients. (2021).

[3] Thomas Degris, Martha White, and Richard Sutton. 2012. Off-Policy Actor-Critic. In *International Conference on Machine Learning*.

[4] Jakob Foerster, Ioannis Alexandros Assael, Nando De Freitas, and Shimon Whiteson. 2016. Learning to communicate with deep multi-agent reinforcement learning. In *Advances in neural information processing systems*. 2137–2145.

[5] Jakob N Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. 2018. Counterfactual multi-agent policy gradients. In *Thirty-second AAAI conference on artificial intelligence*.

[6] Rihab Gorsane, Omayma Mahjoub, Ruan de Kock, Roland Dubb, Siddarth Singh, and Arnu Pretorius. 2022. Towards a standardised performance evaluation protocol for cooperative marl. *arXiv preprint arXiv:2209.10485* (2022).

[7] Shariq Iqbal, Christian A Schroeder de Witt, Bei Peng, Wendelin Böhmer, Shimon Whiteson, and Fei Sha. 2020. Ai-qmix: Attention and imagination for dynamic multi-agent reinforcement learning. *arXiv preprint arXiv:2006.04222* (2020).

[8] Shariq Iqbal and Fei Sha. 2019. Actor-attention-critic for multi-agent reinforcement learning. In *International conference on machine learning*. PMLR, 2961–2970.

[9] Emilio Jorge, Mikael Kågebäck, Fredrik D Johansson, and Emil Gustavsson. 2016. Learning to play guess who? and inventing a grounded language as a consequence. *arXiv preprint arXiv:1611.03218* (2016).

[10] Vijay Konda and John Tsitsiklis. 1999. Actor-critic algorithms. *Advances in neural information processing systems* 12 (1999).

[11] Iou-Jen Liu, Unnat Jain, Raymond A Yeh, and Alexander Schwing. 2021. Cooperative exploration for multi-agent deep reinforcement learning. In *International Conference on Machine Learning*. PMLR, 6826–6836.

[12] Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. 2017. Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments. *Neural Information Processing Systems (NIPS)* (2017).

[13] Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Timothy P. Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. 2016. Asynchronous Methods for Deep Reinforcement Learning. arXiv:1602.01783 [cs.LG]

[14] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. 2013. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602* (2013).

[15] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. 2015. Human-level control through deep reinforcement learning. *nature* 518, 7540 (2015), 529–533.

[16] Igor Mordatch and Pieter Abbeel. 2018. Emergence of grounded compositional language in multi-agent populations. 32, 1 (2018).

[17] Frans A Oliehoek, Christopher Amato, et al. 2016. *A concise introduction to decentralized POMDPs*. Vol. 1. Springer.

[18] Matthew L. Olson, Roli Khanna, Lawrence Neal, Fuxin Li, and Weng-Keen Wong. 2021. Counterfactual State Explanations for Reinforcement Learning Agents via Generative Deep Learning. *Artificial Intelligence* 295 (June 2021), 103455. https://doi.org/10.1016/j.artint.2021.103455

[19] Tabish Rashid, Gregory Farquhar, Bei Peng, and Shimon Whiteson. 2020. Weighted qmix: Expanding monotonic value function factorisation for deep multi-agent reinforcement learning. *Advances in neural information processing systems* 33 (2020), 10199–10210.

[20] Tabish Rashid, Mikayel Samvelyan, Christian Schroeder, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. 2018. Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning. In *International conference on machine learning*. PMLR, 4295–4304.

[21] Mikayel Samvelyan, Tabish Rashid, Christian Schroeder de Witt, Gregory Farquhar, Nantas Nardelli, Tim GJ Rudner, Chia-Man Hung, Philip HS Torr, Jakob Foerster, and Shimon Whiteson. 2019. The StarCraft Multi-Agent Challenge. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*. 2186–2188.

[22] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. 2015. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438* (2015).

[23] Hassam Ullah Sheikh and Ladislau Bölöni. 2020. Multi-agent reinforcement learning for problems with combined individual and team reward. In *2020 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 1–8.

[24] David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. 2014. Deterministic policy gradient algorithms. In *International conference on machine learning*. PMLR, 387–395.

[25] Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinicius Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Z. Leibo, Karl Tuyls, and Thore Graepel. 2018. Value-Decomposition Networks For Cooperative Multi-Agent Learning Based On Team Reward. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems* (Stockholm, Sweden) *(AAMAS '18)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 2085–2087.

[26] Richard S Sutton and Andrew G Barto. 2018. *Reinforcement learning: An introduction*. MIT press.

[27] Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. 1999. Policy gradient methods for reinforcement learning with function approximation. *Advances in neural information processing systems* 12 (1999).

[28] J Terry, Benjamin Black, Nathaniel Grammel, Mario Jayakumar, Ananth Hari, Ryan Sullivan, Luis S Santos, Clemens Dieffendahl, Caroline Horsch, Rodrigo Perez-Vicente, et al. 2021. Pettingzoo: Gym for multi-agent reinforcement learning. *Advances in Neural Information Processing Systems* 34 (2021), 15032–15043.

[29] Hado Van Hasselt, Arthur Guez, and David Silver. 2016. Deep reinforcement learning with double q-learning.

[30] Simon Vanneste, Astrid Vanneste, Kevin Mets, Tom De Schepper, Ali Anwar, Siegfried Mercelis, Steven Latré, and Peter Hellinckx. 2022. Learning to communicate using counterfactual reasoning. *Proc. of the Adaptive and Learning Agents Workshop (ALA 2022)*.

[31] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[32] Yihan Wang, Beining Han, Tonghan Wang, Heng Dong, and Chongjie Zhang. 2020. Dop: Off-policy multi-agent decomposed policy gradients. In *International Conference on Learning Representations*.

[33] Ziyu Wang, Tom Schaul, Matteo Hessel, Hado Hasselt, Marc Lanctot, and Nando Freitas. 2016. Dueling network architectures for deep reinforcement learning. , 1995–2003 pages.

[34] Lex Weaver and Nigel Tao. 2013. The optimal reward baseline for gradient-based reinforcement learning. *arXiv preprint arXiv:1301.2315* (2013).

[35] Jiachen Yang, Ang Li, Mehrdad Farajtabar, Peter Sunehag, Edward Hughes, and Hongyuan Zha. 2020. Learning to incentivize other learning agents. *Advances in Neural Information Processing Systems* 33 (2020), 15208–15219.

[36] Meng Zhou, Ziyu Liu, Pengwei Sui, Yixuan Li, and Yuk Ying Chung. 2020. Learning implicit credit assignment for cooperative multi-agent reinforcement learning. *Advances in Neural Information Processing Systems* 33 (2020), 11853–11864.