

Disentangling Successor Features for Coordination in Multi-agent Reinforcement Learning

Seung Hyun Kim

University of Illinois at Urbana-Champaign
United States
skim449@illinois.edu

Girish Chowdhary

University of Illinois at Urbana-Champaign
United States
girishc@illinois.edu

Neale Van Stralen

University of Illinois at Urbana-Champaign
United States
nealeav2@illinois.edu

Huy T. Tran

University of Illinois at Urbana-Champaign
United States
huytran1@illinois.edu

ABSTRACT

Multi-agent reinforcement learning (MARL) is a promising framework for solving complex tasks with many agents. However, a key challenge in MARL is defining private utility functions that ensure coordination when training decentralized agents. This challenge is especially prevalent in unstructured tasks with sparse rewards and many agents. We show that successor features can help address this challenge by disentangling an individual agent’s impact on the global value function from that of all other agents. We use this disentanglement to compactly represent private utilities that support stable training of decentralized agents in unstructured tasks. We implement our approach using a centralized training, decentralized execution architecture and test it in a variety of multi-agent environments. Our results show improved performance and training time relative to existing methods and suggest that disentanglement of successor features offers a promising approach to coordination in MARL.

KEYWORDS

Reinforcement Learning; Multi-Agent Reinforcement Learning; Coordination

ACM Reference Format:

Seung Hyun Kim, Neale Van Stralen, Girish Chowdhary, and Huy T. Tran. 2022. Disentangling Successor Features for Coordination in Multi-agent Reinforcement Learning. In *Proc. of the 21st International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2022), Online, May 9–13, 2022*, IFAAMAS, 10 pages.

1 INTRODUCTION

Multi-agent reinforcement learning (MARL) has shown promising results for applications including traffic control [20], smart grids [14, 31], autonomous driving [27], and UAV control [44]. Many existing approaches implement centralized architectures to ensure coordination among agents [18, 20, 36]. However, fully centralized approaches struggle when faced with exponentially large state-action spaces and communication constraints. Decentralized execution helps address these issues by allowing agents to make independent decisions without communication. An open question in MARL is

then how to ensure coordination among such decentralized agents, particularly in unstructured tasks with sparse rewards and many agents [1, 23].

Previous work has addressed this problem by defining private utility functions, often referred to as individual value functions, for optimizing individual agent policies. However, current methods struggle to ensure scalable and stable learning in complex environments like the Starcraft Multi-Agent Challenge (SMAC) [32]. Two concepts that underlie these struggles are factoredness and learnability. Factoredness ensures a private utility is aligned with the global utility (e.g., the global value function), while learnability stabilizes training by disentangling an individual agent’s impact on the global utility from that of all other agents [42]. Recent frameworks, such as QMIX [30], QTRAN [37], WQMIX [29], and LICA [46], propose private utilities that achieve high factoredness and show promising results. However, these methods do not explicitly consider learnability, limiting their performance in highly complex environments with many agents. The counterfactual multi-agent (COMA) method implements a technique for computing private utilities with high factoredness and learnability [12], but requires a complex central critic to learn a joint action value function, limiting its performance in large multi-agent systems (MAS).

In this work, we ask whether *successor features (SFs) can be used to disentangle an individual agent’s impact on the global value function from that of all other agents, and if this disentanglement then enables better training of decentralized agents*. More specifically, we expect such disentanglement to improve decentralized training by enabling compact representation of a private utility that has high factoredness and high learnability. We introduce three key ideas to address this question, all of which leverage the ability of SFs to separate environment dynamics from rewards. First, we introduce a private utility, based on a modified formulation of the estimated difference utility (EDU) [42], that uses SFs to disentangle the impact of an agent from the global value function. We use our private utility, named SF-EDU, as an individual value function for training decentralized agents. Second, we introduce a process for learning the SF disentanglement vector used to calculate our SF-EDU. We optimize this disentanglement for learnability, which we estimate using SFs, to improve training stability and performance. Third, we introduce a centralized training, decentralized execution (CTDE) architecture with a shared feature encoding that efficiently implements these ideas.

Proc. of the 21st International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2022), P. Faliszewski, V. Mascardi, C. Pelachaud, M.E. Taylor (eds.), May 9–13, 2022, Online. © 2022 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). All rights reserved.

We test our approach in a variety of multi-agent environments. Our results suggest that disentanglement of SFs is a promising approach for improving performance in MARL relative to baseline methods, particularly in unstructured tasks with many agents, such as SMAC.

2 BACKGROUND

We model our problem as a decentralized partially observable Markov decision process (dec-POMDP), which is defined by the tuple $(\mathcal{I}, \mathcal{S}, \mathcal{O}^i, \mathcal{A}^i, \mathcal{P}, R)$. Here, \mathcal{I} is a finite set of agents, \mathcal{S} is the state space of the environment, \mathcal{O}^i is the observation space of agent $i \in \mathcal{I}$, and \mathcal{A}^i is the action space of agent i . At every time step, agents execute a joint action $\mathbf{a} = (a^1, \dots, a^{|\mathcal{I}|})$ where \mathbf{a} is in the joint action space $\mathcal{A} = \mathcal{A}^1 \times \dots \times \mathcal{A}^{|\mathcal{I}|}$, the environment updates to a new state s based on the transition function $\mathcal{P}(s'|s, \mathbf{a}) : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0, 1]$, and a global reward r is given to the team based on the reward function $R(s, \mathbf{a}) : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$.

We assume each agent has access to its own observation history $\tau_t^i = [o_t^1, \dots, o_t^i]$, where $o^i \in \mathcal{O}^i$ and $\tau_t^i \in T^i$. Our goal is to optimize a stochastic policy for each agent, $\pi^i(a^i | \tau_t^i) : T^i \times \mathcal{A}^i \rightarrow [0, 1]$, such that the global state value $V_G^\pi(s) = \mathbb{E}_\pi(\sum_{l=0}^{\infty} \gamma^l r_{t+l} | s_t = s)$ or the global action value $Q_G^\pi(s, \mathbf{a}) = \mathbb{E}_\pi(\sum_{l=0}^{\infty} \gamma^l r_{t+l} | s_t = s, \mathbf{a}_t = \mathbf{a})$ of the environment is maximized, where $\gamma \in [0, 1)$ is a discount factor and π is a joint policy.

2.1 Factoredness and Learnability

The factoredness, F_{g_i} , of a private utility g_i is defined as,

$$F_{g_i} = \frac{\sum_{\mathbf{z}} \sum_{\mathbf{z}'} u[(g_i(\mathbf{z}) - g_i(\mathbf{z}'))(G(\mathbf{z}) - G(\mathbf{z}'))]}{\sum_{\mathbf{z}} \sum_{\mathbf{z}'} 1}, \quad (1)$$

where \mathbf{z} is a joint move (e.g., a joint state action (s, \mathbf{a})), $u[\cdot]$ is the unit step function which has a value of one when the input argument is positive and zero otherwise, and G is the global utility function. High factoredness implies that positive changes in private utility result in positive changes in global utility, thus ensuring globally beneficial updates in decentralized training.

The learnability, $\lambda_{i, g_i}(\mathbf{z})$, of g_i is defined as,

$$\lambda_{i, g_i}(\mathbf{z}) = \frac{\mathbb{E}_{\mathbf{z}'_i} [|g_i(\mathbf{z}) - g_i(\mathbf{z}_{-i} + \mathbf{z}'_i)|]}{\mathbb{E}_{\mathbf{z}'_{-i}} [|g_i(\mathbf{z}) - g_i(\mathbf{z}'_{-i} + \mathbf{z}_i)|]}, \quad (2)$$

where \mathbf{z}_i are the components of \mathbf{z} that only depend on agent i and \mathbf{z}_{-i} are the components of \mathbf{z} that depend on all agents other than agent i . Learnability measures how sensitive an agent's private utility is to its own actions rather than the actions of other agents; high learnability thus reduces noise from other agents when updating an agent's private utility.

The EDU private utility assures full factoredness and promotes learnability by minimizing the influence of other agents on an individual agent's private utility [42]. For a private utility $g_i(\mathbf{z})$, the EDU is defined as,

$$EDU_i \equiv G(\mathbf{z}) - \mathbb{E}_{\mathbf{z}'_i} [G(\mathbf{z}) | \mathbf{z}_{-i}], \quad (3)$$

where $\mathbb{E}_{\mathbf{z}'_i} [G(\mathbf{z}) | \mathbf{z}_{-i}]$ disentangles (or marginalizes) the impact of all possible joint-actions for agents other than agent i from the global utility. However, implementing EDU is challenging when there is no explicit method for calculating this marginalization term. Function

approximation can help with this challenge [5, 12], but is still limited in environments with many agents or complex interactions among agents. That is, this term is difficult to approximate if the value function is highly elastic in response to the joint actions of other agents. We leverage SFs to overcome this challenge.

For the remainder of this paper, we define the global utility $G(\mathbf{z})$ as the global state value function $V_G^\pi(s)$ and a private utility $g_i(\mathbf{z})$ as an individual state value function $V_i^{\pi^i}(\tau_t^i)$. Note that we use action value functions instead of state value functions when calculating expectations of utility over actions. While we refer to $V_i^{\pi^i}$ and $Q_i^{\pi^i}$ as individual value functions for agent i , we note that they are more precisely utility functions since they do not strictly estimate the expected discounted sum of future rewards [30].

2.2 Successor Features

Successor representation (SR) was introduced in [7] as an approach for separating environment dynamics from rewards in MDPs, which can be used, for example, to enable fast policy adaptation [7, 13, 24] and task decomposition [21]. The SR defines the expected discounted future occupancy of state s' , given starting state s , action a , and policy π , as,

$$M^\pi(s, a, s') = \mathbb{E}_\pi \left[\sum_{l=0}^{\infty} \gamma^l \mathbb{1}[s_{t+l+1} = s'] | s_t = s, a_t = a \right], \quad (4)$$

where $\mathbb{1}[\cdot]$ is one when the input argument is true. The SR effectively measures the interconnectedness of different states in the environment, i.e., the dynamics of the environment. Given the SR, the action value function can be approximated as,

$$Q^\pi(s, a) = \sum_{s'} M(s, a, s') R(s'). \quad (5)$$

SR thus distinctly separates the environment dynamics, captured by M , from the environment reward structure, captured by R .

Successor features (SFs) generalizes SR to allow for the use of function approximation and applications in continuous state spaces [3, 22, 45]. The basis of SFs is that the expected one-step reward can be decomposed into a set of features, $\phi(s)$, and a linear reward weighting, \mathbf{w} , as follows,

$$R(s) = \phi(s)^T \cdot \mathbf{w}. \quad (6)$$

SFs then define the expected discounted future occupancy of features ϕ , given starting state s and policy π , as,

$$\psi^\pi(s) = \mathbb{E}_\pi \left[\sum_{l=0}^{\infty} \gamma^l \phi(s_{t+l+1}) | s_t = s \right], \quad (7)$$

where the i th element of ψ represents the expected discounting of the i th feature of ϕ . The state value function can then be calculated as the following,

$$V^\pi(s) = \psi^\pi(s)^T \cdot \mathbf{w}. \quad (8)$$

The original SR and SF formulations assume a fully observable MDP. We use SFs in our dec-POMDP setup by defining $\phi(s)$ and $\psi^\pi(s)$ of individual agents as $\phi(\tau_t^i)$ and $\psi^{\pi^i}(\tau_t^i)$, respectively. For the remainder of this paper, we omit the π notation in all global value functions and the π^i notation in all individual value functions and SF notations for simplicity.

3 RELATED WORK

The previous works most related to ours focus on decentralized coordination through careful construction of private utilities (often referred to as individual value functions). The value decomposition network (VDN) [39] considers the global value to be a linear summation of individual value functions. QMIX [30] extends VDN by modeling the global value as a non-linear function of individual value functions through the use of a mixing network, with a centralized update that guarantees full factoredness. WQMIX [28] further extends QMIX by modifying the update step to use a weighted update over joint actions, thereby reducing QMIX’s underestimates of the value approximation. LIIR [10] proposes an alternative agent objective functions with a learned intrinsic reward term. LICA [46] presents an actor-critic architecture that uses the mixed critic from QMIX and includes a modified policy-gradient exploration step to improve exploration near locally-optimal solutions.

These methods show promising results, but do not directly address the learnability of decentralized agents. VDN theoretically provides infinite learnability, but only under the assumption that the global value is truly a linear sum of individual value functions, which has been empirically shown to be a weak assumption in complex games. We aim to achieve high learnability, while maintaining high factoredness, in an effort to better handle noisy updates in decentralized settings for improved training speed, stability, and performance. COMA [12] aims to achieve high factoredness and learnability by using an EDU critic as an advantage for a policy gradient. QTRAN [37] also aims for high learnability by using a similar counterfactual joint action value function to learn agent-specific contributions to a task. However, COMA and QTRAN require a function approximation of the joint state action value. We address learnability in a manner that removes this requirement for better scaling in MAS with complex interactions among many agents.

Other works achieve coordination through reward redistribution amongst individual agents [8, 9, 40], but require domain-specific knowledge to define the reward shaping mechanisms. SFs have also been applied to MARL in previous works. [15] uses SFs in lieu of traditional value networks, allowing for efficient policy transfer to new reward structures. However, [15] assumes individual rewards are given to agents; we instead focus on environments where a global reward is given with no prior definition of individual rewards. [16] uses SFs within a VDN structure and uses its decomposition of dynamics and rewards for better value estimation under dynamic policies when learning. Our method instead proposes an entirely new line of work in which we leverage the linear feature composition of SFs to disentangle an individual agent’s value from all other agents in the global value function.

4 OUR APPROACH

We present the Disentangled SFs for Coordination (DISSC) framework, which leverages SFs to disentangle the impact of an individual agent on the global value function from the impact of all other agents. We use this disentanglement to improve coordination in MARL by compactly representing the EDU private utility and learnability. Our framework is composed of three key ideas: (1) an SF-EDU formulation that uses SFs to represent the marginalization

term in the EDU for unstructured tasks; (2) a process for learning an SF disentanglement vector that is optimized for learnability and used to calculate our SF-EDU; and (3) a CTDE architecture that efficiently implements these techniques. Our overall architecture is summarized in Figure 1.

4.1 SF-EDU Private Utility Formulation

A challenge with using the EDU private utility is efficiently calculating the marginalization term, i.e., $\mathbb{E}_{z_i} [G(\mathbf{z}) | z_{-i}]$. Rather than marginalizing $G(\mathbf{z})$ with a joint move, we instead present a modified EDU formulation that uses SFs to marginalize $G(\mathbf{z})$ with ψ . We define our SF-EDU private utility as,

$$EDU_i \approx V_G(s) - \mathbb{E}_{\pi_i} [Q_G(\tau_t^i, a^i) | \psi_{-i}] \quad (9)$$

$$\approx V_G(s) - V_{-i}(\tau_t^i) \quad (10)$$

$$\approx V_G(s) - \psi_{-i}^T \cdot \mathbf{w}, \quad (11)$$

where ψ_{-i} are the SFs associated with the contribution of all agents other than agent i to the global value. We define the second term in Equation (9) with respect to local agent information due to our dec-POMDP formulation. Similar to the original EDU formulation in Equation (3), the second term in Equation (11) represents the expected global value (or utility) associated with all agents other than i . We implement our formulation by calculating the global value V_G with a central critic and the marginalizing term using a decentralized critic. We then define $V_i = EDU_i$ to calculate a generalized advantage estimator $A_i = R_i(\tau_t^i) + [\gamma V_i(\tau_{t+1}^i) - V_i(\tau_t^i)]$ for training our decentralized actor policies. We discuss our implementation in more detail in Section 4.3.

The intuition behind SF-EDU is that certain features in an agent’s observation are likely not relevant to that agent’s contribution to the global value, and instead mostly contain noise associated with other agent contributions to the global value (see Figure 4 for a visual example of this idea). We represent those non-relevant features as ψ_{-i} . In principle, one could identify these non-relevant features using any latent features. However, our choice to use SFs allows us to directly relate the non-relevant features to a marginalized value function (i.e., V_{-i}), due to the unique separation property of SFs, where the value function can be decomposed into the product of a feature representation and a linear weighting, as shown in Equation (8).

4.2 SF Disentanglement Vector for Marginalizing Global Value and Reward

We estimate ψ_{-i} by learning an SF disentanglement vector, β , that filters an agent’s SF encoding to only focus on features relevant to that agent’s contribution to the global value and reward. More formally, we implement β as an element-wise rescaling vector that operates on an agent’s features ϕ and SFs ψ as follows,

$$\phi_{-i} = \phi \odot (1 - \beta), \quad (12)$$

$$\psi_{-i} = \psi \odot (1 - \beta), \quad (13)$$

where \odot denotes an element-wise product. We use the rescaled SFs ψ_{-i} to estimate V_{-i} for calculating our SF-EDU, as shown in eq. (11). Similarly, we leverage Equation (6) to use the rescaled features ϕ_{-i} to estimate the individual reward R_i associated with agent i (i.e.,

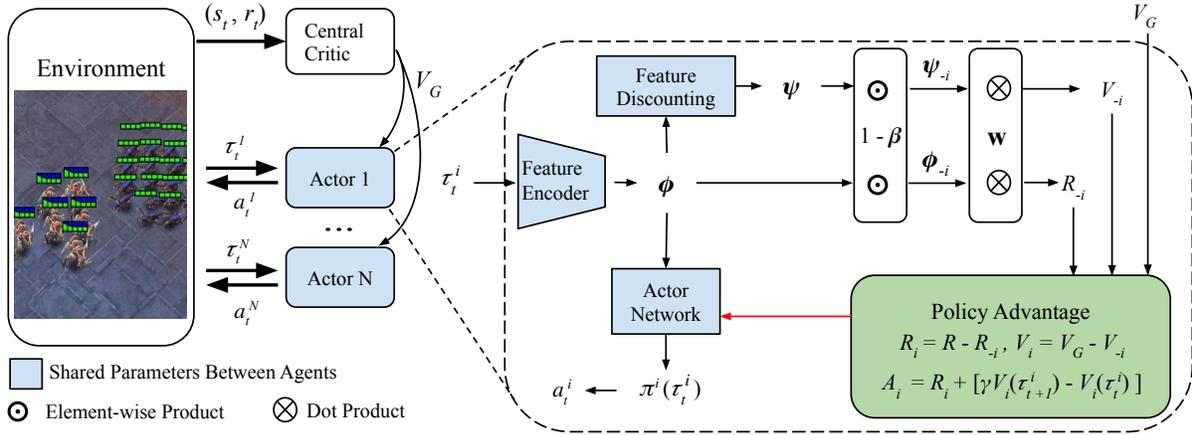


Figure 1: The architecture used to implement our DISSC framework. Our framework uses SFs to learn a disentanglement vector β that allows us to estimate individual agent value and reward functions, V_i and R_i , respectively. We use V_i and R_i to train decentralized agents, supported by a central critic used to estimate the global value function V_G . We optimize β for learnability, which we estimate using SFs, to improve training stability in complex environments with many agents.

the reward generated solely by agent i) as follows,

$$R_i = R - R_{-i} = R - (\phi_{-i}^T \cdot \mathbf{w}). \quad (14)$$

This quantity is used in the generalized advantage estimator for training individual agent policies. We focus on ϕ_{-i} and ψ_{-i} rather than their complements because we find them to improve stability during training, likely due to better initialization.

We learn β by optimizing it with respect to the learnability of SF-EDU. The intuition behind this idea is based on the fact that SF-EDU aims to approximate EDU, which achieves high learnability through its marginalization term. Since we estimate this marginalization term using ψ_{-i} , which is calculated from β , by optimizing β such that SF-EDU has high learnability, we ensure that β effectively filters an agent’s observation to focus on features relevant to its contribution to the global value function.

However, calculating learnability is challenging in complex environments. We therefore use SFs to derive a novel representation of learnability that can be estimated in such environments. Beginning with the original learnability equation and re-formulating it to use value functions as utilities, we represent the learnability $\lambda_i(\mathbf{a})$ of an agent’s individual value function as follows,

$$\lambda_i(\mathbf{a}) = \frac{\mathbb{E}_{\mathbf{a}_t'} \left[|Q_i(\tau_t^i, \mathbf{a}_t) - Q_i(\tau_t^i, \mathbf{a}_t^{-i} + \mathbf{a}_t')| \right]}{\mathbb{E}_{\mathbf{a}_t'} \left[|Q_i(\tau_t^i, \mathbf{a}_t) - Q_i(\tau_t^i, \mathbf{a}_t^{-i'} + \mathbf{a}_t')| \right]} \quad (15)$$

$$\approx \frac{\mathbb{E}_{\pi^i, \tau_t^i \rightarrow \tau_{t+1}^i} \left[|\psi(\tau_t^i) \mathbf{w} - \psi(\tau_t^i) \mathbf{w}^- - \psi(\tau_{t+1}^i) \mathbf{w}^+| \right]}{\mathbb{E}_{\pi^{-i}, \tau_t^i \rightarrow \tau_{t+1}^i} \left[|\psi(\tau_t^i) \mathbf{w} - \psi(\tau_{t+1}^i) \mathbf{w}^- - \psi(\tau_t^i) \mathbf{w}^+| \right]} \quad (16)$$

$$\approx \frac{\mathbb{E}_{\pi^i, \tau_t^i \rightarrow \tau_{t+1}^i} \left[|(\psi(\tau_t^i) - \psi(\tau_{t+1}^i)) \cdot \mathbf{w}^+| \right]}{\mathbb{E}_{\pi^{-i}, \tau_t^i \rightarrow \tau_{t+1}^i} \left[|(\psi(\tau_t^i) - \psi(\tau_{t+1}^i)) \cdot \mathbf{w}^-| \right]}, \quad (17)$$

where,

$$\begin{aligned} \mathbf{w}^+ &= \beta \odot \mathbf{w} \\ \mathbf{w}^- &= (1 - \beta) \odot \mathbf{w}. \end{aligned} \quad (18)$$

Similar to the original learnability, our representation measures the ratio of the expected value (or utility) change over variations of

agent i ’s actions relative to the expected value change over all other agent actions. We approximate the value over variations of agent i ’s actions $\left[Q_i(\tau_t^i, \mathbf{a}_t^{-i} + \mathbf{a}_t') \right]_{\mathbf{a}_t'}$ as the transition of value between two consecutive states $[\psi(\tau_t^i) \cdot \mathbf{w}^- + \psi(\tau_{t+1}^i) \cdot \mathbf{w}^+]_{\pi^i, \tau_t^i \rightarrow \tau_{t+1}^i}$; we use a similar approximation for the value over variations of all other agent actions. These approximations are based on the assumption of independent features in ψ and the fact that we define ψ_i and ψ_{-i} to be complementary vectors, as shown in Equation (18). We enforce complementarity by constraining $\beta \in [0, 1]$ in the update process, further described in Algorithm 1. We also calculate the expectation in the denominator of Equation (17) over the complete joint action rather than the joint action of agents other than i , resulting in Equation (17) being a conservative estimate of learnability, since our computation includes the influence of agent i ’s action.

Based on our derived representation for learnability, shown in Equation (17), we update β in a manner that maximizes learnability using the following loss,

$$L_{\lambda, \beta} = \sum_t \left[\sum_{\mathbf{a}_t^i} \pi_i(\mathbf{a}_t^i | \tau_t^i) \cdot \left\| (\psi(\tau_t^i) - \psi(\tau_{t+1}^i)) \cdot \mathbf{w}^+ \right\| \right] - c_\lambda \left\| (\psi(\tau_t^i) - \psi(\tau_{t+1}^i)) \cdot \mathbf{w}^- \right\|. \quad (19)$$

The numerator and denominator of Equation (17) are optimized separately for numerical reasons and c_λ is introduced to mitigate the impact of magnitude differences between those quantities during training.

4.3 CTDE Architecture

We implement our framework using a CTDE architecture, building from recent work suggesting that similar architectures encourage coordination given a global reward [6, 19]. While we include centralized training in this work, our framework can be applied to fully decentralized architectures with little modification.

Our central critic builds from the formulation used by COMA [12], in that it estimates the global value and sends this value to

Algorithm 1: DISSC Algorithm

```

Initialize  $\Theta, \theta^i, \pi^i, \psi^i$  Initialize  $\beta^i \leftarrow 1$ ;
Initialize trajectory buffer  $D_{\text{central}}, D_{\text{decentral}}^i$ ;
repeat
  Reset environment;
  while  $t < \text{max step}$ 
    Obtain observations  $s_t, \tau_t^1, \tau_t^2, \dots$  from environment;
    Obtain and execute actions  $\pi(\tau_t^1), \dots, \pi(\tau_t^I)$ ;
    Store  $D_{\text{central}} \leftarrow (s_t, r_t)$ ;
    Store  $D_{\text{decentral}}^i \leftarrow (\tau_t^i, r_t, a_t^i)^i$ ;
    if  $|D_{\text{central}}| < \text{central batch size}$  then
      Sample  $(r_t, V_G(s_t)) \forall s_t \in D_{\text{central}}$ ;
      Compute TD-target:
         $TD_{\text{central}} = r_{t+1} + \gamma V_G^{-\Theta}(s_{t+1})$ ;
      Update central critic:
         $\Theta \leftarrow \Theta - \alpha_{\text{central}} \nabla_{\Theta} (TD_{\text{central}} - V_G(s_t))^2$ ;
    end
    if  $|D_{\text{decentral}}^i| < \text{decentral batch size}$  then
      Sample  $\pi^i, r_t, \phi(\tau_t^i), \psi(\tau_t^i), V_i, \forall \tau_t^i \in D_{\text{decentral}}^i$ ;
      Calculate agent advantage defined in Section 4.1;
      Update agent policy:  $\theta_{\pi} \leftarrow \theta_{\pi} - c_{\pi} \nabla_{\theta_{\pi}} L_{\text{ppo}}$ ;
      Update  $\phi$  representation with Equations (20)
        and (21):  $\theta_{\phi} \leftarrow \theta_{\phi} - \alpha_{\phi} \nabla_{\theta_{\phi}} (L_{\text{reward}} + L_{\text{pred}})$ ;
      Update  $\psi$  representation with Equation (22):
         $\theta_{\psi} \leftarrow \theta_{\psi} - \alpha_{\psi} \nabla_{\theta_{\psi}} (L_{\text{SF}})$ ;
      Update learnability rescaling of agents:
         $\beta \leftarrow \beta - \alpha_{\lambda} \nabla_{\beta} L_{\lambda, \beta}$ ;
         $\beta = \text{clip}(\beta, 0.0, 1.0)$ ;
    end
  end
until Finished;

```

individual agents for their EDU calculation. However, we estimate the global state value V_G instead of the global action value Q_G to better handle large-scale MAS. Our decentralized controllers implement their own internal actor-critic architectures. We train the decentralized critics using SFs, where we use the learned ϕ and ψ to estimate the global reward R and state value V_G with Equations (6) and (8), respectively. We optimize individual agents' policies, $\pi^i(a^i|\tau_t^i)$, with the proximal policy optimization (PPO) algorithm [33], using our SF-EDU to calculate the advantage in lieu of the traditional advantage function (as discussed in Section 4.1).

Our architecture uses a shared SF encoder for all agents. This shared encoder is motivated by existing work demonstrating benefits of weight sharing [4, 11, 26] and based on a hypothesis that using SFs, we can model spatial and dynamic features in an environment in a manner that is common to all agents regardless of their type. This weight-sharing significantly reduces network complexity and paves the way towards efficiently utilizing experiences in heterogeneous MARL training.

4.4 Successor Features Implementation

We train the components of our SF critic, the feature encoder, and feature discounting using standard SF losses [17, 45]. We first train

the feature encoder to recover a set of features, ϕ , that encodes information from local surroundings using the following reward estimation loss,

$$L_{\text{reward}, \theta_{\phi}} = \left\| r_t - \phi^{\theta_{\phi}}(\tau_t^i)^T \cdot \mathbf{w} \right\|^2. \quad (20)$$

Using this loss alone, however, often does not generate a dense enough feature set for stable training, particularly in sparse reward environments. Motivated by [2], we address this issue by implementing auxiliary training networks to predict the next state and increase the amount of encoded information seen by ϕ , using the following prediction loss,

$$L_{\text{pred}, \theta_{\phi}} = \left\| o_{t+1}^i - \mathcal{D}(\phi^{\theta_{\phi}}(o_t^i), a_t^i) \right\|^2, \quad (21)$$

where $\mathcal{D}(\phi(s_t^i), a_t^i)$ is a decoder network that predicts the next state observation. The feature discounting network is trained to discount the learned features with the following SF loss,

$$L_{\text{SF}, \theta_{\psi}} = \left\| \phi(\tau_{t+1}^i) + \gamma \psi^{\theta_{\psi}}(\tau_{t+1}^i) - \psi(\tau_t^i) \right\|^2, \quad (22)$$

which is the TD-error of the SFs ψ generalized into a Bellman equation. Our overall algorithm is summarized in Algorithm 1.

5 RESULTS

We test our approach through a set of experiments using the open-source Multi-Agent Particle (MAP) environments [19], SMAC [32], and a capture-the-flag (CtF) environment [43]. We compare to the following published MARL baselines: COMA [12], QMIX [30], QTRAN [37], WQMIX [29], LIIR [10], and LICA [46]. We tuned baselines for environments (and individual maps in SMAC) that they were not originally demonstrated on by running several experiments over common hyperparameters, primarily learning rate and batch size, and method specific variables, such as α in WQMIX. We also extended the ϵ -greedy exploration to 1M episodes for Q-learning baselines to improve exploration in hard SMAC maps and CtF, as suggested in [29]. Note that our method and all baselines implemented use a CTDE architecture. All figures plot the mean performance with one standard deviation shaded.

Similar to existing MARL architectures [10, 29, 30, 46], DISSC used an MLP feature encoder with an LSTM in the Predator Prey and SMAC environments. We used MLPs for all other networks. For the CTF environment, which uses an image-like input, convolutional layers were added to the head of the feature encoder to capture spatial relationships. All networks were optimized with an Adam optimizer with a learning rate of 1E-4 based on the losses described in Equations (20) to (22). We set $c_{\lambda} = 0.5$ for the results shown; our testing found our results to have little sensitivity to the value of c_{λ} . Code is available at <https://github.com/Tran-Research-Group/DISSC>.

5.1 Multi-Agent Particle Environments

We first consider the predator-prey environment from the MAP test suite [19]. The goal of predator-prey is for three predators to capture a randomly moving agent as fast as possible. The environment requires coordination to efficiently and quickly solve the task.

Figure 3 shows results for this environment. We observe that several methods, including DISSC, are able to sufficiently solve

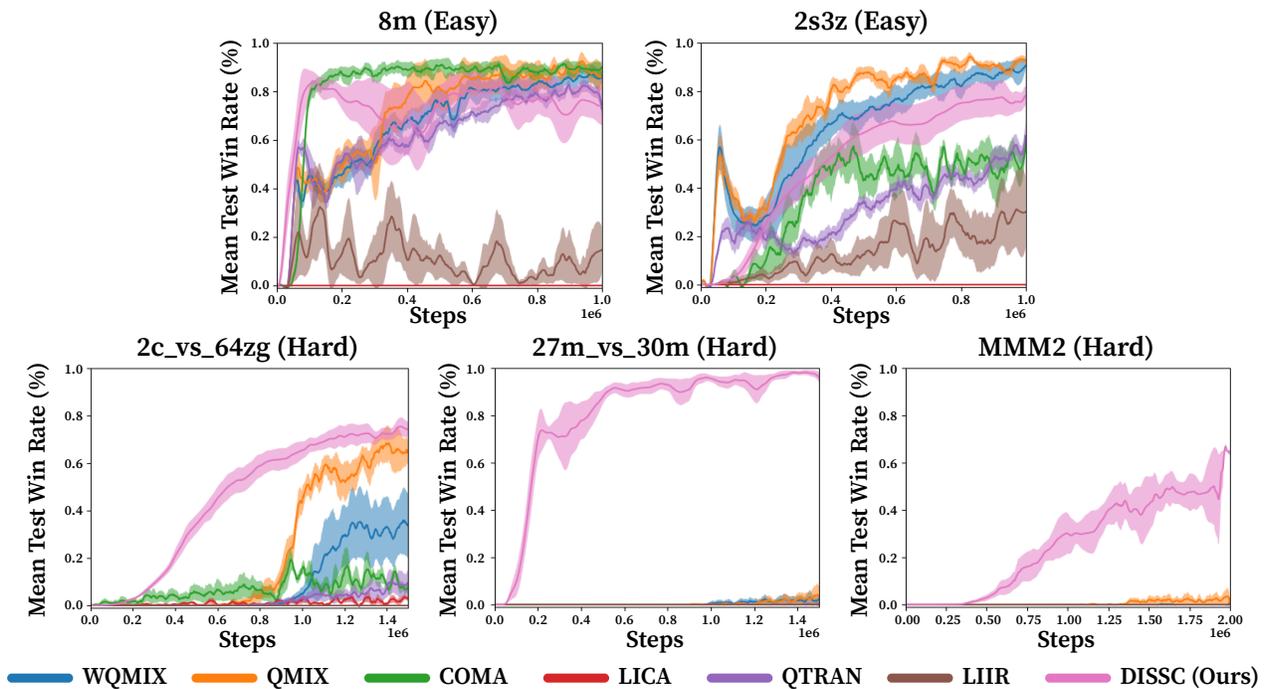


Figure 2: Training convergence plots in five SMAC maps, calculated over five replicates. A higher win-rate indicates better performance. In easier games, 8m and 2s3z, we match performance of the baselines. In harder games (which control more agents), our method has significantly faster learning times and higher performance.

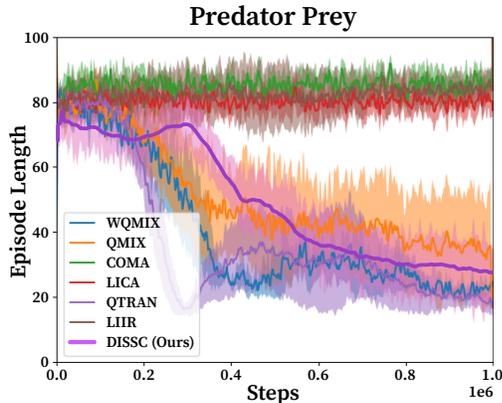


Figure 3: Training convergence plots in the predator-prey environment, over 10 replicates. Most methods solve the task with some variation in learning speed. Lower episode length indicates better performance.

the task by minimizing the time taken to capture the prey. COMA, LICA, and LIIR struggled with this environment. While DISSC does converge to optimal performance, we note that it is slower than some alternatives. We believe that this is due to our added complexity of learning SFs to model the value function; for this simple environment, that complexity does not outweigh the benefit of our disentanglement approach.

5.2 Starcraft Multi-Agent Challenge

We also consider the SMAC environment, a Starcraft II mini-game that focuses on micromanaging a set of agents in combat. We consider several maps ranging in difficulty and heterogeneity of the controlled agents. Easy maps have smaller numbers of controlled units and fight an equal number of enemies (8 marines vs. 8 marines in 8m, 2 stalkers and 3 zealots vs 2 stalkers and 3 zealots in 2s3z). Hard maps either place the agents at a numbers disadvantage (2 colosi vs. 64 zerglings in 2c_vs_64zg), control a large number of units (27 marines in 27m_vs_30m), or control highly heterogenous units (1 Medivac, 2 Mauraders, & 7 Marines in MMM2).

Figure 2 shows our results for various SMAC maps. For easy maps, we see that nearly all methods are able to attain high levels of performance, including DISSC. Our method shows fast learning in 8m but slower initial learning in 2s3z, again suggesting that the benefits of learning SFs towards learning speed may be reduced for simple environments.

In harder maps, we begin to observe the performance benefit of our method. For the 2c_vs_64zg map, only DISSC, QMIX, and WQMIX are able to obtain a non-trivial positive win-rate; all other methods essentially fail to solve the task. However, amongst these three, our method shows significantly faster learning, the highest converged performance, and the lowest variance in win rate. As we move to maps where more agents are controlled (i.e., 2c_vs_64zg, MMM2, 27m_vs_30m), we see that our method is the only one able to solve the task. We expect that these performance gains are due to the learning stability provided by our method through its direct optimization of learnability, which is most beneficial in complex

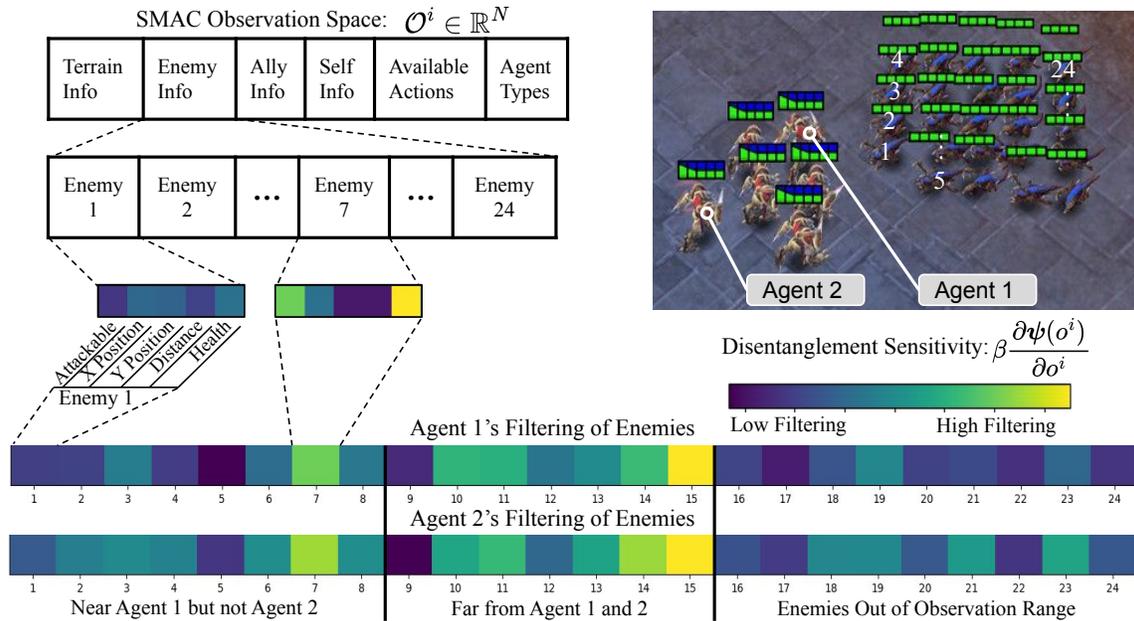


Figure 4: Effect of the disentanglement vector β on agent learning. Agent observations in SMAC are a vector with information about current position, enemies, allies, and available actions. We use Grad-CAM [35] to correlate the sensitivity of the feature space, ψ , with respect to the observation, o^i . The gradient, $\partial\psi/\partial o^i$, can then be multiplied by the disentanglement vector, β , to understand which observation variables are the most heavily filtered during disentanglement. The breakout shows Agent 1 filters Enemy 1 less than Enemy 7, primarily by their “Attackable” and “Health” variables. This filtering occurs because Agent 1 is closest to Enemy 1 and can attack the enemy to generate reward by damaging them. The bottom breakout shows the summation of the filtering across all enemies for Agents 1 and 2 and supports the trend that enemies further away from agents are more heavily filtered.

environments with many agents. There may also be some benefit incurred by our architecture, since we use a simplified central critic, $V_G = f(s)$, instead of, for example, complex mixing networks used by QMIX and WQMIX which scale linearly with the number of agents, $Q_G = f(s, V_1, \dots, V_I)$. Our architecture thus better scales to large MAS like 27m_vs_30m and MMM2.

To better understand our method, we also visualize the effect of our disentanglement vector β on individual agents, shown in Figure 4. We accomplish this by using a Grad-CAM-like [35] approach to correlate the effect of observations to the latent space ψ . More specifically, we calculate a correlation, $\partial\psi/\partial o^i$, that captures the effect of changing the observation on the feature space. We then multiply this correlation by β to evaluate which elements of the observation space are being filtered by β .

We observe two interesting trends from this analysis. First, we see that β heavily filters specific elements of the observation space, primarily the “Enemy Health” and “Enemy is Attackable” variables for nearly every enemy. This result is encouraging, since rewards in SMAC are given for damaging enemy units, suggesting that β learns to assign value based on whether or not an enemy is within attack range and its health. Second, we see that agents filter different features depending on their position with respect to enemies, as agents far away from enemies filter enemy variables more heavily than agents closer to enemies.

5.3 Capture the Flag (CtF)

We further test our approach in a CtF environment to understand its performance in a sparse reward setting. Here, two teams of agents compete against one another to capture their opponent’s flag while simultaneously defending their own flag. Agent interactions focus on flag capture and stochastic engagements in which agents can be temporarily removed from gameplay; these engagements depend on various factors such as agent strength, proximity to other agents, and home field advantage. Agents are randomly re-spawned within their own territory after being removed from gameplay. Flags are randomly re-spawned within their team’s territory immediately after being captured. We use an open-source implementation of the game [43] that operates in a 2D grid-world.

We compare to WQMIX and QMIX (since they show the most promise among baselines in SMAC hard environments) in this experiment, along with a vanilla independent actor-critic (IAC) baseline. We experiment in 5-vs-5 and 7-vs-7 games, where teams are composed of two types of agents: slow, strong “convoy” agents and fast, weak “normal” agents. Figure 5 shows that our method, DISSC, significantly outperforms QMIX and WQMIX, with IAC falling somewhere in the middle. We expect that the poor performance of QMIX and WQMIX is due to convergence to a defensive policy where the agents remain on their own territory to limit their chances of being killed. This sub-optimal policy likely occurs due to limited exploration in this sparse reward environment, a known issue with many Q -learning methods.

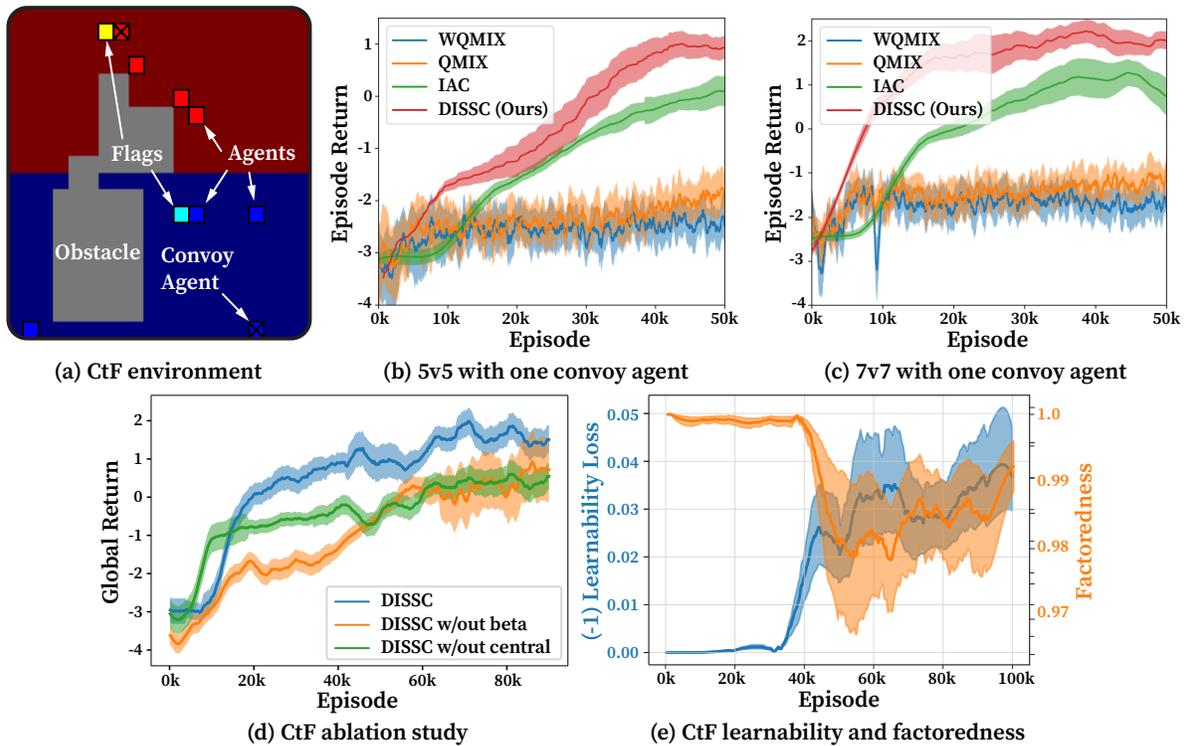


Figure 5: (a) The CtF environment. (b)-(c) Training convergence plots for our considered methods in two CtF settings, calculated over five replicates. Our method shows improved training time, converged performance, and stability over baselines. We also show (d) ablation results and (e) learnability and factoredness improvements during training.

Figure 5d also contains results from an ablation study of DISSC, comparing the effects of our β disentanglement vector and our centralized critic on model performance. We see that removing β reduces training stability, supporting our assertion that our modified EDU improves training stability through its high learnability. We also see that removing the central critic, and instead allowing individual agents to evaluate the global value, results in the lowest converged global return within our model variants, suggesting that local SF critics struggle to estimate the global value. Local critics likely struggle because agents only have access to their own individual observations during training.

We also investigate how factoredness and learnability change during training to better understand our SF-EDU learning process, as shown in Figure 5e. We calculated factoredness using Equation (1) and learnability using Equation (17). We see that during training, DISSC is able to explicitly increase learnability of individual agents, though there appears to be a slight tradeoff between factoredness and learnability. This tradeoff is likely due to our approximation of the EDU, but is minimal relative to changes in learnability. These results suggest that learnability and factoredness may naturally conflict in some environments and therefore require simultaneous optimization of both in such settings.

6 CONCLUSIONS AND FUTURE WORK

A critical challenge in MARL is ensuring coordination among decentralized agents in unstructured and complex tasks. We show that SFs can be used to disentangle the impact of individual agents

on the global value function, from the impact of all other agents. This disentanglement allows us to compactly represent a individual value function that has high factoredness and high learnability such that we can train decentralized agents in a coordinated and stable manner. We implement our approach in a CTDE architecture and demonstrate that it shows improved training time and performance over alternatives in several multi-agent environments.

For future work, we suggest exploring learning SF disentanglements that account for specific agent types. While our results suggest such heterogeneity is not required, further developments may be able to leverage agent-specific disentanglements to identify potential roles or specializations for a given task. This notion is also inspired by recent biological discoveries in mammals that have shown cell specialization to encode environment information in a semi-structured format, instead of an unknown latent space [25, 34, 38, 41].

ACKNOWLEDGEMENTS

This work was supported in part by ONR N00014-20-1-2249 and ARL W911NF2020184.

REFERENCES

- [1] Christopher Amato, Girish Chowdhary, Alborz Geramifard, and N Kemal. 2013. Decentralized Control of Partially Observable Markov Decision Processes. In *52nd IEEE Conference on Decision and Control*. IEEE, 2398–2405.
- [2] Andre Barreto, Diana Borsa, John Quan, Tom Schaul, David Silver, Matteo Hessel, Daniel Mankowitz, Augustin Zidek, and Remi Munos. 2018. Transfer in deep

- reinforcement learning using successor features and generalised policy improvement. *35th International Conference on Machine Learning, ICML 2018 2* (2018), 844–853. arXiv:1901.10964
- [3] André Barreto, Will Dabney, Rémi Munos, Jonathan J. Hunt, Tom Schaul, Hado Van Hasselt, and David Silver. 2017. Successor features for transfer in reinforcement learning. *Advances in Neural Information Processing Systems 2017-December, NIPS (2017)*, 4056–4066. arXiv:1606.05312
- [4] Elildo A.R. Carvalho and Juarez C.B. Pezuti. 2010. Hunting of jaguars and pumas in the Tapajós-Arapuins Extractive Reserve, Brazilian Amazonia. *ORYX* 44, 4 (2010), 610–612. <https://doi.org/10.1017/S003060531000075X> arXiv:1605.06676
- [5] Mitchell Colby, William Curran, Carrie Rebhuhn, and Kagan Turner. 2014. Approximating difference evaluations with local knowledge. *13th International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2014 2*, 2 (2014), 1577–1578.
- [6] Kevin Corder, Manuel M. Vindiola, and Keith Decker. 2019. Decentralized Multi-Agent Actor-Critic with Generative Inference. (2019). arXiv:1910.03058 <http://arxiv.org/abs/1910.03058>
- [7] Peter Dayan. 1993. Improving Generalization for Temporal Difference Learning: The Successor Representation. *Neural Computation* 5, 4 (1993), 613–624. <https://doi.org/10.1162/neco.1993.5.4.613>
- [8] Sam Devlin and Daniel Kudenko. 2011. Theoretical considerations of potential-based reward shaping for multi-agent systems. In *The 10th International Conference on Autonomous Agents and Multiagent Systems*. ACM, 225–232.
- [9] Sam Devlin, Logan Yliniemi, Daniel Kudenko, and Kagan Tumer. 2014. Potential-based difference rewards for multiagent reinforcement learning. In *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*. 165–172.
- [10] Yali Du, Lei Han, Meng Fang, Tianhong Dai, Ji Liu, and Dacheng Tao. 2019. LIIR: Learning individual intrinsic reward in multi-agent reinforcement learning. In *Advances in Neural Information Processing Systems*, Vol. 32.
- [11] Partha S. Dutta, Nicholas R. Jennings, and Luc Moreau. 2005. Cooperative information sharing to improve distributed learning in multi-agent systems. *Journal of Artificial Intelligence Research* 24 (2005), 407–463. <https://doi.org/10.1613/jair.1735>
- [12] Jakob N Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. 2018. Counterfactual multi-agent policy gradients. *Thirty-Second AAAI Conference on Artificial Intelligence* (2018).
- [13] Samuel J. Gershman, Christopher D. Moore, Michael T. Todd, Kenneth A. Norman, and Per B. Sederberg. 2012. The successor representation and temporal context. *Neural Computation* 24, 6 (2012), 1553–1568. https://doi.org/10.1162/NECO_a_00282
- [14] Arman Ghasemi, Amin Shojaei, Hadikolaei, Kailani Jones, Morteza Hashemi, Alexandru G. Bardas, and Reza Ahmadi. 2020. A multi-agent deep reinforcement learning approach for a distributed energy marketplace in smart grids. *2020 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids, SmartGridComm 2020* (2020). <https://doi.org/10.1109/SmartGridComm47815.2020.9302981> arXiv:2009.10905
- [15] Tarun Gupta, Akshat Kumar, and Praveen Paruchuri. 2019. Successor Features Based Multi-Agent RL for Event-Based Decentralized MDPs. *Proceedings of the AAAI Conference on Artificial Intelligence* 33 (2019), 6054–6061. <https://doi.org/10.1609/aaai.v33i01.33016054>
- [16] Tarun Gupta, Anuj Mahajan, Bei Peng, Wendelin Böhmer, and Shimon Whiteson. 2021. Uneven: Universal value exploration for multi-agent reinforcement learning. In *International Conference on Machine Learning*. PMLR, 3930–3941.
- [17] Tejas D. Kulkarni, Ardavan Saeedi, Simanta Gautam, and Samuel J. Gershman. 2016. Deep Successor Reinforcement Learning. (2016). arXiv:1606.02396 <http://arxiv.org/abs/1606.02396>
- [18] Alex Tong Lin, Mark J. Debor, Katia Estabridis, Gary Hewer, and Stanley Osher. 2019. CESMA: Centralized Expert Supervises Multi-Agents. (2019). arXiv:1902.02311 <http://arxiv.org/abs/1902.02311>
- [19] Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. 2017. Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments. *NIPS (2017)*. <https://doi.org/10.1007/BF01744832> arXiv:1706.02275
- [20] Jinming Ma and Feng Wu. 2020. Feudal multi-agent deep reinforcement learning for traffic signal control. In *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS*, Vol. 2020-May. 816–824. www.ifaamas.org
- [21] Marlos C. Machado, Marc G. Bellemare, and Michael Bowling. 2018. Count-Based Exploration with the Successor Representation. (2018). arXiv:1807.11622 <http://arxiv.org/abs/1807.11622>
- [22] Saphora Madjihurem and Laura Toni. 2019. State2vec: Off-Policy Successor Features Approximators. (2019), 1–10. arXiv:1910.10277 <http://arxiv.org/abs/1910.10277>
- [23] Laetitia Matignon, Guillaume J Laurent, and Nadine Le Fort-Piat. 2012. Independent reinforcement learners in cooperative Markov games: a survey regarding coordination problems. *Knowledge Engineering Review* 27, 1 (2012), 1–31.
- [24] I. Momennejad, E. M. Russek, J. H. Cheong, M. M. Botvinick, N. D. Daw, and S. J. Gershman. 2017. The successor representation in human reinforcement learning. *Nature Human Behaviour* 1, 9 (sep 2017), 680–692. <https://doi.org/10.1038/s41562-017-0180-8>
- [25] Maria E. Montchal, Zachariah M. Reagh, and Michael A. Yassa. 2019. Precise temporal memories are supported by the lateral entorhinal cortex in humans. *Nature Neuroscience* 22, 2 (feb 2019), 284–288. <https://doi.org/10.1038/s41593-018-0303-1>
- [26] Lucas Oliveira Souza, Gabriel De Oliveira Ramos, and Celia Ghedini Ralha. 2019. Experience sharing between cooperative reinforcement learning agents. *Proceedings - International Conference on Tools with Artificial Intelligence, ICTAI 2019-Novem* (2019), 963–970. <https://doi.org/10.1109/ICTAI.2019.00136> arXiv:1911.02191
- [27] Praveen Palanisamy. 2019. Multi-Agent Connected Autonomous Driving using Deep Reinforcement Learning. *NeurIPS (2019)*. arXiv:1911.04175 <http://arxiv.org/abs/1911.04175>
- [28] Tabish Rashid, Gregory Farquhar, Bei Peng, and Shimon Whiteson. 2020. Weighted QMIX: Expanding Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning. In *Neurips*. arXiv:2006.10800v2
- [29] Tabish Rashid, Mikayel Samvelyan, Christian Schroeder De Witt, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. 2020. Monotonic value function factorisation for deep multi-agent reinforcement learning. *Journal of Machine Learning Research* 21 (2020). arXiv:2006.10800v2
- [30] Tabish Rashid, Mikayel Samvelyan, Christian Schroeder De Witt, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. 2018. QMIX: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning. Technical Report. arXiv:1803.11485v2
- [31] Martin Roesch, Christian Linder, Roland Zimmermann, Andreas Rudolf, Andrea Hohmann, and Gunther Reinhart. 2020. Smart grid for industry using multi-agent reinforcement learning. *Applied Sciences (Switzerland)* 10, 19 (2020), 1–20. <https://doi.org/10.3390/app10196900>
- [32] Mikayel Samvelyan, Tabish Rashid, Christian Schroeder De Witt, Gregory Farquhar, Nantas Nardelli, Tim G.J. Rudner, Chia Man Hung, Philip H.S. Torr, Jakob Foerster, and Shimon Whiteson. 2019. The StarCraft multi-agent challenge. In *Workshop on Deep Reinforcement Learning at the 33rd Conference on Neural Information Processing Systems (NeurIPS 2019)*. arXiv:1902.04043
- [33] John Schulman, Sergey Levine, Philipp Moritz, Michael I. Jordan, and Pieter Abbeel. 2015. Trust Region Policy Optimization. (2015). <https://doi.org/10.1063/1.4927398> arXiv:1502.05477
- [34] Heidrun Schultz, Tobias Sommer, and Jan Peters. 2015. The role of the human entorhinal cortex in a representational account of memory. <https://doi.org/10.3389/fnhum.2015.00628>
- [35] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. 2020. Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization. *International Journal of Computer Vision* 128, 2 (2020), 336–359. <https://doi.org/10.1007/s11263-019-01228-7> arXiv:1610.02391
- [36] Tianmin Shu and Yuandong Tian. 2018. M3RL: MIND-AWARE MULTI-AGENT MANAGEMENT REINFORCEMENT LEARNING. Technical Report. arXiv:1810.00147v3 <https://github.com/facebookresearch/M3RL>.
- [37] Kyunghwan Son, Daewoo Kim, Wan Ju Kang, David Hostallero, and Yung Yi. 2019. QTRAN: Learning to factorize with transformation for cooperative multi-agent reinforcement learning. In *36th International Conference on Machine Learning, ICML 2019*, Vol. 2019-June. 10329–10346. arXiv:1905.05408
- [38] Kimberly L. Stachenfeld, Matthew M. Botvinick, and Samuel J. Gershman. 2017. The hippocampus as a predictive map. *Nature Neuroscience* 20, 11 (2017), 1643–1653. <https://doi.org/10.1038/nn.4650>
- [39] Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Viniçius 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 International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS*, Vol. 3. 2085–2087. arXiv:arXiv:1706.05296v1
- [40] Zhenggang Tang, Chao Yu, Boyuan Chen, Huazhe Xu, Xiaolong Wang, Fei Fang, Simon Du, Yu Wang, and Yi Wu. 2021. Discovering Diverse Multi-Agent Strategic Behavior via Reward Randomization. *ICLR (2021)*.
- [41] Albert Tsao, May Britt Moser, and Edvard I. Moser. 2013. Traces of experience in the lateral entorhinal cortex. *Current Biology* 23, 5 (mar 2013), 399–405. <https://doi.org/10.1016/j.cub.2013.01.036>
- [42] Kagan Tumer. 2006. Designing agent utilities for coordinated, scalable and robust multi-agent systems. *Coordination of Large-Scale Multiagent Systems* (2006), 173–188. https://doi.org/10.1007/0-387-27972-5_8
- [43] Neale Van Stralen, Seung Kim, H Tran Tran, and Girish Chowdhary. 2020. Evaluating Adaptation Performance of Hierarchical Deep Reinforcement Learning. In *2020 International Conference on Robotics and Automation (ICRA)*.
- [44] Fangkai Yang, Daoming Lyu, Bo Liu, and Steven Gustafson. 2018. Peorl: Integrating symbolic planning and hierarchical reinforcement learning for robust decision-making. *IJCAI International Joint Conference on Artificial Intelligence 2018-July* (2018), 4860–4866. arXiv:1804.07779
- [45] Jingwei Zhang, Jost Tobias Springenberg, Joshka Boedecker, and Wolfram Burgard. 2017. Deep reinforcement learning with successor features for navigation

across similar environments. *IEEE International Conference on Intelligent Robots and Systems 2017-Septe (2017)*, 2371–2378. <https://doi.org/10.1109/IROS.2017.8206049> arXiv:1612.05533

[46] Meng Zhou, Ziyu Liu, Pengwei Sui, Yixuan Li, and Yuk Ying Chung. 2020. Learning Implicit Credit Assignment for Cooperative Multi-Agent Reinforcement Learning. In *Neurips*.