

Learning to Communicate Using Counterfactual Reasoning

Simon Vanneste,¹ Astrid Vanneste,¹ Kevin Mets,² Ali Anwar,¹ Siegfried Mercelis,¹ Steven Latré,²
Peter Hellinckx¹

¹ University of Antwerp - imec IDLab - Faculty of Applied Engineering Sint-Pietersvliet 7, 2000 Antwerp, Belgium

² University of Antwerp - imec IDLab - Department of Computer Science Sint-Pietersvliet 7, 2000 Antwerp, Belgium
simon.vanneste@uantwerpen.be, astrid.vanneste@uantwerpen.be, kevin.mets@uantwerpen.be, ali.anwar@uantwerpen.be,
siegfried.mercelis@uantwerpen.be, steven.latre@uantwerpen.be, peter.hellinckx@uantwerpen.be

Abstract

Learning to communicate in order to share state information is an active problem in the area of multi-agent reinforcement learning (MARL). The credit assignment problem, the non-stationarity of the communication environment and the creation of influenceable agents are major challenges within this research field which need to be overcome in order to learn a valid communication protocol. This paper introduces the novel multi-agent counterfactual communication learning (MACC) method which adapts counterfactual reasoning in order to overcome the credit assignment problem for communicating agents. Secondly, the non-stationarity of the communication environment while learning the communication Q-function is overcome by creating the communication Q-function using the action policy of the other agents and the Q-function of the action environment. Additionally, a social loss function is introduced in order to create influenceable agents which is required to learn a valid communication protocol. Our experiments show that MACC is able to outperform the state-of-the-art baselines in four different scenarios in the Particle environment.

1 Introduction

A lot of research has been done towards single-agent reinforcement learning (Mnih et al. 2013, 2016; Sutton and Barto 1998; van Hasselt, Guez, and Silver 2015; Wang et al. 2015). However, many of the practical applications can naturally be described as multi-agent systems such as industrial robotics and network packet routing. In the field of multi-agent reinforcement learning (MARL), a great deal of literature is available (Busoniu, Babuska, and De Schutter 2008). The agents within a multi-agent system can encounter partial observability due to the individual agents only having access to their individual observations which can make it hard or even impossible to find an effective action policy. Cooperative multi-agent systems can be extended with inter-agent communication to overcome this issue. The communication allows the agent to share information about their individual observation which reduces the complexity of the environment. The action and communication policy are learned simultaneously using the shared team reward.

In the domain of multi-agent reinforcement learning using inter-agent communication several problems arise. In this work, we focus on the credit assignment problem when multiple agents in a multi-agent system are acting and commu-

nicating while only receiving a single team reward as feedback. We use a centralized critic to evaluate the actions and messages of a single agent. This work also focuses on the non-stationarity that arises when agents are trying to learn a communication policy but the policy of the receiving agents changes during the training process. This non-stationarity makes it very difficult to learn the Q-function of the centralized critic for the communication policy. This work uses the centralized training and decentralized execution (CTDE) paradigm which allows us to train a centralized critic that has access to the policy of the different agents. We also assume a non-differentiable discrete communication setting (e.g. $\{0, 1, 2, 3\}$) in which it takes one timestep to receive a message which was sent by another agent. However, with a few modifications it is possible to change this to instant communication or towards communication with a longer delay.

In this paper, we present multi-agent counterfactual communication (MACC) learning, a reinforcement learning method to simultaneously learn to act and communicate in a multi-agent environment. Multi-agent counterfactual reasoning for the action policy, in order to overcome the credit assignment problem, has already been described and used by Foerster et al. (2018) in their COMA method. MACC extends COMA by using counterfactual reasoning to learn a communication protocol using a centralized critic. This critic requires an additional novel communication Q-function which is created from the action Q-function and the policies of the other agents using a novel decomposition into two separate communication Q-functions. This minimizes the non-stationarity of the communication environment for the communication Q-function. Additionally, a social loss function is introduced for the action policy in order to promote social behaviour and thereby improve the learning stability. MACC is evaluated and compared against MADDPG (Lowe et al. 2017) and COMA (Foerster et al. 2018) in four different scenarios of the Particle environment from OpenAI (Lowe et al. 2017; Mordatch and Abbeel 2017). We chose these methods because they most closely match our method and the problem setting that we target.

This paper is structured as follows. In Section 2 of this paper, we discuss relevant literature to our work. Section 3 provides background in Markov Decision Processes and counterfactual reasoning (Foerster et al. 2018). In Section 4, we explain MACC in detail. Section 5 shows the differ-

ent experiments we performed to demonstrate our method. Finally, our conclusions and future work are described in Section 6.

2 Related Work

Recently, several different models for multi-agent communication learning have been presented. The foundations in this research field were laid by Foerster et al. (2016) and Sukhbaatar, Szlam, and Fergus (2016). Foerster et al. (2016) proposed two different methods. In Reinforced Inter Agent Learning (RIAL) the communication policy is learned by applying the reward in the same way as we do for the action policy. However, the best results were achieved with Differentiable Inter Agent Learning (DIAL) which uses gradient feedback from the receiving agent to learn the communication policy. Sukhbaatar, Szlam, and Fergus (2016) use a similar technique in CommNet. However, CommNet uses continuous communication instead of discrete communication and assumes multiple steps of communication can occur before the agents take an action. These fundamental works were followed by more research in the field of multi-agent communication learning. Peng et al. (2017) presented BiC-Net, an actor-critic model that is able to play real-time strategy games such as StarCraft. Mao et al. (2017) proposed two methods, one where communication between actors is learned and one where communication between critics is learned. In contrast to our method, both Peng et al. (2017) and Mao et al. (2017) use a separate local critic for each of the agents.

A lot of recent work investigates non-broadcast communication (Jiang and Lu 2018; Das et al. 2018; Ding, Huang, and Lu 2020). Our experiments use broadcast communication but Section 4 describes our methods with attention for the use of non-broadcast communication. Other recent advances include the work of Ossenkopf, Jorgensen, and Geihs (2019). They use hierarchical deep MARL to improve the long-term coordination of agents. Vanneste et al. (2020) target the lazy agent problem in communication learning by applying value decomposition (Sunehag et al. 2018) on DIAL (Foerster et al. 2016), resulting in improved learning speed and performance. In the multi-agent deep deterministic policy gradient (MADDPG) method, Lowe et al. (2017) introduced the idea of using a centralized critic in MARL instead of a separate critic for each agent. They adapted deep deterministic policy gradient (Silver et al. 2014) by using this centralized critic. They learn communication as a part of the action policy which results in a total action space determined by the action space times the communication space. This is not the case for MACC since it uses a separate action and communication policy. Simões, Lau, and Paulo Reis (2020) introduce Asynchronous Advantage Actor Centralized-Critic with Communication (A3C3), a method based on the single agent Asynchronous Advantage Actor-Critic (A3C) method (Mnih et al. 2016). A3C3 uses a centralized critic to make a value estimation of a centralized observation. The communication network is optimized by propagating the gradients of the receiving actors through the communication network.

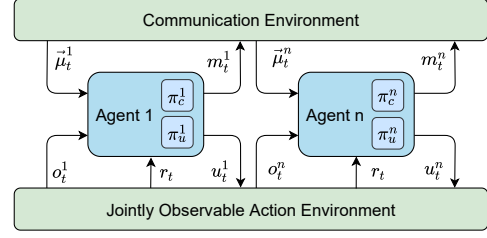


Figure 1: **Dec-MDP** with a separate action environment and communication environment.

Jaques et al. (2019) propose a method that uses social influence as an extra component of the communication reward. The communication policy is trained by a reward composed of the sum of the team reward and a social influence reward. This reward is calculated by determining how much the message influenced the action choice of the other agent. The ideas behind the work of Jaques et al. (2019) and our work seem similar. However, there are major differences. Jaques et al. (2019) still use the team reward and purely looks at the change in action distribution. However, a change in the action distribution does not necessarily result in an improved communication protocol. Instead of using the change in action distribution, we use the action distribution in combination with the action Q-function to determine if a message will result in improved performance.

3 Background

In this section, the Markov Decision Process for multi-agent communication and counterfactual reasoning are discussed. In this work, we use the notation where the superscript indicates the index of a certain agent. The index uses the notation a for the current agent and $-a$ for every agent except the current agent. The subscript is used to indicate if a symbol is used for the action u or communication c environment or to indicate the timestep t . The notation $s_{t,t+1}$ is used as an abbreviated notation for s_t, s_{t+1} .

Markov Decision Processes

In this paper, we use the decentralized Markov Decision Process framework (Dec-MDP) (Oliehoek, Amato et al. 2016) extended with communication as shown in Figure 1. In Dec-MDPs, n agents learn their policy based on the global team reward r . Every agent only receives a partial observation o of the full state s . In addition to the jointly observable action environment, a communication environment is added. Each time-step t agent a receives, alongside the observation o_t^a from the action environment, a number of received messages $\bar{\mu}_t^a$ from the communication environment. The action policy of the agent $\pi_u^a(u_t^a | o_t^a, \bar{\mu}_t^a)$ takes these inputs and samples an action u_t^a . These actions are then processed by the environment and a team reward r_t is given to the agents. The communication policy of the agent $\pi_c^a(m_t^a | o_t^a, \bar{\mu}_t^a)$ uses the observation and received messages to generate an outgoing message m_t^a . The messages are processed by the communication environment to get the input messages for the

next time-step $\bar{\mu}_{t+1} = M(m_t)$. The communication function M determines who receives which messages at the next time-step and $\bar{\mu}_{t+1}^a = M^a(m_t)$ is the part of the communication environment which determines what messages are received by agent a . When combining the policies of all agents we can describe the joint action policy $\pi_u(u_t|o_t, \bar{\mu}_t)$ and the joint communication policy $\pi_c(m_t|o_t, \bar{\mu}_t^a)$. In our work, we allow the agents to only operate in either the action or the communication environment which results in agents that cannot communicate and agents that cannot perform actions respectively.

Counterfactual Multi-Agent Policy Gradient

Foerster et al. (2018) showed that policy gradient agents in a multi-agent system can be trained using a centralised critic. This centralised critic is able to predict the joint state-action utility $Q_u(s_t, u_t)$ which can be used to calculate the advantage for the action environment A_u^a . This calculation uses counterfactual reasoning by subtracting the expected utility of the action policy $V_u^a(s_t, u_t^a)$ from the state-action utility. The expected utility can be expanded into the marginalization over the counterfactual actions u_t^a of the action policy π_u^a which is multiplied with the joint action Q-value of that action permutation. Equation 1 and Equation 2 show the advantage calculation for the action policy of agent a as described by Foerster et al. (2018).

$$A_u^a(s_t, u_t) = Q_u(s_t, u_t) - V_u^a(s_t, u_t^a) \quad (1)$$

$$V_u^a(s_t, u_t^a) = \sum_{u_t^a} \left(\pi_u^a(u_t^a | o_t^a) Q_u(s_t, (u_t^{-a}, u_t^a)) \right) \quad (2)$$

The actions of the other agents u_t^{-a} are constant during the marginalization so that the agent only reasons about its own actions. The action advantage function A_u^a calculates the advantage for the action u_t^a using the joint action Q-function. The joint action Q-function is learned by the centralized critic during training.

4 Methods

In this section, we discuss the multi-agent counterfactual communication (MACC) learning method. Figure 2 shows the high level architecture of MACC. In MACC, multiple actors act in the environment and communicate with each other. The centralized critic learns a joint action Q-function based on the current state and the received rewards and calculates the action and communication advantage for both policies using the action and communication policies of the different agents. The counterfactual reasoning for the action policy will be identical to the COMA calculations with the adaptation to also include incoming messages $\bar{\mu}_t^a$ into the action policy $\pi_u^a(u_t^a | o_t^a, \bar{\mu}_t^a)$, the utility function $V_u^a(s_t, u_t^a | \bar{\mu}_t^a)$ and the advantage function $A_u^a(s_t, u_t, \bar{\mu}_t)$.

Counterfactual Reasoning in the Communication Environment

We adapted the counterfactual reasoning process to work within the communication environment. This is shown in

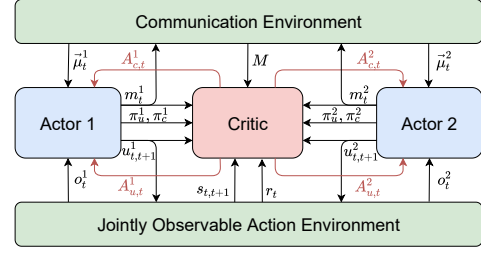


Figure 2: The MACC architecture using a centralised critic.

Equation 3 and 4. First, the state s_t is expanded to $s_{t,t+1}$ to include $t + 1$ which is required when messages take one timestep to arrive. The $t + 1$ timestep can be replaced with t for instant communication and with $t + N$ for communication delayed by N timesteps. Next, the action Q-function $Q_u(s_t, u_t)$ is replaced by a communication Q-function $Q_c(s_{t,t+1}, m_t)$. This communication Q-function is discussed in detail in the next section.

$$A_c^a(s_{t,t+1}, \bar{\mu}_t, m_t) = Q_c(s_{t,t+1}, m_t) - V_c^a(s_{t,t+1}, \bar{\mu}_t, m_t^a) \quad (3)$$

$$V_c^a(s_{t,t+1}, \bar{\mu}_t, m_t^a) = \sum_{m_t^a} \left(\pi_c^a(m_t^a | o_t^a, \bar{\mu}_t^a) \cdot Q_c(s_{t,t+1}, (m_t^{-a}, m_t^a)) \right) \quad (4)$$

Communication Q-function

The communication environment appears non-stationary because the utility of a message changes when the action policy of the other agent changes during training. In this work, the communication Q-function Q_c is composed out of the communication to action Q-function Q_{cu} (the impact on other action policies) and the discounted communication to communication Q-function Q_{cc} (the impact on other communication policies). This decomposition splits the joint communication Q-function into two separate Q-functions to model the impact of a message for the action policy separate from the impact of a message on the other communication policies. This is shown in Equation 5. The communication discount factor γ_c is used to limit the impact of a message on the communication Q-function.

$$Q_c(s_{t,t+1}, m_t) = Q_{cu}(s_{t,t+1}, m_t) + \gamma_c Q_{cc}(s_{t,t+1}, m_t) \quad (5)$$

The communication to action Q-function Q_{cu} represents the expected return when a message is sent to the action policy of the other agents. Equation 6 shows the communication to action Q-function calculations. In these calculations, the critic reasons about the impact of a message on the action policy of an agent which performs an action in the action environment at $t + 1$. The quality of this action, based on messages m_t , can be calculated using the action Q-function Q_u .

Algorithm 1: MACC

Initialise $\theta_u, \theta_u^-, \theta_{\pi_u}, \theta_{\pi_c}$
for each episode **do**
 $s_0 = \text{initial state}, t = 1$
 $\mu_0^a = 0$ for each agent a
while $s_t \neq \text{terminal}$ **and** $t < T$ **do**
for each agent a **do**
 $u_t^a \sim \pi_u^a(o_t^a, \mu_t^a, \theta_{\pi_u}^a)$
 $m_t^a \sim \pi_c^a(o_t^a, \mu_t^a, \theta_{\pi_c}^a)$
end for
 $\tilde{\mu}_{t+1} = M(m_t)$
 Get s_{t+1}, r_t from the environment
 $t = t + 1$
end while
 Calculate Q_u^a, Q_{cu}^a for each agent a
 $Q_{c,t-1}^a = 0$ for each agent a
for $j = t - 2$ **to** 0 **do**
for each agent a **do**
 Calculate $Q_{cc,j}^a$ using $Q_{cu,j+1}^a, Q_{c,j+1}^a$
 $Q_{c,j}^a = Q_{cu,j}^a + \gamma_c Q_{cc,j}^a$
end for
end for
 Calculate A_u, A_c, V_u, V_c using Q_u, Q_c
 Update θ_u, θ_u^- using r
 Update $\theta_{\pi_u}, \theta_{\pi_c}$ using Q_u, Q_c
end for

$$Q_{cu}(s_{t,t+1}, m_t) = \sum_{u'_{t+1}} (Q_u(s_{t+1}, u'_{t+1}) \cdot \prod_a \pi_u^a(u'_{t+1} | o_{t+1}^a, M^a(m_t))) \quad (6)$$

Equation 7 shows the communication to communication Q-function Q_{cc} calculations. In this calculation, the expected future communication Q-value is calculated for message m_t which allows MACC to reason about the impact of a message on other communication policies which send messages to other agents. As Q_{cc} depends on Q_c , the number of calculations grows exponentially. This can be overcome by calculating the communication Q-function for a certain episode from $t = T - 2$ to $t = 0$ and using the already calculated Q_c values for these timesteps. Algorithm 1 shows the full MACC algorithm including the Q_{cc} calculations.

$$Q_{cc}(s_{t,t+1}, m_t) = \sum_a \sum_{m'_{t+1}} (Q_c(s_{t+1,t+2}, m'_{t+1}) \cdot \pi_c^a(m'_{t+1} | o_{t+1}^a, M^a(m_t))) \quad (7)$$

Action Q-function

In order to calculate the advantages for the action policy, the critic for the action policy needs to learn the joint action Q-function $Q_u(s, u, \theta_u)$ which is represented by the neural network parameters θ_u . This neural network is trained

by minimising the loss function from Equation 8. The loss function uses a target network θ_u^{i-} (Mnih et al. 2015), which is a delayed version of the θ_u^i parameters, in order to stabilize the training of the action Q-function. The loss function uses the observed future actions u' (the SARSA update rule as described by Rumery and Niranjan (1994), Sutton and Barto (1998)) instead of using the actions that maximize the Q-function ($\max_{u'} Q_u(s', u', \theta_u)$) in order to minimize the number of inference calls to the action Q-network when the joint action space is large (due to the large number of action permutation over the different agents). In this work, we use a replay buffer to reuse past experiences to train the joint action Q-function. The joint action loss function is on-policy which cannot be used in combination with a replay buffer unless the actions u'_{t+1} are resampled based on the current policy $u'_{t+1} \sim \pi_u(o_{t+1}, M(m'_t))$. However, the MACC action policies can depend on the received messages which depend on the communication policy. This means that the message m'_t also needs to be resampled from the communication policies $m'_t \sim \pi_c(o_t, \mu_t)$. The regularization term within the loss function is controlled by the hyperparameter δ .

$$\mathcal{L}_i(\theta_u^i) = \mathbb{E}_{s_t, u_t, r_t, s_{t+1}} [(r_t + \gamma Q_u(s_{t+1}, u'_{t+1}, \theta_u^{i-}) - Q_u(s_t, u_t, \theta_u^i))^2] + \delta \sum_n |\theta_u^{i,n}| \quad (8)$$

Social Loss

The MACC methods use the change in the future expected reward through the choice of alternative actions in function of an input message to learn a communication policy. However, when the action policy ignores the received messages, it becomes impossible to learn a valid communication protocol which in turn does not give the action policy any incentive to use the received messages. In order to promote social behaviour from the action policy, which in this context means taking different actions based on the messages of other agents, an additional social loss function is added to the action policy loss function. This loss function encourages the action policy to have a different policy distribution for different received messages $\tilde{\mu}'$. The social loss function is shown in Equation 9. This loss function will iterate over the different input message combinations and increase the loss function when the action distribution is similar in order to promote social behaviour.

$$\mathcal{L}_i^s(\theta_{\pi_u}^i) = - \frac{\lambda}{k * (k - 1)} \cdot \sum_{x,y=0}^k |\pi_u^a(o^a, \tilde{\mu}'_x, \theta_{\pi_u}^i) - \pi_u^a(o^a, \tilde{\mu}'_y, \theta_{\pi_u}^i)| \quad (9)$$

Note that this social loss is an additional loss function which can be controlled by the hyperparameter λ which allows us to tune the social loss loss in order to encourage social behaviour without removing the possibility for the agent to ignore the messages when this is required in a certain environment. Additionally, if the sending agent cannot improve the

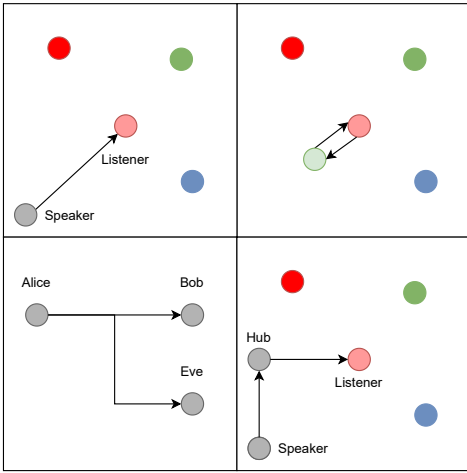


Figure 3: The four different Particle environment scenarios in which the arrows represent the communication topology. The top left figure shows the Speaker Listener scenario. The Simple reference scenario is visualized in the top right. The bottom left shows the Simple Crypto scenario. The Speaker listener with Communication Hub is show in the bottom right.

team reward by sending different messages, the communication policy learns to always send the same message in order to promote the best action distribution.

5 Experiments

Our experiments use different scenarios of the particle environment (Lowe et al. 2017) where multiple agent are simulated in a 2D environment. These environments are often used to benchmark MARL systems in order to validate different communication topologies. Additionally, we also propose a novel scenario based on the Speaker Listener scenario, the Speaker Listener with Communication Hub scenario. These scenarios are selected to validate different multi-agent communication configurations using the RLLib framework (Liang et al. 2018). In these experiments, MACC is evaluated with and without the social loss ($\lambda = 0$) and compared with COMA (Foerster et al. 2018) and the RLLib MADDPG (Lowe et al. 2017) implementation because these methods target the credit assignment problem, use a centralized critic and can be used to train discrete non-differentiable inter-agent communication. We used the same hyperparameters for MADDPG as in the work of Lowe et al. (2017) and determined the hyperparameters for the other methods empirically and by using a grid search. The results of our experiments are the average of five training runs for every method.

Speaker Listener

In the speaker listener scenario (see Figure 3), two agents have to work together to reach a goal. The environment has three different random goals with a color for every goal. The listener agent needs to navigate to a certain goal but has no

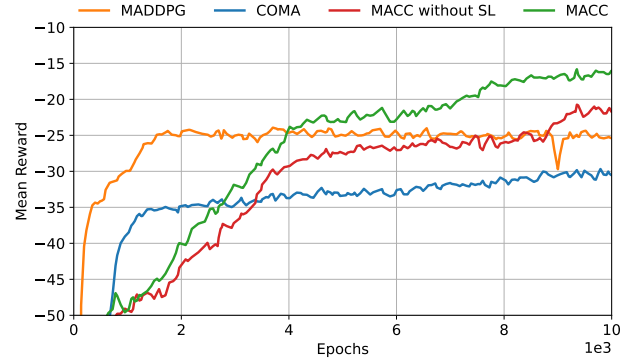


Figure 4: The mean episode reward during training for the speaker listener environment.

Table 1: The mean episode reward and standard deviation of the reward for the speaker listener environment between epoch 9e3 and 10e3.

	Mean Reward	Std Reward
MACC	-16.60	1.32
MACC without SL	-21.74	6.27
COMA	-30.35	3.78
MADDPG	-25.35	5.97

information about which goal is the target goal. This information is only available to the speaker agent. The speaker agent needs to learn to encode this information by sending one of the four possible discrete messages to the listener agent. The reward is determined by the distance of the listener agent to the target goal. In this environment, a reward of around -15 shows that the agents have learned a valid communication protocol and the listener agent is able to navigate to the correct target goal location. A reward of -35 is achievable without inter-agent communication.

The training results of the different methods are shown in Figure 4 and Table 1. These results show that MADDPG is able to learn a basic communicate protocol while learning this much faster than the other methods. However the high standard deviation shows that this method can get stuck in a local optimum and the method is not able to reach the -15 reward. COMA is only able to learn a limited form of communication because the centralized critic is suffering from the non-stationarity of the communication environment. MACC and MACC without social loss are able to outperform MADDPG and COMA. However, MACC without social loss has a lower average reward and a higher standard deviation than MACC with social loss. These results show that the social loss function prevents the MACC agents from converging towards a local optimum.

Simple Reference

The simple reference scenario (see Figure 3) is an extension to the speaker listener scenario where both agents have to go to a random target goal and share the target goal of the

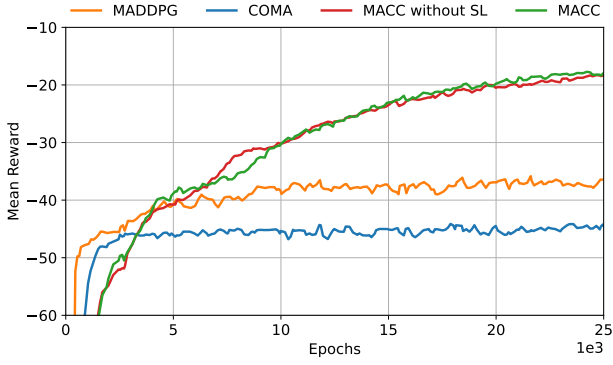


Figure 5: The mean episode reward during training for the simple reference environment.

Table 2: The mean episode reward and standard deviation of the reward for the simple reference environment between epoch 24e3 and 25e3.

	Mean Reward	Std Reward
MACC	-18.08	5.20
MACC without SL	-18.41	5.31
COMA	-44.56	2.84
MADDPG	-37.24	2.68

other agent with that agent. Both agents can have a different random target goal in order to make sure the agents need to learn to communicate. Since both agents need to learn the same behaviour, the neural network parameters are shared across the agents in this scenario. The reward is based on the average distance of both agents to their target goal. The agents can achieve a reward of -15 when they are communicating and -35 when they cannot communicate. This is a more challenging environment compared to the speaker listener environment because of the credit assignment problem for two agents that simultaneously act and communicate using a single team reward. Additionally, the action space of the agents is increased as both agents need to both act and communicate.

The results for the simple reference scenario are presented in Figure 5 and Table 2. MADDPG is able to learn a valid action policy but is not able to learn a communication protocol since the average reward is too low. COMA is not able to learn a valid action or communication policy because of learning instability caused by the non-stationary communication environment, the credit assignment problem and the large action space of the agents. This environment is challenging for MADDPG and COMA since the action space is the number of actions times the number of possible output messages as they do not have a separate communication policy. This shows an important drawback of MADDPG and COMA. However, since MACC splits the action and communication policy, it is able to learn valid communication protocols both with and without SL. We hypothesise that the similar performance between MACC and MACC with-

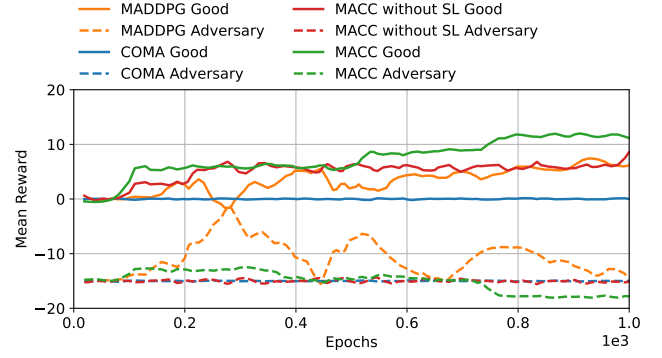


Figure 6: The mean episode reward during training for the simple crypto environment.

Table 3: The mean episode reward and standard deviation of the reward for the simple crypto environment between epoch 9e2 and 10e2.

	Mean Reward	Std Reward
MACC	11.62	4.09
MACC without SL	6.34	5.09
COMA	0.04	0.28
MADDPG	6.69	4.37

out SL is due to the parameter sharing between the agents. This causes the agent to learn with a bigger batch preventing the convergence towards a local optima.

Simple Crypto

The simple crypto scenario (see Figure 3) is a mixed cooperative-competitive environment where Alice needs learn to encrypt one bit of information into a message (discrete message with four possible symbols) which is shared with Bob. If Bob can correctly decrypt this bit of information, both Alice and Bob receive a reward of 2. A reward of -2 is presented when Bob incorrectly decrypts the message. A third adversarial agent (Eve) also tries to decrypt the information. Eve is rewarded 0 for decrypting the message and -2 for not decrypting the message correctly. When Eve can decrypt the message, 2 is subtracted from the reward of Alice and Bob. Both Alice and Bob have access to the same shared key during an episode (of length 30) which can be used to encrypt and decrypt the single bit of information. The encryption and decryption need to be part of the communication and action policy respectively. Alice and Bob are grouped as the good agents and Eve as the adversary. The good agents can achieve a reward of 15 when they can successfully encrypt and decrypt a message without the adversary agent being able to decrypt the message (resulting in random guessing). The reward range for the adversary agent is between 0 and -30. In this environment, a separate centralized critic will be trained for each team as they will have a different reward function. The goal of this experiment is not to propose a new encryption method but to validate commu-

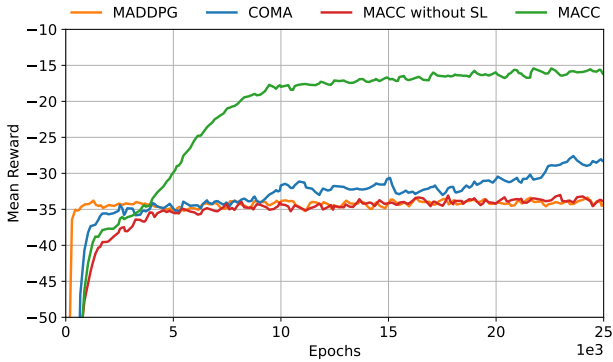


Figure 7: The mean episode reward during training for the speaker listener with communication hub environment.

Table 4: The mean episode reward and standard deviation of the reward for the speaker listener with communication hub environment between epoch 24e3 and 25e3.

	Mean Reward	Std Reward
MACC	-15.75	2.92
MACC without SL	-33.87	1.75
COMA	-28.47	3.30
MADDPG	-33.95	1.80

nication learning in a mixed cooperative-competitive environment.

In this experiment, we use the good agents to evaluate the different methods as the goal is to learn a valid communication and encryption policy. The results of the simple crypto experiments are shown in Figure 6 and Table 3. The MADDPG method is able to learn an encryption and decryption policy however the large standard deviation shows that not every training run is successful. The COMA method is not able to learn an encryption and decryption policy because of the non-stationarity of the communication environment. MACC without social loss is able to achieve similar performance to MADDPG and has a similarly large standard deviation which indicates that some of the training runs fail and converge towards a local optimum. Finally, the MACC method is able to outperform the other methods with a higher mean reward. Nevertheless, the high standard deviation, due to the high non-stationarity of the competitive environment, still indicates that some future improvements are required for these kinds of environments.

Speaker Listener with Communication Hub

The speaker listener with communication hub (see Figure 3) is a custom adapted version of the speaker listener scenario in which the speaker agent shares the target goal location with the communication hub agent. This agent processes this information and sends a new message to the listener agent. This agent then goes to the target location. The team reward depends on the distance between the listener and the goal location identical to the speaker listener environment. This

is a very challenging environment as the communication environment becomes even more non-stationary since an additional policy changes the message between the speaker and listener. Additionally, the messages take an extra timestep to reach the listener and the credit assignment becomes more challenging.

The results from the speaker listener with communication hub experiments are shown in Figure 7 and Table 4. These results show that MADDPG and MACC without social loss did not learn a valid communication protocol and have converged towards a local optimum in which communication is not used. This local optimum is created by the combination of the non-stationary communication environment and the hub policy which can disrupt the communication between the speaker and listener. COMA is able to slightly outperform both MADDPG and MACC without social loss. In certain trainings runs, COMA is able to learn a combination of policies which can share a single bit of information which can be used to navigate to the target location. MACC is able to learn a valid communication protocol by using the communication to communication Q-function Q_{cc} with a $\gamma_c = 0.9$ and a high social loss of 0.1 for the communication hub allowing this policy to map a received message onto a unique output message.

6 Conclusion

In this paper, we presented a novel method for **multi-agent communication learning called MACC** which is able to reason about the impact of a certain action or message using counterfactual reasoning by using a centralised critic. The critic is able to reason about the impact of a certain message on both the action and communication policy. In the experiments, MACC and MACC without social loss are compared with MADDPG and COMA, which also use a centralized critic, in four different scenarios in the Particle environment. These experiments show that MACC is able to learn a valid communication protocol in a range of different communication scenarios and can outperform both MADDPG and COMA. The social loss can be used by MACC to promote social behaviour in the receiving agent. This social behaviour allows the agents to experiment with action and communication policies. The results empirically show that MACC without the social loss function can get stuck in local optima because the receiving agent is ignoring certain input message which prevents the agents from learning a valid communication policy. In this paper, we limited our experiments to environments with only three agents. In the future, this research needs to be extended to environments with more agents which increases the computational complexity of MACC. Additionally, we also believe that the social loss function could be improved in order to further limit the amount of local optima which are still possible in certain environments as demonstrated in the simple crypto environment. In this work, we also assumed to have full access to the policy of every agent. This requirement could be relaxed by learning a model of the policy of the other agents.

7 Acknowledgments

This work was supported by the Research Foundation Flanders (FWO) under Grant Number 1S94120N and Grant Number 1S12121N. We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research.

References

- Busoniu, L.; Babuska, R.; and De Schutter, B. 2008. A Comprehensive Survey of Multiagent Reinforcement Learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 38(2): 156–172.
- Das, A.; Gervet, T.; Romoff, J.; Batra, D.; Parikh, D.; Rabbat, M.; and Pineau, J. 2018. TarMAC: Targeted Multi-Agent Communication. *arXiv:1810.11187*.
- Ding, Z.; Huang, T.; and Lu, Z. 2020. Learning individually inferred communication for multi-agent cooperation. *arXiv preprint arXiv:2006.06455*.
- Foerster, J.; Assael, I. A.; De Freitas, N.; and Whiteson, S. 2016. Learning to communicate with deep multi-agent reinforcement learning. In *Advances in neural information processing systems*, 2137–2145.
- Foerster, J. N.; Farquhar, G.; Afouras, T.; Nardelli, N.; and Whiteson, S. 2018. Counterfactual multi-agent policy gradients. In *Thirty-second AAAI conference on artificial intelligence*.
- Jacques, N.; Lazaridou, A.; Hughes, E.; Gulcehre, C.; Ortega, P.; Strouse, D.; Leibo, J. Z.; and De Freitas, N. 2019. Social Influence as Intrinsic Motivation for Multi-Agent Deep Reinforcement Learning. In *International Conference on Machine Learning*, 3040–3049.
- Jiang, J.; and Lu, Z. 2018. Learning Attentional Communication for Multi-Agent Cooperation. In Bengio, S.; Wallach, H.; Larochelle, H.; Grauman, K.; Cesa-Bianchi, N.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 31*, 7254–7264. Curran Associates, Inc.
- Liang, E.; Liaw, R.; Nishihara, R.; Moritz, P.; Fox, R.; Goldberg, K.; Gonzalez, J. E.; Jordan, M. I.; and Stoica, I. 2018. RLlib: Abstractions for Distributed Reinforcement Learning. In *International Conference on Machine Learning (ICML)*.
- Lowe, R.; Wu, Y.; Tamar, A.; Harb, J.; Abbeel, P.; and Mordatch, I. 2017. Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments. *Neural Information Processing Systems (NIPS)*.
- Mao, H.; Gong, Z.; Ni, Y.; and Xiao, Z. 2017. ACCNet: Actor-Coordinator-Critic Net for "Learning-to-Communicate" with Deep Multi-agent Reinforcement Learning. *arXiv:1706.03235*.
- Mnih, V.; Badia, A. P.; Mirza, M.; Graves, A.; Lillicrap, T. P.; Harley, T.; Silver, D.; and Kavukcuoglu, K. 2016. Asynchronous Methods for Deep Reinforcement Learning. *arXiv:1602.01783*.
- Mnih, V.; Kavukcuoglu, K.; Silver, D.; Graves, A.; Antonoglou, I.; Wierstra, D.; and Riedmiller, M. 2013. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*.
- Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A. A.; Veness, J.; Bellemare, M. G.; Graves, A.; Riedmiller, M.; Fidjeland, A. K.; Ostrovski, G.; et al. 2015. Human-level control through deep reinforcement learning. *nature*, 518(7540): 529–533.
- Mordatch, I.; and Abbeel, P. 2017. Emergence of Grounded Compositional Language in Multi-Agent Populations. *arXiv preprint arXiv:1703.04908*.
- Oliehoek, F. A.; Amato, C.; et al. 2016. *A concise introduction to decentralized POMDPs*, volume 1. Springer.
- Ossenkopf, M.; Jorgensen, M.; and Geihs, K. 2019. Hierarchical Multi-Agent Deep Reinforcement Learning to Develop Long-Term Coordination. In *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing, SAC '19*, 922–929. New York, NY, USA: Association for Computing Machinery. ISBN 9781450359337.
- Peng, P.; Yuan, Q.; Wen, Y.; Yang, Y.; Tang, Z.; Long, H.; and Wang, J. 2017. Multiagent Bidirectionally-Coordinated Nets for Learning to Play StarCraft Combat Games. *CoRR*, abs/1703.10069.
- Rummery, G. A.; and Niranjan, M. 1994. *On-line Q-learning using connectionist systems*, volume 37. University of Cambridge, Department of Engineering Cambridge, UK.
- Silver, D.; Lever, G.; Heess, N.; Degris, T.; Wierstra, D.; and Riedmiller, M. 2014. Deterministic policy gradient algorithms. In *International conference on machine learning*, 387–395. PMLR.
- Simões, D.; Lau, N.; and Paulo Reis, L. 2020. Multi-agent actor centralized-critic with communication. *Neurocomputing*, 390: 40–56.
- Sukhbaatar, S.; Szlam, A.; and Fergus, R. 2016. Learning Multiagent Communication with Backpropagation. In Lee, D.; Sugiyama, M.; Luxburg, U.; Guyon, I.; and Garnett, R., eds., *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc.
- Sunehag, P.; Lever, G.; Gruslys, A.; Czarnecki, W. M.; Zambaldi, V.; Jaderberg, M.; Lanctot, M.; Sonnerat, N.; Leibo, J. Z.; Tuyls, K.; and Graepel, T. 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, AAMAS '18*, 2085–2087. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems.
- Sutton, R. S.; and Barto, A. G. 1998. *Introduction to Reinforcement Learning*. Cambridge, MA, USA: MIT Press, 1st edition. ISBN 0262193981.
- van Hasselt, H.; Guez, A.; and Silver, D. 2015. Deep Reinforcement Learning with Double Q-learning. *arXiv:1509.06461*.
- Vanneste, S.; Vanneste, A.; Bosmans, S.; Mercelis, S.; and Hellinckx, P. 2020. Learning to Communicate with Multi-agent Reinforcement Learning Using Value-Decomposition Networks. In Barolli, L.; Hellinckx, P.; and Natwichai, J., eds., *Advances on P2P, Parallel, Grid, Cloud and Internet Computing*, 736–745. Cham: Springer International Publishing. ISBN 978-3-030-33509-0.

Wang, Z.; Schaul, T.; Hessel, M.; van Hasselt, H.; Lanctot, M.; and de Freitas, N. 2015. Dueling Network Architectures for Deep Reinforcement Learning. arXiv:1511.06581.

A Hyperparameters

The hyperparameters for the different experiments are shown in Tables 5-8. Every experiment uses fully connected layers with a ReLU activation function. The COMA and MACC method will wait 400 iterations before training the actor policies of the different agents to prevent the training of these policies with an untrained critic. The MADDPG method will wait for 30 iterations.

Table 5: Hyperparamaters for the Speaker Listener scenario

	MADDPG	COMA	MACC without SL	MACC
Critic learning rate	0.01	0.0025	0.01	0.01
Critic gamma	0.95	0.9	0.9	0.9
Critic regularizer	N/A	0.0001	0.0001	0.0001
Critic tau	0.01	0.0001	0.0001	0.0001
Critic hidden layers	[64, 64]	[512, 1024, 512]	[512, 1024, 512]	[512, 1024, 512]
Critic learning rate	0.01	4e-05	0.001	0.001
Action entropy beta	N/A	0.001	0.001	0.001
Action hidden layers	[64, 64]	[256, 256]	[256, 256]	[256, 256]
Action social loss lambda	N/A	N/A	0	0.005
Communication learning rate	N/A	N/A	0.005	0.005
Communication entropy beta	N/A	N/A	0.002	0.002
Communication hidden layers	N/A	N/A	[]	[]
Train batch size	900	900	900	900

Table 6: Hyperparamaters for the Simple Reference scenario

	MADDPG	COMA	MACC without SL	MACC
Critic learning rate	0.01	0.0025	0.005	0.005
Critic gamma	0.95	0.9	0.9	0.9
Critic regularizer	N/A	0.0001	0.0001	0.0001
Critic tau	0.01	0.0001	0.0001	0.0001
Critic hidden layers	[64, 64]	[512, 1024, 512]	[512, 1024, 512]	[512, 1024, 512]
Action learning rate	0.01	4e-05	4e-05	4e-05
Action entropy beta	N/A	0.001	0.001	0.001
Action hidden layers	[64, 64]	[256, 256]	[256, 256]	[256, 256]
Action social loss lambda	N/A	N/A	0	0.005
Communication learning rate	N/A	N/A	0.005	0.005
Communication entropy beta	N/A	N/A	0.001	0.001
Communication hidden layers	N/A	N/A	[]	[]
Train batch size	900	900	900	900

Table 7: Hyperparamaters for the Simple Crypto scenario

	MADDPG	COMA	MACC without SL	MACC
Critic learning rate	0.01	0.0025	0.01	0.01
Critic gamma	0.95	0.9	0.9	0.9
Critic regularizer	N/A	0.0001	0.0001	0.0001
Critic tau	0.01	0.0001	0.0001	0.0001
Critic hidden layers	[64, 64]	[512, 1024, 512]	[512, 1024, 512]	[512, 1024, 512]
Action learning rate	0.01	4e-05	0.01	0.01
Action entropy beta	N/A	0.001	0.001	0.001
Action hidden layers	[64, 64]	[32, 32, 32]	[32, 32, 32]	[32, 32, 32]
Action social loss lambda	N/A	N/A	0	0.005
Communication learning rate	N/A	N/A	0.001	0.001
Communication entropy beta	N/A	N/A	0.001	0.001
Communication hidden layers	N/A	N/A	[32, 32, 32]	[32, 32, 32]
Train batch size	900	900	900	900

Table 8: Hyperparamaters for the Speaker Listener with Communication Hub scenario

	MADDPG	COMA	MACC without SL	MACC
Critic learning rate	0.01	0.0025	0.0025	0.0025
Critic gamma	0.95	0.9	0.9	0.9
Critic regularizer	N/A	0.0001	0.0001	0.0001
Critic tau	0.01	0.0001	0.0001	0.0001
Critic hidden layers	[64, 64]	[512, 1024, 512]	[512, 1024, 512]	[512, 1024, 512]
Action learning rate	0.01	4e-05	4e-05	4e-05
Action entropy beta	N/A	0.001	0.001	0.001
Action hidden layers	[64, 64]	[256, 256]	[256, 256]	[256, 256]
Action social loss lambda	N/A	N/A	0	0.002
Communication learning rate	N/A	N/A	0.005	0.005
Communication entropy beta	N/A	N/A	0.001	0.001
Communication hidden layers	N/A	N/A	[]	[]
Communication gamma (γ_c)	N/A	N/A	0.9	0.9
Train batch size	900	900	900	900