UTS UNIVERSITY OF TECHNOLOGY SYDNEY

Towards a Unified Multi-agent Reinforcement Learning Framework

by Siyi Hu

Thesis submitted in fulfilment of the requirements for the degree of

Doctor of Philosophy

under the supervision of Professor Xiaojun Chang

University of Technology Sydney
Faculty of Engineering and Information Technology

May 2024

CERTIFICATE OF ORIGINAL OWNERSHIP

I, *Siyi Hu*, declare that this thesis, submitted in fulfilment of the requirements for the award of *Doctor of Philosophy*, in the *School of Computer Science*, *Faculty of Engineering and Information Technology* at the University of Technology Sydney. This thesis is wholly my own work unless otherwise referenced or acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis. This document has not been submitted for qualifications at any other academic institution. This research is supported by the Australian Government Research Training Program.

SIGNATURE: Production Note:
Signature removed prior to publication.

DATE: 16th May, 2024

PLACE: Sydney, Australia

Abstract

The field of Multi-Agent Reinforcement Learning (MARL) has rapidly evolved, yet integrating diverse tasks and algorithms into a cohesive system remains a complex challenge. This thesis proposes a unified framework aimed at improving adaptability, scalability, and cooperative dynamics among agents across various tasks and environments. Our research is structured around three primary advancements: the development of a flexible architecture, the analytical quantification of agent roles, and the creation of an integrative library for MARL. Initially, we introduce a novel architectural model capable of accommodating varying task configurations through a transformer-based approach that separates policy decisions from input observations. This model enhances transfer capabilities and accelerates training processes, showcasing substantial improvements in diverse MARL applications. Following this, we delve into the concept of Role Diversity, which quantifies and utilizes behavioral differences among agents to optimize policy performance. This analysis highlights how understanding these diversities can influence and improve key MARL strategies such as parameter sharing, communication mechanisms, and credit assignment, thereby enhancing overall system efficiency and adaptability. Finally, we develop a comprehensive MARL library that standardizes environment and algorithm integration, facilitating the flexible mapping of policies and streamlined development of multi-agent systems. This tool effectively simplifies the complexity of deploying diverse learning algorithms and managing multiple tasks, promoting a more systematic approach to MARL. Together, these innovations contribute significantly to the MARL field, offering novel insights and methodologies that advance the unified and efficient implementation of MARL.

Dedication

This thesis is dedicated to my wife and my parents, whose unwavering support and encouragement have been my anchor and inspiration throughout this journey.

Acknowledgments

First and foremost, I extend my deepest gratitude to my supervisor, Prof. Xiaojun Chang, whose guidance was indispensable to the completion of this thesis. I am immensely thankful for his selfless support, patience, and the profound insights he provided throughout my research journey. His expertise in the related fields has been a cornerstone of my academic development.

I am also deeply grateful to Prof. Yaodong Yang, Prof. Xiaodan Liang, Prof. Zhihui Li, Prof. Hao Dong, and many other esteemed professors. Their invaluable advice and guidance have significantly shaped my academic experience during my PhD studies.

I owe a great debt of gratitude to my colleagues and coauthors, including Dr. Mingjie Li, Dr. Changlin Li, Dr. Chuanlong Xie, Dr. Fengda Zhu, Zizheng Pan, Kaijie Yang, Ceyao Zhang, Yifan Zhong, Jiarong Liu, Haodong Hong, Yi Zhang, and many others. Working alongside them and engaging in intellectual discussions has been both a privilege and a major source of inspiration throughout my PhD journey.

Lastly, I would like to express my heartfelt thanks to my parents and my wife for their unwavering support and love. Their encouragement has been my strength and has empowered me to pursue my goals with confidence.

List of Publications

Related to the Thesis :

1. **Siyi Hu**, Fengda Zhu, Xiaojun Chang, Xiaodan Liang. "UPDeT: Universal Multi-agent Reinforcement Learning via Policy Decoupling with Transformers." Proceedings of the 9th International Conference on Learning Representations, pp.1-15. 2021

2. **Siyi Hu**, Chuanlong Xie, Xiaodan Liang, Xiaojun Chang. "Policy Diagnosis via Measuring Role Diversity in Cooperative Multi-agent RL." Proceedings of the 39th International Conference on Machine Learning, pp.9041-9071, 2022

3. **Siyi Hu**, Yifan Zhong, Minquan Gao, Weixun Wang, Hao Dong, Xiaodan Liang, Zhihui Li, Xiaojun Chang, Yaodong Yang. "MARLlib: A Scalable and Efficient Multi-agent Reinforcement Learning Library." Journal of Machine Learning Research 24, pp.1-23, 2023

Others :

4. Yifan Zhong, Jakub Grudzien Kuba, Xidong Feng, **Siyi Hu**, Jiaming Ji, Yaodong Yang. "Heterogeneous-agent reinforcement learning." Journal of Machine Learning Research 25, pp.1-67, 2024

5. Jiarong Liu, Yifan Zhong, **Siyi Hu**, Haobo Fu, Qiang Fu, Xiaojun Chang, Yaodong Yang. "Maximum Entropy Heterogeneous-Agent Reinforcement Learning." Proceedings of the 13th International Conference on Learning Representations, pp.1-15. 2024

6. Ceyao Zhang, Kaijie Yang, **Siyi Hu**, Zihao Wang, Guanghe Li, Yihang Sun, Cheng Zhang, Zhaowei Zhang, Anji Liu, Song-Chun Zhu, Xiaojun Chang, Junge Zhang, Feng Yin, Yitao Liang, Yaodong Yang. "ProAgent: Building Proactive Cooperative

Agents with Large Language Models." Proceedings of the 38th Annual AAAI Conference on Artificial Intelligence, pp.17591-17599, 2024

Table of Contents

LIST OF FIGURES

FIGURE **Page**

List of Tables

Table **Page**

CHAPTER 1

INTRODUCTION

Reinforcement Learning (RL) [91] is a cornerstone paradigm for autonomous decision-making, enabling agents to optimize their behaviors through interactions with their environments [61, 63, 80]. Its application ranges from robotics [26, 43, 95] to game theory [37, 48, 108], focusing on achieving optimal outcomes through sequential decisions. As RL is applied to increasingly complex and dynamic scenarios, Multi-Agent Reinforcement Learning (MARL) has emerged as a vital extension [12, 65, 110]. In MARL, multiple agents operate within a shared environment, engaging in a spectrum of behaviors from competitive [11, 78, 111] to cooperative [5, 46, 79]. This transition from single-agent to multi-agent scenarios introduces new challenges [45, 89, 92], offering rich opportunities for exploring innovative learning dynamics.

The complexities in MARL primarily stem from agent interactions, which affect the environmental state and reward structures [66]. These interactions create a non-stationary environment, complicating the learning algorithms and their implementation as strategies and actions of agents continuously influence the decision-making context. Understanding and developing efficient strategies for MARL involves addressing key issues such as credit assignment [24, 75, 76], formulating stable strategies in adversarial settings [11, 53, 85], and scaling algorithms to accommodate large number of agents [103, 113, 116], each posing unique challenges.

Multi-task MARL As the field of MARL continues to evolve, an important development has been the integration of multi-task learning strategies [18, 99, 112]. This

advancement enables agents to train across multiple tasks simultaneously, enhancing their generalization capabilities and robustness, and reducing the need for distinct models for each scenario [35, 36, 106]. Conventional algorithms in MARL require the input and the output dimensions to be fixed, makes the multi-task learning impossible. Thus the application of current methods is limited in real-world applications.

A significant contribution to this area is the development of UPDeT [35], a transformative approach that introduces a scalable and adaptable framework into the multi-task learning domain, marking a departure from traditional methods. The focus is on developing architectures and strategies that enable effective knowledge sharing among agents [55, 60, 82], boosting MARL system efficiency and adaptability. Such multi-task learning frameworks require innovative algorithm designs to manage the complexities of multiple agents learning within shared environments [39, 42, 118].

Task-first MARL Current researches focus on developing algorithms on the tasks they are good at but lack the study of why the performance declines on other tasks [32, 70, 94]. Sometimes, even adopting the state-of-the-art algorithms does not guarantee an optimized performance [24, 102, 109]. From this perspective, understanding the relationship between algorithms and tasks is instrumental. Firstly, it can help elucidate the strengths and limitations of existing MARL algorithms, providing clear insights into their performance boundaries and operational effectiveness. Secondly, it emphasize that the design of more specialized algorithms should be precisely tailored to the unique dynamics of particular multi-agent environments.

We introduce a metric to help measure the characteristic of different MARL tasks and its relationship to the algorithms used [33]. We define role diversity as a critical measurement for analyzing and improving the effectiveness of policies within multi-agent environments, providing a framework for diagnosing policy performance and guiding the training process. By systematically evaluating the efficiency of various MARL algorithms in different contexts using role diversity, this research not only enhances our theoretical knowledge but also drives practical advancements.

A Unified Library for MARL Single-agent RL has achieved successful unification for both algorithms (e.g., SpinningUp [2], Tianshou [107], RLlib [54], Dopamine [13], and Stable-Baselines series [21, 29, 74]) and environments (e.g., Gym [10], Gymnasium [27]). However, MARL faces unique challenges in developing a comprehensive and high-quality library.

Firstly, MARL algorithms are diverse, targeting group cooperation or individual competition and varying in agent parameter sharing strategies, such as HATRPO [45] and MAPPO [109]. They also differ in central information utilization, e.g., VDN [87] vs. MADDPG [56]. Libraries like EPyMARL [70] attempt to unify MARL algorithms using independent learning, centralized critic, and value decomposition categorization but fail to address all these challenges. The diversity of MARL algorithms remains a major unification obstacle.

Secondly, multi-agent environment interfaces are inconsistent, reflecting their task-specific designs (e.g., asynchronous interaction in Hanabi, action masks in SMAC [79], and mixed local observation and global state in MAgent [113]). This inconsistency complicates unified agent-environment interaction processing, causing coupling issues between algorithm implementation and task environment. PettingZoo [93] collects diverse multi-agent tasks but is inconvenient for CTDE-based algorithms due to missing global state and action mask information. Solutions like the MAPPO benchmark [109], which provides unique runner scripts for each environment, pose long-term maintenance challenges and hinder new task extensions.

Acknowledging the critical need for standardized tools and frameworks, there is an urgent need for a comprehensive library specifically designed to streamline the development and deployment of MARL systems. The primary goal of such a library is to alleviate the common challenges associated with compatibility and integration that researchers and practitioners often face. By providing a robust and integrated set of tools, it enables users to devote more of their efforts towards strategic development and innovation within MARL [34, 70, 114]. Other benefits include simplifying the technical processes involved in setting up and running MARL experiments, as well as enhancing the accessibility and reproducibility of research outcomes [34]. As a result, this accelerates the advancement of MARL technologies by ensuring that enhancements in algorithmic strategies or environment models can be easily adopted and built upon by the broader community.

We create MARLlib [34], a comprehensive library designed to facilitate the integration, testing, and deployment of MARL algorithms, marks a significant contribution to achieving a unified framework for MARL. MARLlib provides robust tools that enhance the reproducibility and accessibility of MARL research, allowing enhancements in algorithmic strategies or environment models to be readily adopted by the community.

Contributions This thesis has made several significant contributions to the field of MARL, pushing forward the capabilities of these systems in managing complex, dynamic

multi-agent environments. The main contributions are outlined as follows:

1. We introduce the Universal Policy Decoupling Transformer (UPDeT), a novel transformer-based architecture for MARL. UPDeT leverages the self-attention mechanism to address critical challenges such as partial observability and coordination among multiple agents. This development marks a significant advancement over traditional methods, enhancing both the scalability and adaptability of MARL systems.

2. We establishe role diversity as a crucial metric for analyzing and improving the effectiveness of policies in MARL. By systematically studying different dimensions of role diversity,Äîaction-based, trajectory-based, and contribution-based,Äîthis work provides a nuanced framework for diagnosing policy performance and guiding the training process within cooperative MARL environments.

3. We build MARLlib, a comprehensive library designed to facilitate the integration, testing, and deployment of MARL algorithms. MARLlib addresses the prevalent challenges in the MARL community related to algorithm compatibility and integration, providing a robust platform that enhances the reproducibility and accessibility of MARL research.

These contributions are instrumental in advancing theoretical knowledge and practical applications in MARL, supporting the development of more sophisticated, efficient, and adaptable multi-agent systems. They align with the overarching goal of this thesis to move towards a unified framework for MARL, setting a strong foundation for future research in this rapidly evolving field. The organization of the thesis is as follows:

Chapter 1: Introduction This chapter provides a comprehensive introduction to the field of MARL. It outlines the significance of MARL, the challenges it addresses, and the specific objectives and contributions of the thesis. The chapter sets the stage for the detailed discussions and analyses presented in subsequent chapters.

Chapter 2: Literature Review The literature review covers the foundational theories and recent advancements in RL and MARL. This chapter discusses key concepts, methodologies, and the state-of-the-art approaches in MARL, highlighting the gaps and challenges that this thesis aims to address.

Chapter 3: Universal Multi-agent Reinforcement Learning with Transformers This chapter introduces the novel transformer-based architecture, Universal Policy Decoupling Transformer (UPDeT), designed to enhance policy learning in MARL. It details the theoretical underpinnings, the policy decoupling strategy, and the temporal unit structure of the proposed framework. Experimental results demonstrating the efficacy of

UPDeT in various MARL scenarios are also presented.

Chapter 4: Policy Diagnosis via Measuring Role Diversity in Cooperative Multi-agent RL In this chapter, the concept of role diversity is introduced as a metric for analyzing and optimizing policy performance in cooperative MARL environments. The chapter provides a detailed methodology for quantifying role diversity and explores its impact on various training strategies. Experimental evaluations highlight the benefits of incorporating role diversity into policy diagnosis.

Chapter 5: One Platform for All: A Unified Multi-agent RL Library This chapter discusses the development of MARLlib, a comprehensive library designed to unify and standardize MARL algorithms and environments. It covers the motivation behind creating the library, its features, and the benefits it offers for MARL research and development. Comparative analyses with existing libraries and benchmarks demonstrate the robustness and versatility of MARLlib.

Chapter 6: Conclusion and Future Works The final chapter summarizes the key contributions of the thesis, reflecting on the advancements made in the field of MARL through the proposed frameworks and methodologies. It also outlines potential directions for future research, emphasizing the ongoing challenges and opportunities in developing more sophisticated and adaptable multi-agent systems.

CHAPTER 2

LITERATURE REVIEW

2.1 Deep Reinforcement Learning

Reinforcement Learning (RL) is a type of machine learning where an agent learns to make decisions by performing actions in an environment to maximize some notion of cumulative reward [40, 91]. RL is modeled as a Markov decision process (MDP) [6] where the outcomes are partly random and partly under the control of a decision maker.

The key components of any RL system include the environment, states, actions, rewards, policies, and value functions. Research in this field has been propelled by the formulation and solution of the Bellman equation [6], which provides a recursive decomposition for the value functions associated with optimal policies. This area has seen extensive application ranging from automated control systems to economics and game playing, where systems learn to make a series of decisions that maximize a reward metric over time.

Deep Reinforcement Learning (DRL) has emerged as a powerful technique combining reinforcement learning principles with deep learning [63]. This integration has facilitated the creation of models that can process high-dimensional sensory inputs and make decisions accordingly, thereby handling complex environments that were previously intractable with traditional RL methods.

The foundation of DRL can be traced back to the seminal work by [62], who introduced the Deep Q-Network (DQN). This algorithm successfully integrated deep neural networks

with Q-learning [105], showing remarkable performance on multiple Atari 2600 games, purely from pixel inputs. This breakthrough underscored the potential of deep learning architectures in approximating value functions and policies directly from raw sensory data.

Following this, various improvements and extensions to DQN were proposed. Double DQN introduced by [98] addressed the overestimation bias of Q-values in DQN. Dueling Network Architectures for Deep Reinforcement Learning by [104] enhanced the network's ability to distinguish between state values and the advantages of particular actions.

The introduction of Policy Gradient methods, specifically those employing Actor-Critic frameworks such as A3C (Asynchronous Advantage Actor-Critic) by [61], further expanded the capabilities of DRL. These methods allow direct optimization of the policy function, offering a more stable and robust learning process in environments with high-dimensional or continuous action spaces.

Recently, Proximal Policy Optimization (PPO) by [80] has gained prominence due to its simplicity and effectiveness, reducing the complexity of hyperparameter tuning while achieving high performance. Furthermore, the integration of DRL with other areas such as Natural Language Processing [97] and computer vision [51] has opened new avenues for cross-disciplinary applications. The use of DRL in strategic game playing was famously demonstrated by AlphaGo [84], which defeated a world champion in the complex board game of Go, highlighting the technique's strategic decision-making capabilities. DRL continues to be a vibrant area of research, with ongoing innovations that enhance its efficiency, scalability, and applicability across various domains.

2.2 Multi-agent Reinforcement Learning

Multi-agent Reinforcement Learning (MARL) extends the classical reinforcement learning framework to scenarios involving multiple agents, each influencing and adapting to the actions of others. This complexity introduces a non-stationary environment where conventional single-agent learning algorithms underperform. MARL is crucial in domains requiring sophisticated agent coordination, such as precision agriculture, underwater exploration, and autonomous vehicle systems, where the interplay of multiple agents significantly impacts overall system performance [25, 67, 68].

2.2.1 Multi-agent Systems

There are several different ways to model a multi-agent system, based on *Stochastic Games*. Stochastic Games model the interactions among multiple agents and their environment. Two settings are commonly seen: *Fully Observable Stochastic Games (FOSGs)* and *Partially Observable Stochastic Games (POSGs)*.

FOSGs. FOSGs are expanded versions of *Markov Decision Processes (MDPs)* [77, 108]. MDPs are foundational in reinforcement learning, defined by a tuple $\langle S, A, P, R, \gamma \rangle$ where S represents the state space, A the action space, P the transition probability function, R the reward function, and γ the discount factor. In MARL, the interaction of multiple agents turns an MDP into a stochastic game or Markov game. Here, the game dynamics are described by the set $\langle I, S, A, P, \{R_i\}_{i \in I}, \gamma \rangle$, where I denotes the set of agents, each with their own actions and rewards, thereby capturing the impact of collective and individual strategies on the state transitions and outcomes.

POSGs. Transitioning from FOSGs, POSGs significantly extend the complexity of decision-making processes by introducing uncertainty in state observations. This form of stochastic game is particularly relevant in real-world scenarios characterized by information asymmetry or sensor limitations, where agents do not have access to the complete state of the environment. Conceptually, POSGs are closely related to *Partially Observable Markov Decision Processes (POMDPs)* [72].

POMDPs. A POMDP models decision-making challenges under uncertainty with a formal tuple $\langle S, A, T, R, O, Z, \gamma \rangle$, where S denotes the state space, A the action space, $T : S \times A \times S \to [0,1]$ the state transition probabilities, $R : S \times A \times S \to \mathbb{R}$ the reward function, O the observation space, $Z : S \times O \to [0,1]$ the observation function, and γ the discount factor. In POSGs, each agent deals with its own observation space O_i and observation function Z_i, which define the probability of observing a particular outcome given the current state and action.

Dec-POMDPs. Decentralized Partially Observable Markov Decision Processes (Dec-POMDPs) [66] expand on POMDPs by introducing multiple agents that must act based on their own incomplete and noisy observations. In a Dec-POMDP, described by the tuple $\langle I, S, A, T, R, O, Z, \gamma \rangle$, each agent operates under partial observability of the environment, influenced by both the joint actions and observations of the group. The challenge lies in coordinating strategies under uncertainty, requiring agents to infer not just the environment's state but also the potential actions of their peers. This model is critical for designing systems where agents need to autonomously adapt to complex, dynamic settings without complete information.

Interact Form. The design of multi-agent systems also needs to account for whether the interactions among agents occur in a turn-based (*Extensive-form*) [81] format or simultaneously (*Normal-form*). Turn-based games, where agents take actions one after another, often require strategies that can predict and counteract opponents' future moves, as seen in strategic board games like Go or Chess [85]. Here, MARL can be used to learn complex strategies over a sequence of moves, with each agent's decision influenced by the predicted reactions of their adversaries. On the other hand, in simultaneous games, all agents act at the same time, typically without knowledge of the concurrent decisions of other agents [79]. This setup is common in many real-world applications like financial markets or tactical military simulations, where agents must make decisions based on incomplete information and a prediction of other agents' actions.

Task Mode. MARL proves highly effective in systems that necessitate coordinated action among multiple agents in diverse multi-agent task modes, such as *cooperative*, *competitive*, or a combination of both task modes (*mixed*), depending on the objectives and operational constraints of the system. In cooperative modes [5, 71, 79], agents work together towards a common goal, such as in robotic swarms where harmonized movements and tasks are crucial for accomplishing complex activities like area surveillance or search-and-rescue operations [69]. Conversely, in competitive modes, each agent aims to maximize its own payoff at the expense of others [4, 56, 78], which is often seen in resource allocation tasks within telecommunications networks where agents compete for bandwidth. These systems significantly benefit from the MARL approach by leveraging the collective capabilities of agents to achieve tasks that are beyond the scope of individual actors, demonstrating the power of learning and adaptation in shared environments.

Regardless of the game structure or the task mode, MARL applications in these multi-agent systems emphasize the importance of strategic decision-making and adaptability. Techniques from diverse learning paradigms are utilized to engineer agents capable of operating effectively in dynamic, often unpredictable environments. This dual capability not only enhances the performance in specific tasks but also broadens the applicability of MARL to a wider range of complex problems.

2.2.2 Training Paradigms

Traditional algorithms in RL such as Q-learning are often adapted for scenarios where agents learn independently - a paradigm known as Decentralized Training with Decentralized Execution (DTDE) [19, 92]. Each agent independently optimizes its policy,

which is particularly useful in environments where communication among agents during execution is either limited or infeasible.

However, the complexities of MARL often require more sophisticated approaches such as Centralized Training with Decentralized Execution (CTDE) [56, 76, 90]. The CTDE paradigm allows for the collection and use of global information during the training phase but requires that each agent only uses its own local observation for decision-making during execution. This approach is beneficial in scenarios where agents must operate independently yet benefit from learning in a rich, informative environment during their training. Techniques such as actor-critic methods and policy gradient strategies are often employed under CTDE. These methods enable the agents to handle continuous action spaces and complex policy structures, with a central critic that aggregates global state information to guide the policy optimization of decentralized actors.

On the other hand, the Centralized Training with Centralized Execution (CTCE) paradigm [38, 86, 89] involves both centralized training and execution. This approach is suitable for environments where centralized control during execution is possible and can be advantageous, such as in tightly coordinated tasks where synchronicity is critical. Under CTCE, all agents' actions are chosen based on the full global state, making it possible to optimize joint strategies directly.

The choice of training and execution paradigms in MARL depends heavily on the specific requirements and constraints of the application domain, including the need for real-time decision making, the availability of global state information, and the level of cooperation or competition among the agents.

2.3 Challenges

MARL presents a unique set of challenges that stem from the complexities of coordinating multiple intelligent agents within a shared environment. These challenges are amplified by the agents' need to learn and adapt not only from their interactions with the environment but also from the actions of other agents. As such, MARL systems must address several critical issues to effectively implement learning and decision-making strategies that can cope with the dynamic and often unpredictable nature of multi-agent scenarios.

Non-Unique Learning Goals In MARL, defining clear learning goals presents a significant challenge, contrasting sharply with single-agent scenarios where maximizing

long-term return is typically the sole objective [8, 9, 83]. MARL goals can be multifaceted and vague, often leading to debates about appropriate success criteria. While convergence to a Nash Equilibrium (NE) is traditionally valued, its relevance is questioned due to the bounded rationality of agents who may not achieve perfect reasoning or infinite mutual modeling [7, 8]. Instead, practical MARL might focus on developing optimal strategies for agents within a specific class, as suggested in the literature. This shift from an equilibrium agenda to an AI agenda reflects deeper concerns about the suitability of NE as the dominant performance criterion. Alternative approaches, such as cyclic equilibrium concepts, stability and rationality requirements, and no-regret algorithms, propose different success metrics that accommodate the complex dynamics of MARL environments.

Non-Stationarity Another profound challenge in MARL is non-stationarity, caused by the concurrent learning activities of multiple agents, which alters the environment dynamically [3, 17, 59]. This non-stationarity disrupts the stationary Markovian assumptions underlying traditional single-agent RL methods, complicating the direct application of these techniques. Each agent's actions influence not only their own rewards but also affect the state evolution and the rewards of other agents, requiring strategies that account for the joint behavior of all agents. Although independent learning-where each agent optimizes its policy as if in a stationary environment-might empirically perform well, it generally struggles to converge in theory [23, 92, 96]. Non-stationarity is a well-recognized issue in MARL, extensively reviewed in the literature, highlighting the necessity for models that effectively address the mutable nature of agent interactions in these settings.

Knowledge Transfer and Scalability Training agents to perform across various interconnected tasks increases learning efficiency and agent versatility [18, 99, 112]. This method leverages transfer learning or multi-task learning techniques to apply knowledge gained from one task to enhance performance in others. A significant challenge in this approach is managing the learning balance to prevent negative transfer without causing overfitting to the nuances of specific tasks [55, 60, 82]. Recent advancements include modular networks and policy distillation, which allow agents to learn task-specific strategies while sharing a core knowledge base. The importance of scalable learning mechanisms is also emphasized, particularly in multi-task environments where agents confront diverse and changing task demands [35, 36, 106]. Techniques borrowed

from CTDE strategy, common in cooperative MARL scenarios, are in urgent need for addressing these complexities. The problem centralized on policy development on task execution, ensuring that agents dynamically adapt to new challenges while capitalizing on shared resources and collective learning [39, 42, 118].

2.4 Libraries and Benchmarks

Significant progress has been achieved in single-agent reinforcement learning with the introduction of robust libraries such as *OpenAI Baselines*, *RLlib*, and *Tianshou* [21, 54, 107]. However, constructing a comprehensive and high-quality library for MARL introduces unique challenges. Unlike areas like image classification where standardized datasets such as ImageNet are prevalent [20], MARL lacks uniform datasets. This field typically involves highly customizable scenarios that vary in the number of agents, map dimensions, reward mechanisms, and the status of participating units. This variability complicates the establishment of a consistent research baseline. While platforms like PettingZoo [93] and Melting Pot [52] strive to provide unified and scalable test environments, MARL research still faces hurdles due to the diverse requirements and data structures that affect the implementation and performance of learning algorithms.

Additionally, the adaptation of MARL environments to various algorithms poses another significant challenge. The simultaneous presence of cooperative and competitive goals within the same framework can prevent the straightforward application of strategies designed specifically for cooperative contexts. Moreover, certain algorithms that demand comprehensive environmental data like the global state might not function optimally in settings that do not provide such information. These discrepancies lead to a situation where existing libraries [32, 70, 109] exhibit limited coverage of tasks and lack a cohesive algorithmic structure, which ultimately hampers their extensibility and leads to cumbersome codebases. It is essential to develop a universal MARL framework that can segregate environments from algorithms effectively, ensure broad compatibility, and provide a standardized suite for evaluating MARL systems.

CHAPTER 3

UNIVERSAL MULTI-AGENT REINFORCEMENT LEARNING WITH TRANSFORMERS

3.1 Introduction

Multi-task Multi-Agent Reinforcement Learning (MARL) presents significant challenges to traditional agent training models, demanding that agents adapt across a diverse range of tasks without the need for retraining from scratch. This requires architectures capable of dynamically adjusting to the varied characteristic of different game scenarios or real-world applications.

The current state of MARL methodologies predominantly rely on action-value functions (Q) to develop multi-agent algorithms [57, 76, 90]. However, these algorithms often struggle to effectively differentiate between observations from various environmental entities. Typically, these methods combine observations into a single input vector [22, 76, 115], operating under the assumption that neural networks can inherently distinguish these inputs to map optimally to policies. This approach can overlook the detailed physical meanings embedded in each action and the specific correlations between observations and outputs. Moreover, if observations from different agents are not clearly separated, it can misdirect individual functions and compromise the efficiency of a centralized value function. Additionally, the prevalent focus on fixed input and output dimensions restricts the flexibility necessary for effective transfer learning, thus limiting the practical deployment of these models in real-world settings.

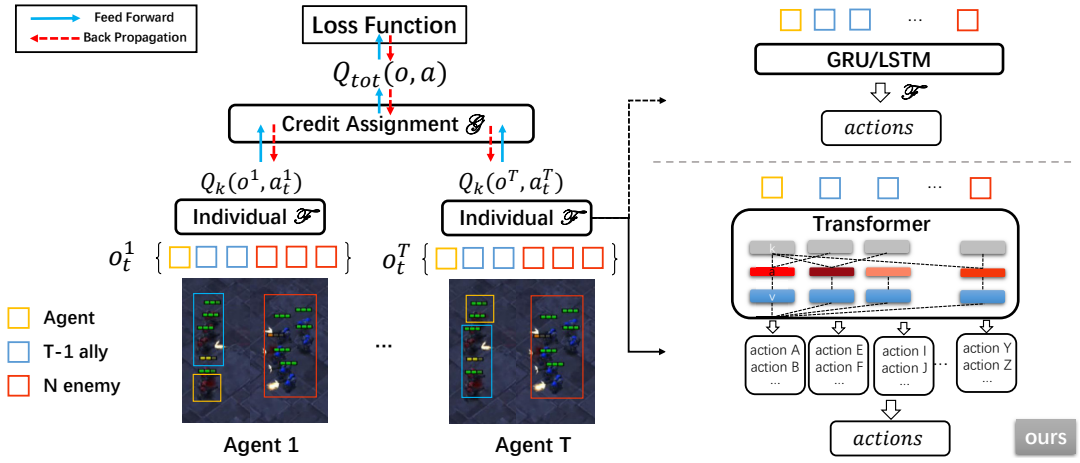


Figure 3.1: An overview of the MARL framework. Our work replaces the widely used GRU/LSTM-based individual value function with a transformer-based function. Actions are separated into action groups according to observations.

To address these deficiencies, we propose the development of an innovative MARL framework that surpasses the traditional constraints on input and output dimensions. Our method is designed to be compatible with existing MARL methodologies while enhancing system explainability and performance across both single-task and multi-task scenarios. This approach aims to significantly improve the potential for transfer learning across a broad spectrum of MARL applications.

3.2 Multi-task MARL

3.2.1 Preliminaries

In this section, we introduce the fundamentals of multi-agent learning paradigm, which is essential for understanding how individual behaviors impact collective dynamics within interactive environments. We focus on settings *Partially Observable MDPs* combined with *Stochastic Games* and their decentralized extensions where agents operate with limited information.

Markov Decision Processes (MDPs) In reinforcement learning (RL), agent-environment interactions are modeled using Markov Decision Processes (MDPs). An MDP is defined by the tuple $\langle S, A, P, R, \gamma \rangle$:

- S is the state space,

- A is the set of actions,
- $P : S \times A \times S \to [0,1]$ is the transition probability,
- $R : S \times A \times S \to \mathbb{R}$ is the reward function,
- γ is the discount factor for future rewards.

Each time step involves: 1. the agent observing its state, 2. choosing an action, 3. receiving a reward, and 4. transitioning to a new state. The goal is to develop a policy π that maximizes cumulative discounted rewards.

Partially Observable MDPs (POMDPs) POMDPs extend MDPs to scenarios with incomplete state information. Defined by the tuple $\langle S, A, O, P, Z, R, \gamma \rangle$, they include:

- O as the observation space,
- $Z : S \times O \to [0,1]$ as the observation probability function.

In POMDPs, agents make decisions based on partial observations, making this model suitable for environments where full state visibility is restricted.

Decentralized POMDPs (Dec-POMDPs) Dec-POMDPs address multi-agent scenarios under partial observability. Characterized by the tuple $\langle I, S, A, T, R, O, Z, \gamma \rangle$, they include:

- I is the set of agents,
- $A = A_1 \times A_2 \times \cdots \times A_N$ is the joint action space,
- O as the composite observation space of all agents,
- $Z : S \times A \times O \to [0,1]$ as the observation probability.
- R specifies the reward shared by all agents.

Dec-POMDPs require agents to base decisions on both individual observations and inferred information about other agents' states, aiming to maximize collective or individual rewards through coordinated policies.

3.2.2 Transformers as Policy

Drawing on the self-attention mechanism [100], we introduce a transformer-based MARL framework termed Universal Policy Decoupling Transformer (**UPDeT**). UPDeT organizes the action space into distinct action groups that correlate with specific observation

entities, forming matched pairs of observation entities and action groups. This organization enables precise alignment of actions with relevant observations. Employing a self-attention mechanism, UPDeT learns the intricate dynamics between various observation entities, enhancing the policy's effectiveness at the action-group level. This method, which we refer to as *Policy Decoupling* (detailed in Sec. 3.2.3), leverages the self-attention maps and embeddings of each observation entity to refine policy decisions within grouped actions. By integrating the transformer architecture with the policy decoupling strategy, UPDeT achieves significant performance improvements over traditional RNN-based models in multi-agent settings.

Here we outline the mathematical formulation of UPDeT, specifically focusing on the computation of a global Q-function using a self-attention mechanism.

Initially, agent observations are transformed into semantic embeddings to accommodate various observation spaces. For instance, if an agent a_i at time step t observes k entities, these observations are embedded using an embedding layer E:

$$e_i^t = \{E(o_{i,1}^t), \ldots, E(o_{i,k}^t)\}. \tag{3.1}$$

Here, i represents the index of the agent within the set $\{1, \ldots, n\}$.

Subsequently, the value functions for each agent are estimated as follows:

$$q_i^t = Q_i(h_i^{t-1}, e_i^t, \mathbf{u}_t). \tag{3.2}$$

In this model, h_i^{t-1} denotes the historical hidden state from the previous time step, reflecting the dependency of the POMDP policy on past information. The observation embedding e_i^t and a candidate action $u_i^t \in U$ are also incorporated, with θ_i defining the parameters of Q_i.

The global Q-function, Q_π, integrates these individual value functions:

$$Q_\pi(s_t, \mathbf{u}_t) = F(q_1^t, \ldots, q_n^t), \tag{3.3}$$

where F_i is a credit assignment function, defined by ϕ_i for each agent a_i, as commonly applied in value decomposition networks (VDN), summing the individual Q-values.

Following [100], the self-attention mechanism utilizes three matrices-keys ($\mathbf{K}$), queries ($\mathbf{Q}$), and values ($\mathbf{V}$)-to compute the attention as:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}}\right)\mathbf{V}, \tag{3.4}$$

with d_k representing the dimension scaling factor. In our approach, self-attention is employed to derive meaningful relationships from the embeddings of observed entities

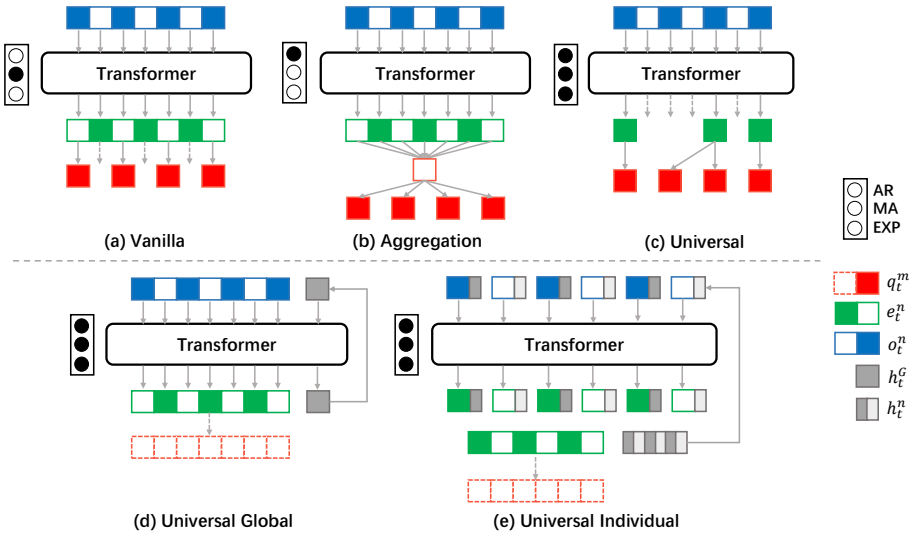


Figure 3.2: The figure depicts three distinct approaches to policy decoupling (top segment) alongside two types of temporal unit variants (lower segment). *AR*, *MA*, and *EXP* denote Action Restriction, Multi-task, and Explainable, respectively. The symbols o, e, q, and h are used to represent observations, embeddings, Q-values, and hidden states, which encompass n observational entities and m potential actions. The symbol G signifies the global hidden state, and t indicates the current timestep. Black circles are used to mark the variant that includes the respective feature, with variant (d) being our suggested UPDeT, which demonstrates superior performance. Comprehensive descriptions of all five variants are detailed in Section 3.2.3.

and global temporal information. To adapt this for decentralized MARL, we define separate key, query, and value matrices for each agent, treating them as identical across the layers of the transformer. The process is structured as:

$$R_i^1 = \{h_i^{t-1}, e_i^t\}, \tag{3.5}$$

$$R_i^{l+1} = \text{Attention}(R_i^l, R_i^l, R_i^l). \tag{3.6}$$

Finally, the output of the last transformer layer, R_i^L, is projected onto the value function space using a linear function P:

$$Q_i(h_i^{t-1}, e_i^t, u_i) = P(R_i^L, u_i). \tag{3.7}$$

3.2.3 Policy Decoupling

Implementing a single transformer with a self-attention mechanism may not sufficiently address diverse policy distributions. A versatile mapping function P specified in Eq. 3.7 is essential for managing varying input and output dimensions and enhancing representation capabilities. We introduce a concept called *policy decoupling* as a fundamental

component of UPDeT, utilizing correlations between inputs and outputs to refine this strategy.

The essence of the policy decoupling strategy is distilled into three primary objectives:

- Point ①: Flexibility in policy dimensions. Traditional transformer blocks require the output dimension to be equal to or less than the input dimension, which is often unsuitable for MARL tasks where the number of actions may exceed the number of entities.
- Point ②: Simultaneous multi-task handling. A consistent model architecture that does not necessitate new parameters for each new task is crucial. However, achieving this often complicates fulfilling the requirements of Point ①.
- Point ③: Enhanced model explainability. Replacing conventional RNN-based models with a structure that provides clearer insights into policy generation is preferable.

In response to these objectives, we propose three methods of policy decoupling: **Vanilla Transformer**, **Aggregation Transformer**, and **UPDeT** (ours), detailed further in Fig. 3.2 and discussed in the experiments section as our baselines.

With UPDeT, the challenge lies in establishing a robust mapping from entity features to the policy distribution. UPDeT begins by aligning each input entity with its relevant output policy segment. This alignment, typically straightforward in MARL tasks due to the commonality of interactive actions, significantly simplifies the model's learning process. When an entity feature corresponds to multiple interactive actions, we categorize the action space into several groups. Each group contains actions associated with a specific entity. This process, depicted in the left part of Fig. 3.3, ensures that if an entity feature's action-group contains more than one action, a shared fully connected layer is employed to map these to the action dimension. Conversely, if an entity feature lacks a corresponding action, it is discarded without loss of information, as the transformer aggregates essential details into each output. The complete pipeline of UPDeT is illustrated in the right part of Fig. 3.3. With this approach, UPDeT imposes no constraints on action choices and introduces no new parameters for different scenarios, enabling a single model to be trained and deployed universally across multiple tasks. Moreover, matching entity features with action-groups as shown provides explainability through attention heatmaps, aligning with Point ③.

Policy decoupling in UPDeT allows handling multiple tasks simultaneously by maintaining a fixed architecture that does not require the introduction of new parameters

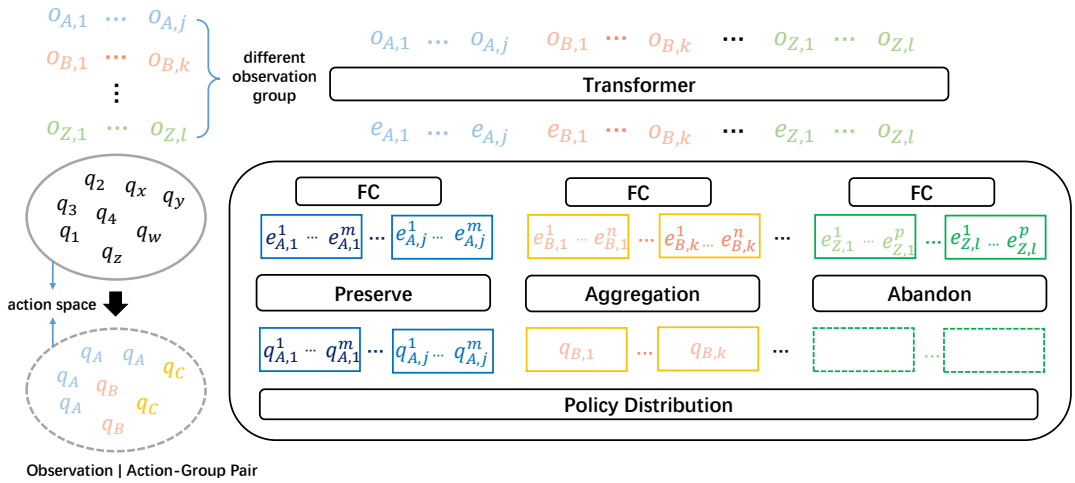


Figure 3.3: The primary architecture of our proposed UPDeT is outlined, where o, e, and q symbolize the observation entities, feature embeddings, and Q-values for each action, respectively. Three operations, 'preserve', 'aggregation', and 'abandon', are employed to formulate the policy distribution without the addition of new parameters. Further information on these operations is available in Section 3.2.3.

for new tasks. The policy is segmented into various action groups according to the corresponding observation entities, enabling the model to adapt its behavior based on specific task requirements without overhaul. This strategy significantly reduces the complexity traditionally associated with multi-task learning and improves the model's scalability and transfer capabilities across different MARL environments.

3.2.4 Temporal Unit Structure

Transformers equipped with policy decoupling strategies face challenges in Dec-POMDPs [66], where each agent a selects actions based on a policy $\pi^a(u^a|\tau^a)$, with u and τ representing the action and action-observation history, respectively. While recurrent models like GRUs and LSTMs capture this history through hidden states, integrating these states with transformer blocks has been underexplored. We propose two methodologies in UPDeT to integrate hidden states:

1) **Global Temporal Unit:** This method integrates a global hidden state directly into the transformer block, as shown in Eq. 3.8:

$$
\begin{aligned}
R^1 &= \{h_G^{t-1}, e_1^t\}, \\
R^l &= \text{Attention}(R^{l-1}, R^{l-1}, R^{l-1}), \\
\{h_G^t, e_L^t\} &= R^L.
\end{aligned} \tag{3.8}
$$

Here, G indicates a 'global' context, which simplifies integration and ensures robust performance across various scenarios.

2) **Individual Temporal Unit:** In contrast to the global method, this approach maintains a distinct hidden state for each entity, enhancing control over historical information:

$$\begin{aligned} R^1 &= \{h_1^{t-1}, \ldots, h_j^{t-1}, e_1^t\}, \\ R^l &= \text{Attention}(R^{l-1}, R^{l-1}, R^{l-1}), \\ \{h_1^t, \ldots, h_j^t, e_L^t\} &= R^L. \end{aligned} \tag{3.9}$$

Although this method offers precise tracking of history for each entity, it increases the complexity of learning individual hidden states.

These variants are further evaluated in Section 4.1.2, discussing their performance and trade-offs.

3.2.5 Optimization

For optimization, we employ the standard squared TD error used in DQNs [63]:

$$L(\theta) = \sum_{i=1}^{b} \left[\left(y_i^{DQN} - Q(s, u; \theta) \right)^2 \right], \tag{3.10}$$

where b denotes the batch size. In partially observable settings, we extend the concept of Deep Recurrent Q-Networks (DRQN) [28], replacing traditional GRU [16] or LSTM [30] units with our transformer-based temporal units to enhance sequential decision-making.

3.3 Results

We evaluate UPDeT and its policy decoupling variants in challenging StarCraft II micromanagement scenarios, comparing it against a traditional RNN-based model. UPDeT shows significant performance improvements in both single and multiple scenario transfer tasks.

3.3.1 Single Scenario

In this series of experiments, we examine the effectiveness of UPDeT across various scenarios from SMAC [79], including *3 Marines vs 3 Marines* (3m, Easy), *8 Marines vs 8 Marines* (8m, Easy), *4 Marines vs 5 Marines* (4m_vs_5m, Hard+), and *5 Marines vs*

6 Marines (6m_vs_7m, Hard). The scenarios use only the player's units as agents, and defeated enemy units are excluded from the action space.

We also review state-of-the-art MARL methods like VDN [90], QMIX [76], and QTRAN [31], detailed at `https://github.com/oxwhirl/pymarl`. Although other methods like COMA [24] and IQL [92] were considered, their inconsistent performance is documented in several studies [57, 76, 115]. UPDeT is integrated with VDN, QMIX, and QTRAN, showing enhancements over the traditional GRU-based model. The robustness of these results is supported by eight replication trials, with the win rate fluctuations and standard deviations presented in Fig. 3.4.

The results of applying various policy decoupling strategies are detailed in Fig. 3.4. Vanilla Transformer, serving as our baseline for transformer models, achieves only partial success. It maps each output embedding either to an action or discards it, yet fails to converge in our tests. The Aggregation Transformer, an adaptation of the Vanilla model, consolidates embeddings into a single global one before projecting them onto a policy distribution. This model meets limited criteria and underperforms compared to the GRU model, underscoring the necessity for policy decoupling strategies in transformer models to surpass traditional RNN performance.

We then implement UPDeT to explore the optimal architecture for temporal units, as shown in Fig. 3.4. Our findings indicate that omitting a hidden state significantly degrades performance, whereas incorporating a global hidden state fosters faster convergence without sacrificing final results. For a broader evaluation, UPDeT is integrated with VDN, QMIX, and QTRAN to assess its effectiveness against RNN-based models in the 5m_vs_6m (Hard) scenario, as illustrated in Fig. 3.4. When combined with UPDeT, all three MARL methods exhibit notable improvements over the GRU model. This analysis confirms that UPDeT enhances any contemporary MARL method to achieve superior outcomes.

Further, UPDeT is integrated with VDN and evaluated across varying scenarios from Easy to Hard+, as depicted in Figs. 3.4. UPDeT consistently excels in easier settings and significantly outperforms the GRU model in tougher scenarios. In the 4m_vs_5m (Hard+) scenario, UPDeT achieves an approximate 80% improvement over the GRU model. Lastly, an ablation study on UPDeT with matched and mismatched observation-entity | action-group configurations, as shown in Fig. 3.4, reveals that disrupting the typical 'attack' action and enemy unit pairing results in severely diminished performance, even falling below that of the GRU model. This underscores the importance of combining policy decoupling with aligned observation-entity | action-group strategies for UPDeT to

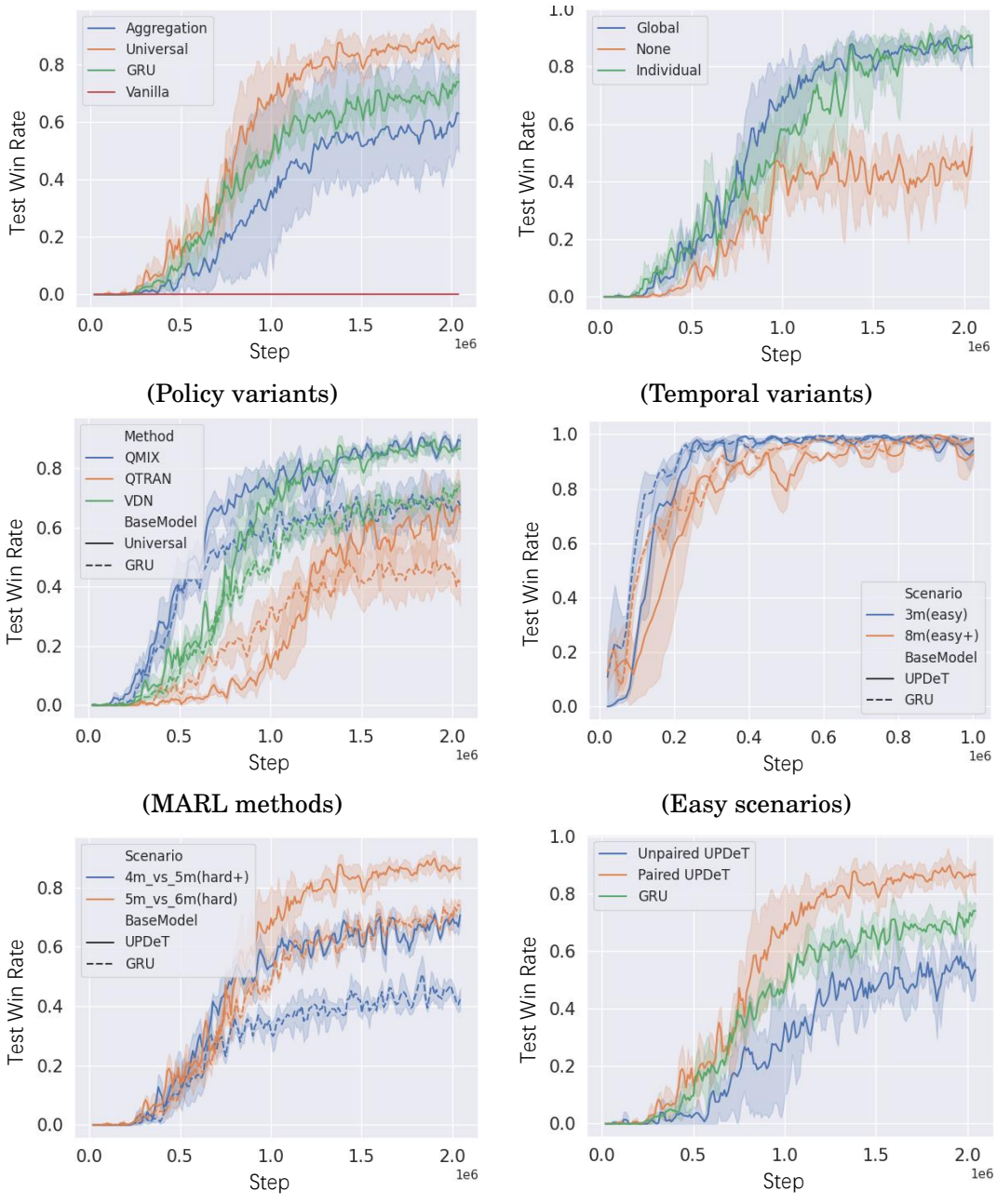
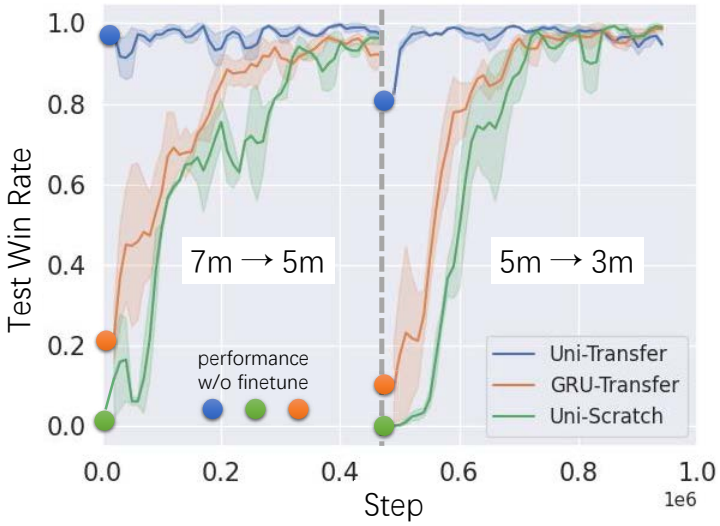
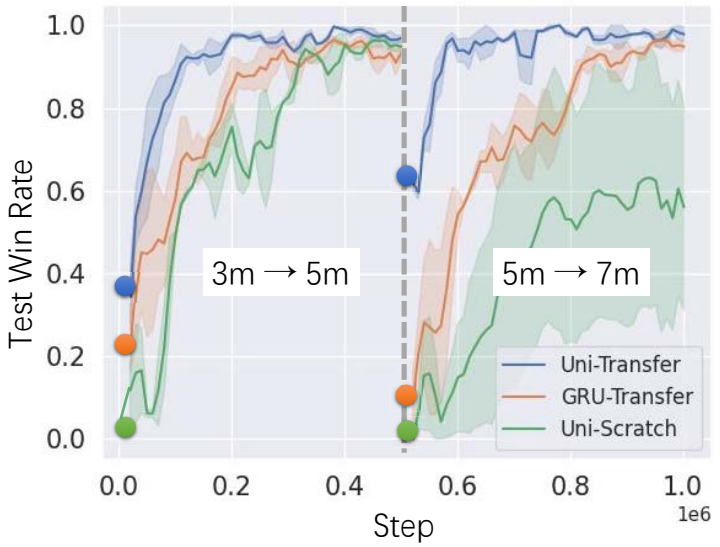


Figure 3.4: Experiment result with different task settings.

formulate effective policies.



(Transfer from 7 marines to 3 marines)



(Transfer from 3 marines to 7 marines)

Figure 3.5: Experimental outcomes for transfer learning using UPDeT (labeled as Uni-Transfer) and a GRU unit (denoted GRU-Transfer), alongside UPDeT training initiated from scratch (Uni-Scratch), are presented. The models are initially loaded from a source scenario at the 0 time step and subsequently fine-tuned on target scenarios at 500k time steps. A circle point in the results marks the performance of the models on new scenarios prior to any fine-tuning.

3.3.2 Multiple Scenarios

In this section, we explore UPDeT's transfer capabilities relative to RNN-based models. The models are initially trained in the 3m scenario and subsequently used to train further in the 5m and 7m scenarios, with a reverse training sequence from 7m to 3m. Despite

UPDeT's architecture remaining unchanged, the RNN model requires adjustments to the source model's architecture to accommodate the new scenario dimensions during training on target scenarios. The GRU cell parameters are retained, while the fully connected layer is reinitialized to suit the new scenario dimensions.

The final outcomes, shown in Figs. 3.5, demonstrate UPDeT's considerable superiority over the GRU model, requiring significantly fewer timesteps to converge compared to the GRU and training from scratch. Moreover, UPDeT exhibits robust generalization capabilities without the need for finetuning, suggesting that it acquires a durable policy with meta-level skills.

3.4 Conclusion

The introduction of UPDeT marked a notable shift from traditional RNN-based models to a more dynamic and scalable transformer-based approach. By leveraging the unique self-attention mechanism of transformers, UPDeT effectively addresses the challenges of partial observability and multi-agent coordination. This shift not only enhances the learning efficiency and decision-making accuracy of individual agents but also fosters better cooperative strategies among agents in decentralized settings.

Our experimental evaluations, conducted in the challenging micromanagement scenarios of StarCraft II, have demonstrated the superior performance of UPDeT compared to conventional models. UPDeT's ability to generalize across different scenarios without the need for extensive retraining underscores its potential for real-world applications where adaptability and robustness are critical. Moreover, the policy decoupling technique introduced by UPDeT has proven instrumental in aligning specific agent actions with corresponding environmental cues, thereby improving the interpretability and effectiveness of the learned policies.

The success of UPDeT in these complex scenarios provides promising directions for future research. The flexibility and scalability of the transformer architecture invite further exploration into other areas of reinforcement learning, such as dynamic task allocation and real-time strategy games. Additionally, this work sets a new benchmark for practical implementations of deep learning models in multi-agent environments. The continued refinement and adaptation of UPDeT will play a crucial role in harnessing the full potential of multi-agent systems across various domains.

CHAPTER 4

POLICY DIAGNOSIS VIA MEASURING ROLE DIVERSITY IN COOPERATIVE MULTI-AGENT RL

4.1 Introduction

Multi-Agent Reinforcement Learning (MARL) achievements have largely been empirically driven [5, 46, 88]. A fundamental challenge is the fair evaluation of various algorithms in MARL, as illustrated in Fig. 4.1a. Current research often focuses on algorithmic performance in specific tasks while neglecting their degradation in others [31, 76, 102, 109]. Additionally, advanced algorithms does not guarantee optimal results [24, 90, 102], largely due to the diverse attributes and objectives of agents in different scenarios. This indicates that a single algorithm may not be universally effective, necessitating tailored policy adjustments or training modifications.

For example, *Parameter Sharing* is a common strategy in MARL, where agents share model parameters [31, 76, 90]. While this can enhance policy optimization through a shared experience buffer, its effectiveness varies across different scenarios [15, 70, 94]. In some cases, selective or no parameter sharing significantly boosts performance over full parameter sharing. The underlying causes for these variations in effectiveness, however, are not well understood. This research explores how role diversity can influence the selection of parameter sharing strategies.

Communication is another critical aspect of MARL, enabling agents to share vital information to coordinate their actions effectively [38, 41, 49, 56, 89]. While sometimes

constrained [56, 79], communication often offers flexibility in how and when information is integrated [41, 86]. Our analysis delves into the relationship between role diversity and communication, showing how the former dictates the latter's necessity.

Credit Assignment also receives substantial attention in MARL. Techniques often integrate Q-learning or policy gradient methods with additional modules like value decomposition [31, 76, 90, 102] or shared critic functions [24, 56, 109] to optimize individual policies. While some argue that direct use of reward signals is unnecessary [70, 92], selecting the optimal credit assignment method, including independent learning (IL), is challenging due to varying performance across tasks. This work posits that role diversity significantly impacts the effectiveness of credit assignment strategies.

A well-conceived MARL policy should avoid any of these strategies that can lead to substantial performance setbacks, referred to as the barrel effect. This motivates the need for a metric to evaluate the characteristics of various MARL tasks. We introduce **Role Diversity** as such a metric, quantifying behavioral differences among agents via action-based, trajectory-based, and contribution-based dimensions, collectively termed as role distance. Our theoretical and experimental analyses confirm that distinct types of role diversity distinctly affect algorithmic, approximation, and statistical errors across the main MARL training strategies-underscoring the utility of role diversity.

Role diversity provides guidelines for identifying policy weaknesses and assessing whether a superior training strategy based on role diversity is feasible, thereby enabling more accurate algorithm performance comparisons across different MARL tasks.

4.2 Role Diversity

This section elucidates how role diversity is defined and measured within a multi-agent task, highlighting its utility in evaluating existing policies and determining the necessity for policy revisions.

Role diversity serves as a key differentiator of agent characteristics in MARL, as evidenced by several studies [15, 50, 101, 102]. Traditional definitions of an agent's role-such as roles perceived as high-level options within a hierarchical RL framework [102], or based on environmental impact similarities under a random policy [15]-fall short in capturing the full spectrum of behavioral differences among agents. Our approach aims to comprehensively delineate roles across three dimensions: *action-based*, *trajectory-based*, and *contribution-based*. This refined definition facilitates the measurement of role distance between agents and employs this metric to quantify role diversity for each

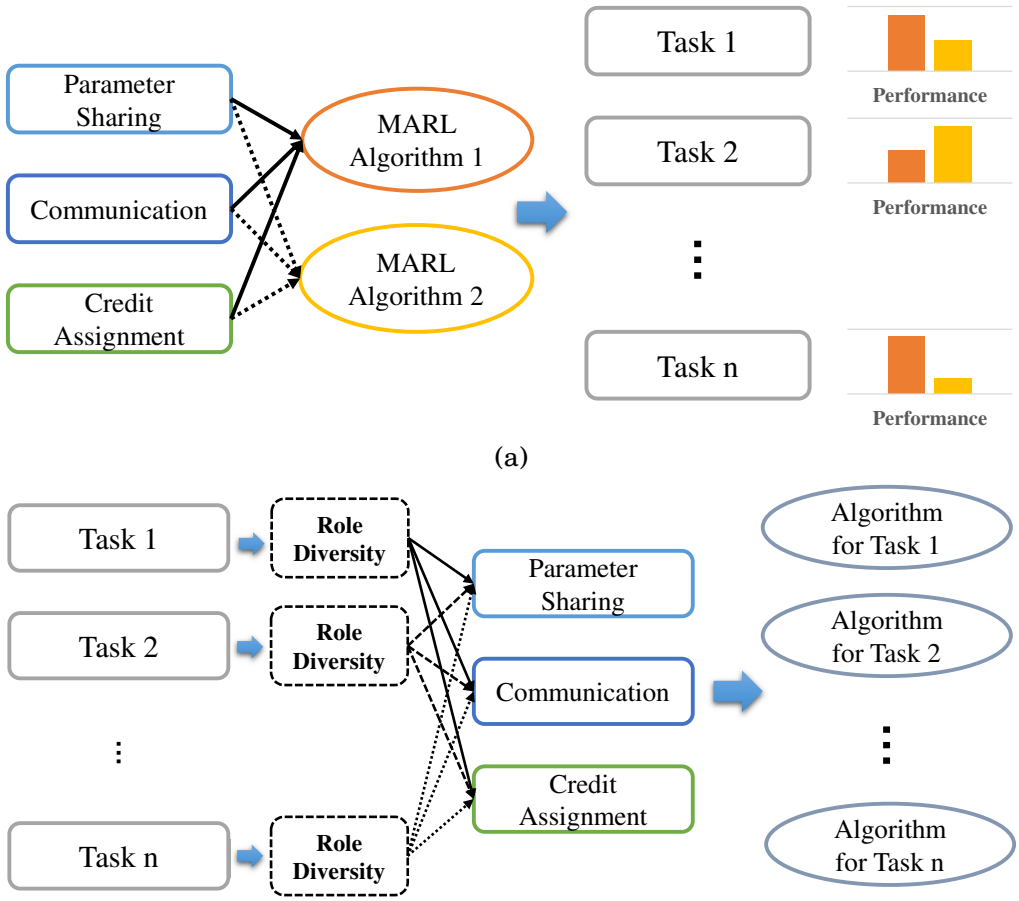


Figure 4.1: (a) Algorithms 1 and 2 serve as illustrative examples demonstrating the assembly of MARL policies via diverse training methodologies. These methodologies encompass three distinct aspects of MARL: parameter sharing, communication, and credit allocation. It is important to note that the effectiveness of Algorithms 1 and 2 varies across tasks from 1 to n. (b) Employing Role Diversity as a metric for each task facilitates the identification of any inappropriate applications of specific training strategies in the existing policy. This evaluation aids in selecting an optimal mix of training strategies, thereby enhancing overall policy performance.

MARL task, establishing a robust link between role diversity and optimization efficacy in MARL.

4.2.1 Action-based Role

In MARL, distinct actions performed by agents, given their specific states, inherently define their roles. Although actions at a single timestep might suggest different roles

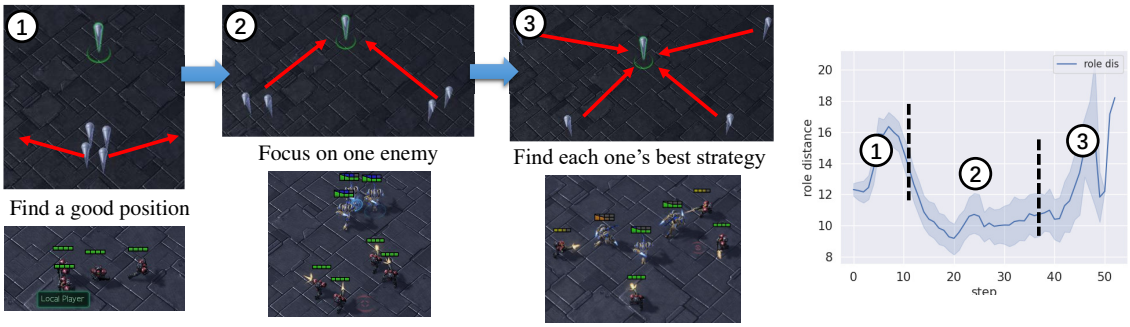


Figure 4.2: Illustration of action-based role diversity within a single episode.

[102], consistent behavior across a series of actions is a more accurate indicator of role differentiation. For instance, two soccer players passing the ball might perform differing actions at each timestep, yet exhibit similar roles over the course of play, as shown in studies like [46]. Hence, we focus on a sequence of actions to define the action-based role, utilizing frequency statistics of actions over an interval encompassing h steps before and after a specific timestep. The action-based role difference is mathematically defined as:

$$r_t^i = \frac{1}{2h+1} \sum_{t-h}^{t+h} \pi^i, \tag{4.1}$$

where t denotes the current timestep, h the interval, and i the agent index. The distance between the action-based roles of two agents, i and j, is quantified using the symmetrical Kullback-Leibler divergence:

$$d_{(act)t}^{i,j} = KL(r_t^i \| r_t^j) + KL(r_t^j \| r_t^i), \tag{4.2}$$

where $d_{(act)t}^{i,j}$ represents the action-based role distance at timestep t, and KL denotes the *Kullback-Leibler* divergence.

4.2.2 Trajectory-Based Role

Trajectory-based role diversity extends beyond mere action variance to include agents' movement paths, which can reveal substantial differences in roles over time. In a scenario such as a soccer match, where players frequently exchange passes, their actions might appear similar, yet their trajectories-paths taken over the field-highlight their distinct roles. This differentiation is crucial in many cooperative MARL settings, particularly under partial observation conditions [79, 88, 113], where agents' limited vision scopes reduce commonality in perceived information. The trajectory-based role is therefore assessed by comparing agents' movement paths.

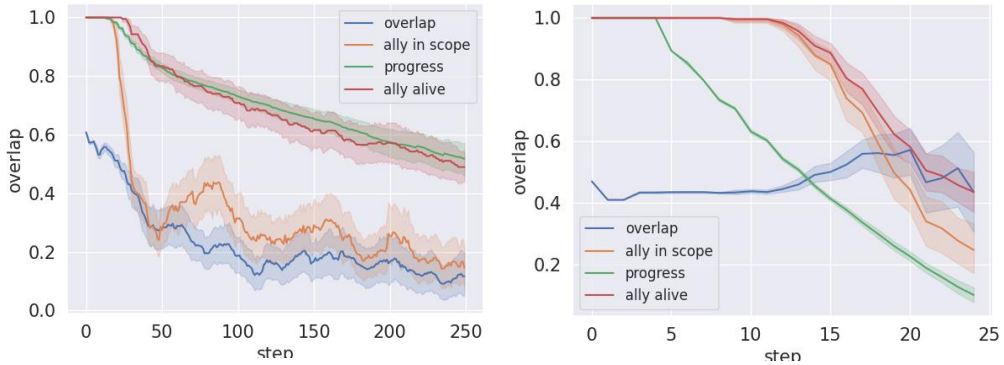


Figure 4.3: Trajectory-based role diversity observed in one episode.

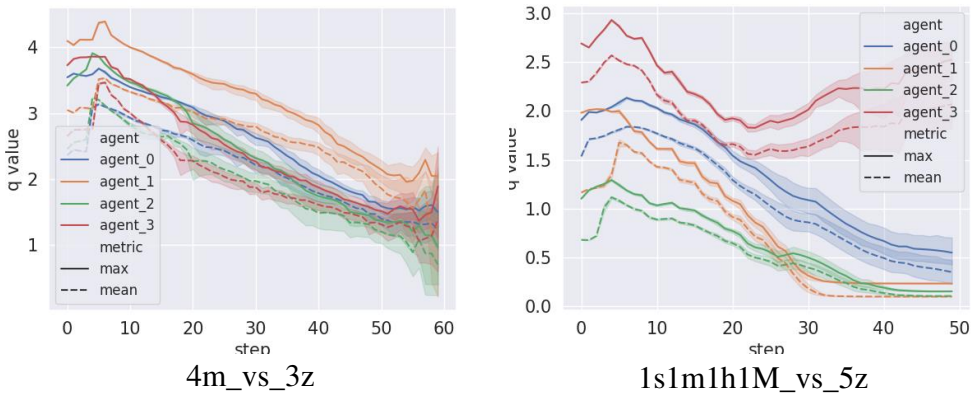


Figure 4.4: Contribution-based role diversity in one episode.

Trajectory differences between two agents i and j at timestep t, denoted as $d^{i,j}_{(traj)t}$, are measured using the observation overlap percentage, a direct metric reflecting the proportion of shared space in their movements. This method provides a straightforward, scalable measure of role diversity based on agents' trajectories, as demonstrated in Figure 4.3.

To explore the practical applications of these metrics, we extend our analysis to real-world scenarios, incorporating both controlled game environments and realistic settings where agents operate based on actual imagery and semantic data. This holistic approach not only underscores the theoretical constructs of our role diversity metrics but also highlights their versatility and adaptability in varying MARL contexts.

4.2.3 Contribution-Based Role

In modern MARL environments, agents often start with diverse attributes to create a varied and realistic multi-agent system [46, 56, 79, 88]. For instance, in simulations like those in [46], roles such as forward and goalkeeper have significantly different observation ranges, action capabilities, and reward structures. While these differences are apparent, quantifying the distance between such roles presents a challenge.

To tackle this, we introduce the concept of *contribution*, reflecting each agent's differential impact within a cooperative setup. In MARL, the goal is to develop an optimal joint policy from the individual policies of N agents, making the quantification of each agent's contribution towards the collective outcome crucial. Techniques in [24, 76, 90] optimize individual policies effectively by using role-specific reward signals. We measure an agent's contribution using their Q value or state value, indicative of their role in the team's success. The contribution-based role diversity between any two agents is calculated as follows:

$$d^{i,j}_{(cont)t} = \frac{|v^t_i - v^t_j|}{\max(v_i, v_j)} \quad \forall i, j \in N, \tag{4.3}$$

where v represents the Q or state value of an agent, and $|v^t_i - v^t_j|$ is the absolute difference in values between two agents at timestep t, normalized against the maximum value difference among all agents, thus maintaining the diversity scale from 0 to 1.

4.2.4 Distance to Diversity

The overall role diversity within a multi-agent system (MAS) is quantified by averaging the role distances among different agent pairs, covering various aspects of roles. For a system with N agents, this overall diversity is computed as:

$$D_t = \frac{1}{N(N-1)/2} \sum_{i=0}^{N-1} \sum_{j=i+1}^{N} d^{i,j}_t, \tag{4.4}$$

where $d^{i,j}_t$ could represent any of the role distances, such as action-based $d^{i,j}_{(act)t}$, trajectory-based $d^{i,j}_{(traj)T}$, or contribution-based $d^{i,j}_{(cont)T}$. This formula provides a comprehensive measure of action-based, trajectory-based, and contribution-based role diversity within the team.

4.3 Policy Diagnosis

This section leverages role diversity metrics to assess the efficacy of selected training strategies within a MARL framework. We present case studies demonstrating how variations in role diversity types impact policy outcomes during an episode and highlight deficiencies in current training methodologies. We examine three prevalent training strategies in MARL: parameter sharing, communication, and credit assignment.

For action-based role diversity, an illustration is provided in Fig. 4.2. We analyze a battle scenario from SMAC, designated 4m_vs_3z, identifying three critical phases: *Securing Optimal Positions*, *Concentrating on a Common Adversary*, and *Developing Individual Tactics*. Initially, agents seek strategically advantageous positions, enhancing role diversity. In the subsequent phase, as agents focus on a common target, their strategies converge, reducing role diversity. The final phase sees agents adapting individually to enemy maneuvers, leading to increased role diversity. The decision to share or segregate model parameters greatly affects policy optimization, with high role diversity favoring individual policies in the first and last phases, while the middle phase suggests that a shared policy could be more effective.

The example of trajectory-based role diversity is detailed through observation overlap percentage curves in Fig. 4.3 for scenarios 3s_vs_5z and 4m_vs_5m. As the gameplay progresses, the noticeable divergence in these curves indicates differing strategic approaches. In 3s_vs_5z, pronounced trajectory-based role diversity suggests a need for individual observations and limited agent communication, enhancing training outcomes. Conversely, in 4m_vs_5m, lower trajectory-based diversity supports shared observations, facilitating learning.

Regarding contribution-based role diversity, as depicted in Fig. 4.4, there is significant variance in the Q value curves across scenarios. In 4m_vs_3z, contribution-based role diversity is relatively subdued compared to 1s1m1h1M_vs_5z, suggesting that a learnable credit assignment module is advantageous in the former but detrimental in the latter. This emphasizes the need to tailor credit assignment strategies according to the specific role diversities encountered in different scenarios.

4.4 Experiments

In this section, we investigate how role diversity influences model performance and informs adjustments in training strategies within cooperative MARL settings. Our key

findings are: 1. Parameter sharing strategies' performance strongly correlates with Action-Based Role diversity (Sec. 4.4.1); 2. The effectiveness of communication mechanisms is linked to Trajectory-Based Role diversity (Sec. 4.4.2); 3. The success of credit assignment methods depends on Contribution-Based Role diversity (Sec. 4.4.3); 4. Training strategy decisions should align with the observed role diversity in different scenarios. We primarily utilize the Multi-Agent Particle Environment (MPE) [56] and StarCraft Multi-Agent Challenge (SMAC) [79] as experimental platforms, adapting them to suit specific parameter sharing and communication needs. All experiments are conducted using eight random seeds, with role diversity metrics derived from our baseline policy, VDN [56], without parameter sharing or communication to ensure training robustness and efficiency.

4.4.1 Parameter Sharing

Action-based role diversity significantly influences the speed and efficacy of parameter sharing strategies in cooperative MARL. We select diverse scenarios from MPE and SMAC benchmarks to capture a range of action-based role diversities. The performance of various parameter sharing configurations and their corresponding model performance curves are depicted in Table 4.1, Fig. 4.5, and Fig. 4.6. For SMAC, we employ two metrics to quantify r_t^u in Eq. 4.1: *real action diversity* and *semantic action diversity*, with the latter categorizing actions into semantic groups such as *move & attack*. MPE scenarios are limited to movement actions, thus do not feature semantic action diversity. Action-based role diversity is calculated per Eq. 4.2. Our baseline MARL credit assignment strategy, as detailed in Table 4.1, utilizes VDN [90] combined with full, partial, or no parameter sharing. We also evaluate other popular credit assignment methods like IQL [92], IA2C, MADDPG [56], MAPPO [109], MAA2C, and QMIX [76] under various parameter sharing settings in Figs. 4.6 and 4.5. IA2C and MAA2C extend A2C [61] to multi-agent contexts.

Table 4.1 shows the performance of three training strategies: *No Shared*, *Partly Shared*, and *Fully Shared* using the VDN method. As action-based role diversity increases, the *Shared* strategy experiences a decline in convergence speed (measured in half the training steps) and in final reward (full training steps). Notably, scenarios with identical agent types (e.g., 3s_vs_5z, 4m_vs_4z) do not necessarily correlate with minimal action-based role diversity, and vice versa (e.g., 1s1m1h1M_vs_3z), indicating the importance of role identification prior to establishing an effective policy. Figs. 4.5 and 4.6 illustrate model performances for two parameter sharing strategies: *No Shared* and

Shared, across different credit assignment methods. For policy gradient-based methods, we extend training from the standard 2M steps to 20M steps (a tenfold increase) due to the slower convergence rates of these methods (e.g., MAPPO, MAA2C) compared to Q-value-based approaches. Our results indicate that while different credit assignment methods have a marginal impact on parameter sharing strategies, a consistent trend emerges where non-parameter sharing strategies outperform as action-based role diversity increases.

In conclusion, scenarios with significant action-based role diversity typically favor a non-parameter sharing strategy to enhance convergence speed and peak performance, whereas scenarios with minimal diversity benefit more from shared parameters.

Table 4.1: Evaluation of three parameter sharing approaches across various scenarios. The term *Warm-up* denotes the point at which the reward values begin to diverge among the strategies. The symbol + indicates the additional reward obtained from the baseline established during the warm-up phase. The figures to the left and right of the / indicate the rewards obtained at halfway through the training and upon completion of the training, respectively. The optimal performance for each scenario is highlighted in bold red. The Role Diversity column is shaded in gradient grey, with darker shades indicating greater diversity in roles. A comprehensive analysis is provided in Sec. 4.4.1.

Benchmark	Scenario	Role Diversity	Warm-up	No shared	Partly shared	Shared
MPE	SimpleSpread	14.1	-598.3	+137.0 / +142.9	+149.0 / +176.4	**+154.1 / +198.0**
	Tag	17.8	3.8	+43.4 / +57.3	**+47.0 / +60.9**	+48.8 / +59.2
	Adversary	18.3	10.7	+5.2 / +5.7	**+6.2 / +6.6**	+5.4 / +5.9
	DoubleSpread-2	17.6	7.3	**+47.8 / +53.2**	+28.6 / +34.6	+3.6 / +15.9
	DoubleSpread-4	19.5	22.0	**+29.5 / +192.4**	+12.0 / +91.3	+11.4 / +5.3
SMAC	2m	3.1 / 12.2	6.0	+9.2 / +11.1	+15.5 / +15.6	**+18.1 / +17.6**
	4m_vs_4z	3.3 / 19.3	4.4	+ 8.8 / +12.7	**+ 10.5 / +14.7**	+5.4 / +8.4
	4m_vs_3z	3.8 / 12.1	7.2	+12.4 / +12.1	**+12.5 / +12.5**	+11.9 / +12.3
	1c1s1z_vs_1c1s3z	8.7 / 22.0	11.8	**+4.1 / +6.1**	+3.7 / +5.9	+2.7 / +5.4
	1s1m1h1M_vs_5z	6.2 / 22.5	6.2	**+6.4 / +9.1**	+4.2 / +8.5	+3.7 / +6.1

4.4.2 Communication

In fields like computer vision and natural language processing, it is generally accepted that more information leads to better model optimization, due to larger datasets with more detailed and accurate annotations enhancing model training. However, in reinforcement learning, where data sampling depends on a potentially randomly initialized policy, excessive information might complicate policy optimization and degrade data quality, creating a counterproductive cycle. It is crucial to determine when and how additional information through communication mechanisms should be integrated in

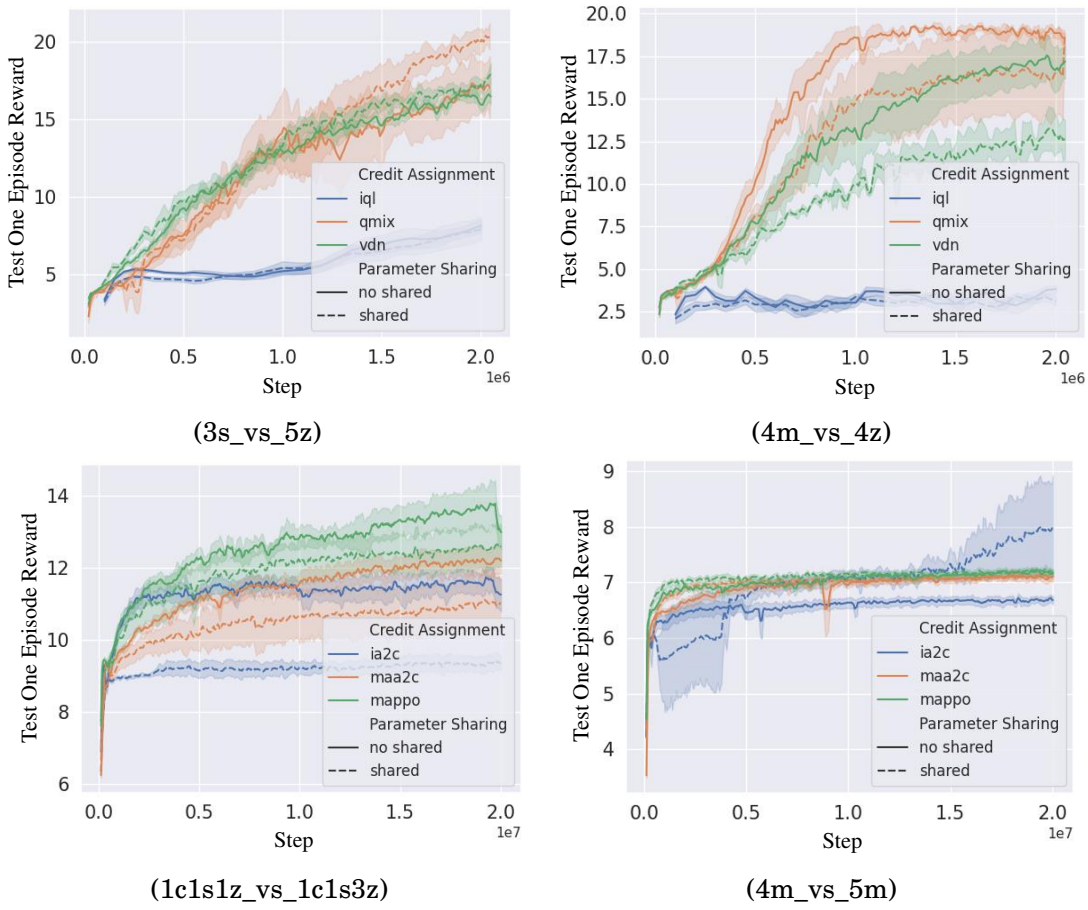


Figure 4.5: Performance trends in Q-value based credit allocation using *Shared* and *No Shared* parameter sharing approaches.

cooperative MARL. The necessity for this extra information heavily depends on the similarity of support sets used for policy optimization among agents, largely influenced by their trajectory-based role diversity as defined in Sec. 4.2.2. Scenarios with minimal trajectory-based role diversity indicate uniform support set patterns, suggesting that unified input through communication could benefit policy optimization. Conversely, our experiments show that scenarios with significant observation overlaps, indicative of substantial trajectory-based role diversity, better accommodate communication.

Additional experiments to explore how the pattern of input observations (support sets) affects model performance when altering the scope of vision are detailed in Table 4.2. These results confirm that model performance strongly correlates with the scope of vision and depends on trajectory-based role diversity. Scenarios with low trajectory-based role diversity are conducive to a broader vision scope, indicating that similar support set patterns enhance policy optimization. In contrast, scenarios with high trajectory-

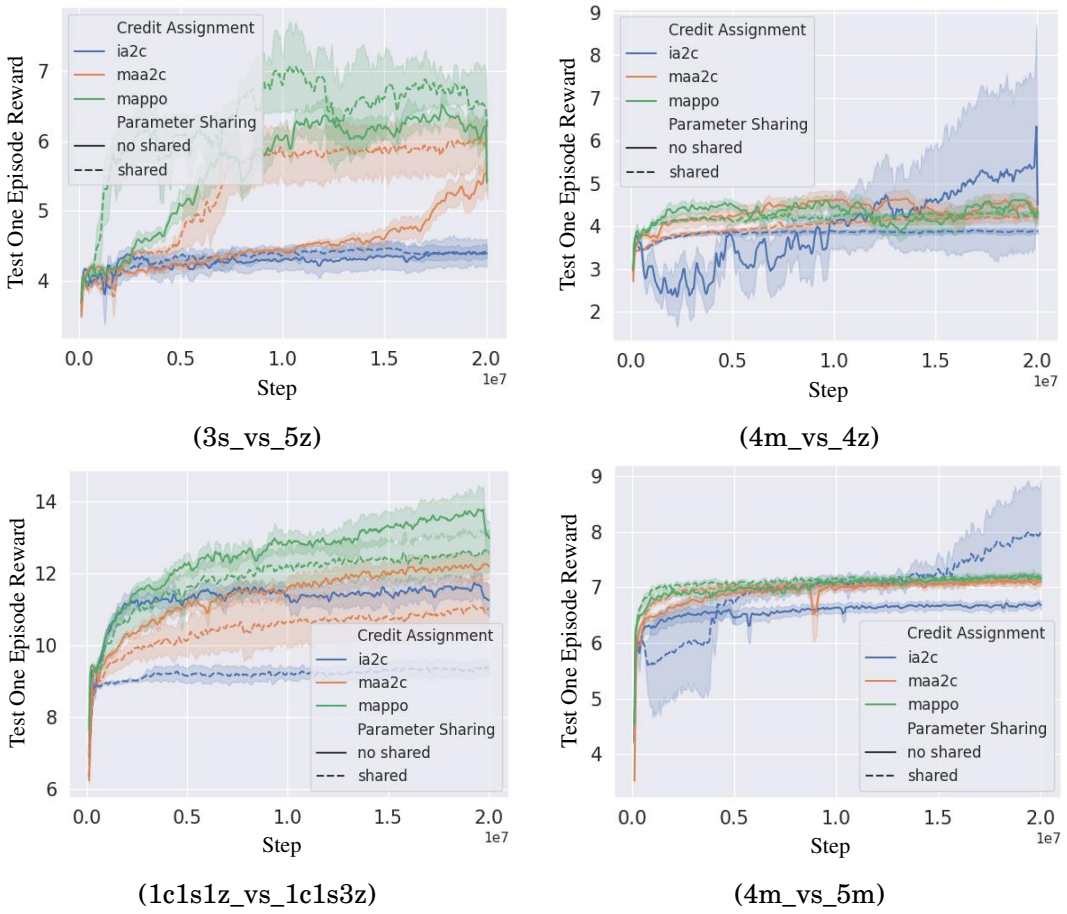


Figure 4.6: Performance curves on policy gradient-based(row three and four) credit assignment.

based role diversity benefit from a reduced vision scope, promoting diverse pattern inputs advantageous for policy optimization. This supports the notion that in cases with minimal trajectory-based role diversity, additional information provided through communication can help form a consistent pattern of support sets, generally preferable.

4.4.3 Credit Assignment

The efficacy of various credit assignment methods in MARL is closely linked to the contribution-based role diversity within scenarios. We focus on three main Q value-based MARL algorithms: VDN [90], QMIX [76], and IQL [92], evaluating their performance across different setups with varying levels of contribution-based role diversity, as measured by the Q values (Eq. 4.3). In scenarios with low Q diversity (approximately 0.5), QMIX tends to outperform VDN under both shared and non-shared parameter strategies.

Table 4.2: The impact of varying vision scopes (6-9-18) on model performance is evident. The smallest scope, denoted by *6*, corresponds to the agents' attack range. Optimal performance is highlighted in red. Columns representing vision scope feature a gradient of red, with deeper shades indicating superior policy performance. A comprehensive analysis is provided in Sec. 4.4.2.

scenario	obs overlap	scope	performance	scenario	obs overlap	scope	performance
1s1m1h1M_vs_3z	0.41	6	15.6 / 19.5 / 19.5	4m_vs_5m	0.47	6	6.3 / 9.2 / 10.4
		9	16.4 / 19.5 /19.7			9	6.5 / 10.1 / 10.9
		18	16.1 / 19.6 / 19.9			18	6.8 / 10.9 / 11.1
1s1m1h1M_vs_4z	0.25	6	8.4 / 15.3 / 18.8	1c1s1z_vs_1c1s3z	0.40	6	11.5 / 15.1 / 17.6
		9	8.4 / 15.7 / 19.7			9	12.3 / 16.0 / 17.8
		18	7.8 / 11.8 / 15.9			18	12.4 / 15.3 / 17.6
1s1m1h1M_vs_5z	0.18	6	6.6 / 14.2 / 17.7	3s_vs_5z	0.21	6	6.0 / 15.1 / 17.5
		9	6.3 / 12.6 / 15.3			9	5.4 / 12.9 / 16.4
		18	5.9 / 8.9 / 10.4			18	5.2 / 9.0 / 12.1

Table 4.3: The influence of action-based role diversity on the effectiveness of various parameter sharing strategies is observed in the MPE [56] and SMAC [79] benchmarks. Optimal performance in each case is indicated by a ✓in a red-colored cell. Algorithms marked with asterisks are statistically comparable in performance.

scenario	Q diversity	no shared			shared		
		vdn	qmix	iql	vdn	qmix	iql
1c1s1z_vs_1c1s3z	<0.1	12.3 / 15.9 / 17.9	12.9 / 17.8 / 19.4 ✓	10.8 / 12.3 / 12.2	11.2 / 14.5 / 17.2	12.5 / 15.8 / 18.4 ✓	9.8 / 11.2 / 11.9
3s_vs_5z		5.4 / 12.9 / 16.4	4.6 / 13.5 / 17.0 ✓	4.6 / 5.1 / 7.9	6.0 / 13.6 / 17.2	4.2 / 12.9 / 20.0 ✓	4.3 / 5.3 / 7.8
4m_vs_4z		4.3 / 13.2 / 17.1	4.3 / 18.3 / 18.8 ✓	3.3 / 3.2 / 3.7	4.6 / 9.8 / 12.8	4.3 / 14.8 / 16.5 ✓	2.6 / 3.2 / 3.2
4m_vs_5m	0.1-0.5	6.5 / 10.1 / 10.9 *	7.0 / 9.9 / 10.9 *	4.8 / 7.6 / 8.1	6.8 / 11.9 / 12.6 *	6.9 / 12.4 / 13.3 *	5.1 / 8.1 / 8.5
4m_vs_3z		7.5 / 19.6 / 19.3 *	6.5 / 19.7 / 19.3 *	4.5 / 5.7 / 11.1	6.3 / 19.1 / 19.5 *	6.1 / 19.7 / 19.7 *	4.2 / 4.5 / 5.7
1s1m1h1M_vs_3z	>0.5	16.4 / 19.6 / 19.6 ✓	6.5 / 7.5 / 7.8	11.1 / 16.9 / 19.2	16.1 / 19.6 / 19.8 ✓	9.9 / 9.8 / 8.9	12.2 / 17.9 / 19.6
1s1m1h1M_vs_4z		8.4 / 16.0 / 19.8 ✓	4.9 / 5.1 / 6.1	7.4 / 9.0 / 10.7	8.1 / 13.5 / 18.2 ✓	5.5 / 5.0 / 5.0	7.1 / 8.5 / 8.5
1s1m1h1M_vs_5z		6.3 / 12.6 / 15.3 ✓	4.2 / 4.2 / 3.6	5.5 / 6.1 / 6.5	6.2 / 9.9 / 12.3 ✓	4.0 / 2.5 / 4.2	5.4 / 6.3 / 6.3

However, as Q diversity increases, QMIX's performance begins to decline. For instance, in scenarios with markedly diverse Q value distributions, such as 1s1m1h1M_vs_3/4/5z, VDN significantly surpasses QMIX. Meanwhile, IQL, although generally underperforming relative to VDN and QMIX, shows robustness in scenarios with minimal Q diversity and performs adequately in simpler settings like 1s1m1h1M_vs_3z.

The findings suggest that QMIX, which incorporates a learnable neural network for value decomposition, may not be ideal in scenarios with substantial contribution-based role diversity due to the network's inability to effectively minimize approximation errors for Q_{tot} under diverse reward conditions. This complexity adds unnecessary burden, particularly when the contribution to the overall reward varies significantly among agents. Unlike QMIX, IQL treats each Q_{tot} as an individual Q value and does not suffer from these drawbacks. Hence, in environments with high contribution-based role diversity, it is advisable to avoid credit assignment methods that employ learnable value

decomposition modules, opting instead for strategies that allow for more straightforward and direct credit assignment. This approach can prevent potential performance issues caused by complex value decompositions in diverse agent settings.

4.5 Conclusion

We have established role and role diversity as fundamental metrics for evaluating and diagnosing policies in cooperative multi-agent tasks. Through both theoretical analysis and practical experiments on established MARL benchmarks, we have demonstrated a pronounced correlation between role diversity and model performance.

Our comprehensive analysis suggests that role diversity is an invaluable metric for identifying policy weaknesses and selecting appropriate training strategies within a cooperative MARL framework. Specifically, we advocate for strategic adjustments based on role diversity measurements:

- **Action-Based:** High diversity indicates the suitability of a non-parameter sharing strategy to allow for greater individual agent flexibility, while low diversity suggests that parameter sharing could enhance overall efficiency.
- **Trajectory-Based:** High diversity implies that limiting communication could prevent complications arising from conflicting agent trajectories, whereas low diversity supports the use of enhanced communication to streamline collective agent actions.
- **Contribution-Based:** High diversity favors the use of independent learning or fixed credit assignment methods to cater to the varied contributions of individual agents, while low diversity benefits from integrated strategies that consolidate these contributions.

Furthermore, the utility of role diversity extends beyond fully optimized policies: Role diversity metrics can provide critical insights into policy effectiveness even before policies are fully trained. Preliminary analysis at early training stages (e.g., after 100k timesteps) can reveal significant differences in role diversities, guiding early intervention and strategic adjustments. Additionally, when policy diagnosis indicates a preferable strategy, it is not always necessary to restart the training process from scratch. For instance, if a switch from a parameter-sharing to a non-sharing strategy is advised, existing models can be redistributed among agents and fine-tuned individually, thus preserving prior learning while aligning with new strategic directions.

CHAPTER 5

ONE PLATFORM FOR ALL: A UNIFIED MULTI-AGENT RL LIBRARY

5.1 Introduction

While single-agent RL has achieved substantial integration of both algorithms (e.g., SpinningUp [2], Tianshou [107], RLlib [54], Dopamine [13], and the Stable-Baselines series [21, 29, 74]) and environments (e.g., Gym [10], Gymnasium [27]), MARL encounters unique challenges in establishing a unified, high-quality library system. The main challenge stems from the diverse range of MARL algorithmic pipelines, which vary in their objectives-some promote cooperation among agents, while others encourage competition to maximize individual gains at the expense of others. These algorithms also differ in their approaches to parameter sharing; for example, HATRPO [45] requires independent parameters, whereas MAPPO [109] benefits from shared parameters. Furthermore, the application of centralized information varies; some algorithms integrate value functions (e.g., VDN [87]), and others centralize them entirely (e.g., MADDPG [56]). Although platforms like EPyMARL [70] attempt to harmonize MARL algorithms by categorizing them into independent learning, centralized critic, and value decomposition frameworks, they fall short in addressing the full spectrum of these challenges.

Moreover, the diversity of interfaces in multi-agent environments, tailored to specific tasks (e.g., asynchronous interactions in Hanabi, action masks in SMAC [79], and the

blend of local observation with global state in MAgent [113]), leads to significant inconsistencies. These inconsistencies hinder the creation of a uniform agent-environment interaction framework, complicating the alignment between algorithm implementations and task environments. Consequently, an algorithm developed for one environment may not seamlessly transition to another due to these interface discrepancies. While platforms like PettingZoo [93] offer a diverse set of multi-agent tasks, they do not adequately support CTDE-based algorithms, as they lack explicit provisions for critical data like global state and action masks. Alternative approaches, such as the MAPPO benchmark [109], which equips each environment with a distinct runner script, introduce their own challenges, including increased maintenance complexity and difficulties in extending to new tasks.

5.2 Existing Libraries

Creating a unified platform for MARL research is valuable yet challenging. The field has evolved from simple, single-task libraries to more sophisticated tools and APIs capable of handling a variety of tasks and advanced algorithms.

PyMARL [79], the first and most recognized MARL library, was originally developed for the SMAC [79] environment and focuses on team-based cooperative learning to achieve higher team rewards. However, PyMARL has not been updated recently and fails to incorporate recent advancements in the field. Extensions such as PyMARL2 [32] and EPyMARL [70] have been developed to address these limitations.

PyMARL2 [32] enhances the original by introducing a fine-tuned version of QMIX [76], providing state-of-the-art performance on SMAC. It expands the library to include ten algorithms, integrating numerous code-level enhancements.

EPyMARL [70] aims to provide a comprehensive framework for unifying cooperative MARL algorithms. It categorizes algorithms into independent learning, value decomposition, and centralized critics, although it remains limited to cooperative contexts. EPyMARL implements nine algorithms and has added three more cooperative environments for broader algorithm testing.

While these libraries build on the centralized training decentralized execution (CTDE) paradigm set by PyMARL, other MARL libraries have been developed with different focuses.

MARL-Algorithms [58] covers a broader array of topics including improved credit assignment, communication-based learning, graph-based learning, and multi-task cur-

Table 5.1: A comparison between current MARL libraries and our MARLlib. (x) stands for the number of available algorithms. * denotes that the benchmark has a unique framework of its own.

Library	Task Mode	Supported Env	Algorithm	Parameter Sharing	Async Sampling	Framework
PyMARL [73]	cooperative	1	Independent Learning (1) Centralized Critic (1) Value Decomposition (3)	full-sharing		*
PyMARL2 [32]	cooperative	1	Independent Learning (1) Centralized Critic (1) Value Decomposition (9)	full-sharing		PyMARL
MARL-Algorithms [58]	cooperative	1	CTDE (6) Communication (1) Graph (1) Multi-task (1)	full-sharing		*
EPyMARL [70]	cooperative	4	Independent Learning (3) Centralized Critic (4) Value Decomposition (2)	full-sharing non-sharing		PyMARL
MAlib [117]	self-play	2 + PettingZoo [93] OpenSpiel [47]	Population-based (9)	full-sharing group-sharing non-sharing	✓	*
MAPPO benchmark [109]	cooperative	4	Multi-agent PPO (1)	full-sharing non-sharing	✓	pytorch-a2c-ppo-acktr-gail [44]
MARLlib	cooperative collaborative competitive mixed	10 + PettingZoo	Independent Learning (6) Centralized Critic (7) Value Decomposition (5)	full-sharing group-sharing non-sharing	✓	Ray [64]/RLlib [54]

riculum learning, featuring nine algorithms but primarily tested on SMAC.

MAPPO benchmark [109], the official repository for MAPPO, focuses on cooperative MARL across four environments, aiming to establish a robust baseline with its singular focus on MAPPO.

MAlib [117] introduces a newer library that integrates game theory with MARL for population-based solutions across a spectrum of multi-agent tasks.

Although these existing libraries provide solid platforms for MARL research, they exhibit significant limitations, notably in task coverage and environment diversity, as highlighted in Table 5.1. Furthermore, these libraries often neglect the organizational structure of algorithms, resulting in limited extensibility and cumbersome codebases.

5.3 MARLlib: A Scalable MARL Library

To address the inherent challenges of MARL, we introduce **MARLlib**, a comprehensive library built on top of Ray [64] and RLlib [54]. MARLlib leverages the core strengths of RLlib and introduces four innovative features, positioning it as a pivotal resource for the MARL research community.

1. Unified algorithm pipeline with a novel agent-level distributed dataflow: MARLlib is architected around the principle that all MARL paradigms can effectively be decomposed into a series of single-agent processes, where each agent independently manages its dataflow and policy optimization. This framework supports a wide array of task types-cooperative, collaborative, competitive, and mixed-within a single algorithmic framework. Algorithms are categorized based on their use of centralized information, enhancing modularity and extensibility. This design is illustrated in Figure 5.1, where MARLlib successfully integrates multiple algorithms using the proposed distributed dataflow architecture.

2. Unified multi-agent environment interface: MARLlib introduces a standardized interface that follows the Gym convention, designed to be compatible with a broad range of existing multi-agent environments. This interface supports asynchronous interactions and provides all necessary data for various algorithms, effectively decoupling the algorithms from the environment specifics. MARLlib is compatible with ten diverse environments, such as SMAC [79], MAMuJoCo [71], and others, each selected for their unique characteristics and the variety of MARL challenges they present.

3. Effective policy mapping: To accommodate different task requirements, MARLlib offers flexible parameter sharing options-full-sharing, non-sharing, and group-sharing-through an easy-to-use policy mapping API, originally developed in RLlib. These settings can be adjusted via a configuration file, allowing users to experiment with different strategies effortlessly. This feature is crucial for adapting the same algorithm to various scenarios without modifying the underlying codebase.

4. Exhaustive performance evaluation: We conducted extensive testing across 23 different scenarios from five distinct environment suites, averaging results over four random seeds. This rigorous testing, involving over a thousand individual experiments, not only validates the functionality of MARLlib but also provides a reliable benchmark for the community. The results are accompanied by detailed hyper-parameter documentation to ensure reproducibility and facilitate further analysis, discussed in Section 5.4.

Through these features, MARLlib establishes itself as a versatile platform for developing, training, and evaluating MARL algorithms, promising to drive both theoretical and practical advancements in the field.

5.3.1 Agent-level Dataflow

Centralized Training with Decentralized Execution (CTDE) is a widely recognized approach for addressing multi-agent challenges, where agents independently execute and

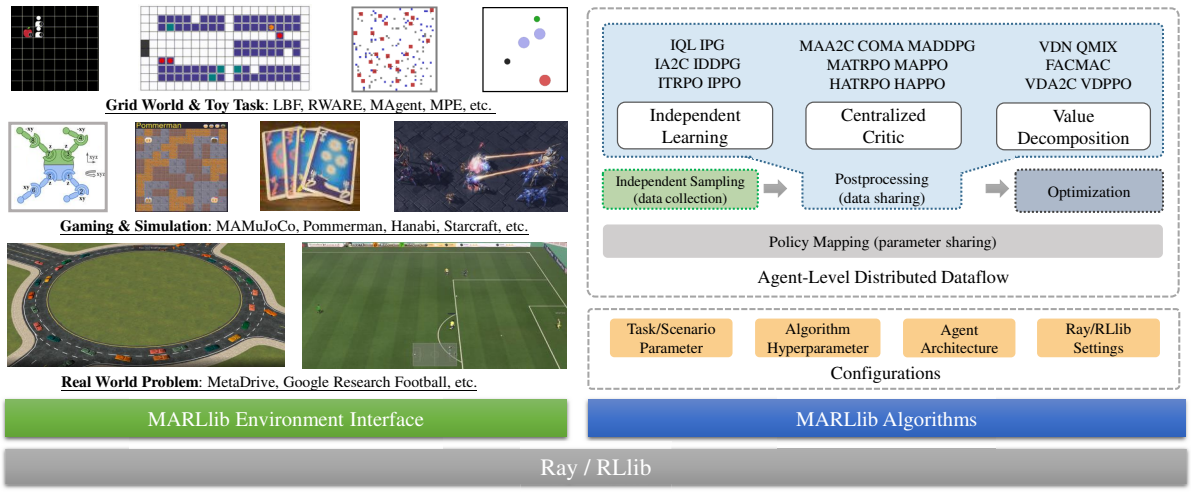


Figure 5.1: An overview of Multi-Agent RLlib (MARLlib). MARLlib unifies environment interfaces to decouple environments and algorithms. Beyond, it unifies independent learning, centralized critic, and value decomposition algorithms with an agent-level distributed dataflow, and allows flexible parameter sharing by means of policy mapping. The whole pipeline can be fully determined by configuration files. To our best knowledge, with the widest coverage of algorithms and environments, MARLlib is one of the most comprehensive MARL research platform.

optimize their policies while utilizing central data during training to align their updates. In this paradigm, the learning process is generally divided into two phases: data collection and model refinement. During model refinement, all data acquired in the collection phase are accessible, facilitating centralized training. Nonetheless, this integration of data selection and model optimization into a single phase complicates adapting the algorithm for various operational modes such as cooperative and competitive scenarios, necessitating a complete redesign of the learning workflow.

MARLlib effectively tackles this complexity by reorganizing the collective dataflow into a distributed system at the agent level. It treats each agent as a separate entity in both data gathering and optimization, while still allowing for the sharing of central data during the postprocessing phase (a feature in RLlib API that processes data prior to optimization, adapted here to support varied algorithms) to maintain consistency. During postprocessing, agents exchange both observed data (environmental samples) and predicted data (derived from their actions or Q-values). Each agent possesses its own data repository, collecting their individual experiences and shared information. Once learning commences, agents no longer need to share information and can independently optimize their strategies. This restructured dataflow ensures that agents are fully independent in

data management and optimization, enhancing the framework's ability to handle diverse task types.

Furthermore, while CTDE frameworks typically employ a similar agent-level dataflow, specific data processing techniques remain distinct. Drawing inspiration from EPyMARL, we categorize these techniques into three types: independent learning, centralized critic, and value decomposition, which facilitates the sharing of modules and broadens the framework's applicability. Independent learning allows for solo agent learning; centralized critic methods employ shared data to refine the critic, which in turn directs the decentralized agents' optimization; and value decomposition approaches derive a combined value function and dissect it for individual agent use during decision-making. These categorizations dictate our data-sharing tactics in the postprocessing phase, as depicted in Figure 5.2.

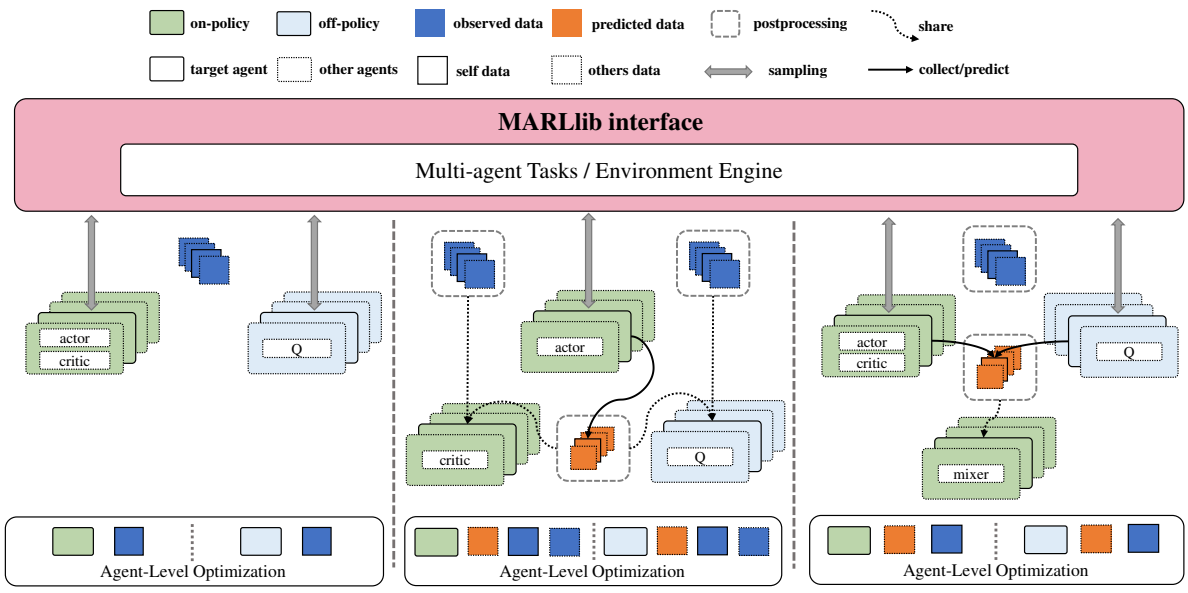
Thus, by maintaining the distinctive attributes of each algorithm and adopting an agent-level distributed dataflow, our approach demonstrates its effectiveness in harmonizing various algorithms under the CTDE umbrella, capable of addressing all modes of tasks while matching the efficacy of traditional implementations.

5.3.2 Universal Interface

In numerous reinforcement learning frameworks, the OpenAI Gym protocol involving `observation, reward, done, and info` is typically employed. However, this standard interface does not easily extend to MARL, where distinct challenges arise due to the presence of multiple agents, each experiencing individualized data streams. Additionally, in MARL, supplementary data such as action masks and global states are often available, and the reward structure can vary significantly, sometimes represented as either scalar values or dictionaries. Moreover, agent interactions with the environment may not occur simultaneously, introducing further complexities.

To navigate these complexities, MARLlib has redefined the multi-agent interaction interface in two primary ways:

Firstly, MARLlib has standardized the interface data structure. By adapting the conventional Gym API, MARLlib introduces a modified interface, `obs, reward, done, info`, which is versatile enough to accommodate various multi-agent scenarios. Specifically, the observation from the environment is encapsulated in a dictionary with keys `observation, action_mask, global_state`, making it suitable for a broad range of applications and consistent with RLlib's data handling conventions. Additional observational details are stored under `info`. The `reward` is structured as a dictionary keyed



(a) Independent Learning (b) Centralized Critic (c) Value Decomposition

Figure 5.2: Illustration of MARLlib's agent-level distributed dataflow. Observed data include environmental samples like rewards or global states, while predicted data comprise agent-generated metrics such as Q-values or selected actions. The postprocessing stage serves as the data sharing medium. Each agent upholds its distinct learning process, utilizing collected data for personal policy optimization, thereby achieving a distributed dataflow. The figure outlines three distinct dataflow models- independent learning, centralized critic, and value decomposition-each differentiated by their use of central data. Independent learning approaches, like IQL, inherently bypass data sharing, depicted in (a). Centralized critic models, such as MAPPO, amalgamate and disseminate both observed and predicted data during postprocessing to support distributed learning, as shown in (b). Value decomposition methods, like FACMAC, mandate the sharing of predicted data, with observed data sharing being optional based on the algorithm's specific needs, represented in (c).

by the agent ID. For cooperative endeavors, a scalar team reward is replicated across agents in this dictionary format. The `done` key is also a dictionary, featuring a singular key `"__all__"` to indicate the termination of all agents' activities.

Secondly, MARLlib enhances support for both synchronous and asynchronous interactions between agents and environments. Unlike previous MARL frameworks such as PyMARL that focus primarily on synchronous interactions, MARLlib accommodates the dynamics of asynchronous tasks, which are prevalent in complex games like Go and Hanabi. This is facilitated by Ray/RLlib's robust data collection system, where data are gathered and cataloged by agent ID. The entire dataset is only processed when a termination signal `done` is received, indicating the end of an episode.

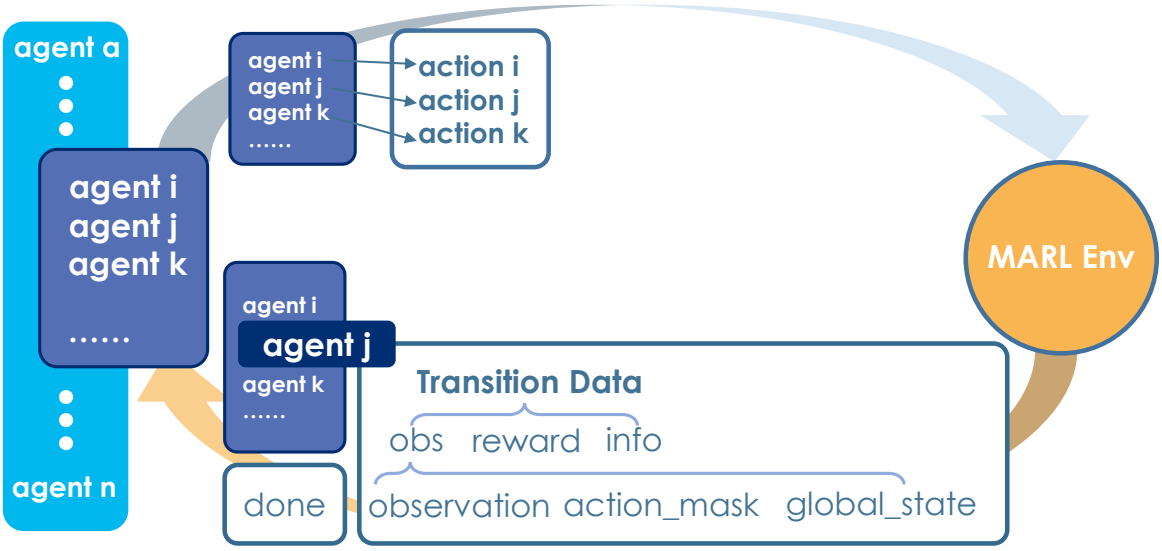


Figure 5.3: The agent-environment interaction in MARLlib integrates various environmental interfaces into a cohesive framework by introducing a Gym-inspired interface consisting of `obs, reward, done, info`. This unified interface caters to the complexities of multi-agent systems by structuring each component as a dictionary keyed by agent ID, ensuring that each agent's specific data is neatly organized and accessible. MARLlib is particularly versatile in supporting both synchronous and asynchronous interactions between agents and their environments, accommodating diverse operational scenarios. The `done` component serves as a universal signal that indicates the termination of all agents, ensuring a synchronized conclusion to the episode. Data is meticulously organized by agent within the figure to provide a clear visual representation of how MARLlib manages agent-specific information, illustrating the framework's capacity to handle intricate multi-agent dynamics efficiently.

This refined approach, depicted in Figure 5.3, ensures that MARLlib can effectively manage both the varied data needs of multi-agent environments and the different modes of agent interaction, thereby offering a unified and flexible platform for developing MARL applications.

5.3.3 Policy Mapping

In multi-agent settings, an effective parameter sharing strategy can significantly enhance the performance of algorithms. Unfortunately, many existing approaches offer limited sharing options and tend to involve redundant implementations. For instance, the MAPPO benchmark redesigns its architecture for both shared and separate parameter settings, whereas EPyMARL duplicates model structures to accommodate both sharing

modes.

MARLlib addresses these inefficiencies by introducing a versatile policy mapping API from RLlib, which supports three primary modes of parameter sharing: *full-sharing*, where all agents share the same parameters; *non-sharing*, where no parameters are shared among agents; and *group-sharing*, where agents within a designated group share parameters. This is achieved through the innovative use of a policy mapping function that links each agent's virtual policies to physical policies that are actively maintained, utilized, and optimized. Agents assigned to the same physical policy automatically share parameters.

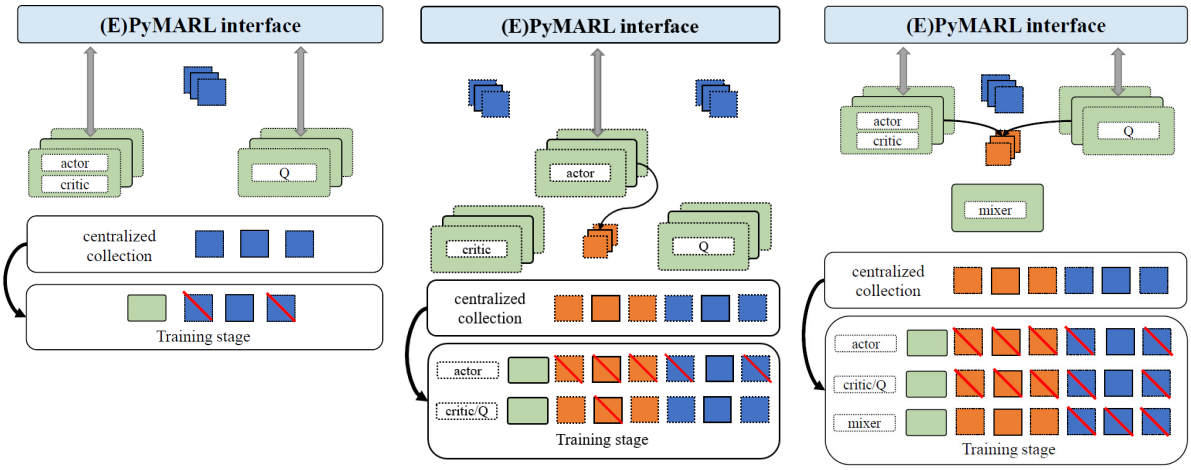
This policy mapping is transparent to agents, allowing them to sample data and undergo optimization as if they were interacting with their individual policies. Such a system enables diverse parameter sharing configurations without complicating the underlying algorithmic framework. In practical terms, MARLlib requires only a simple maintenance of a policy mapping dictionary for each environment, containing all necessary details to facilitate various sharing modes. Moreover, further customization of the parameter sharing strategy is feasible by adapting the policy mapping API to meet specific requirements, providing a flexible and efficient solution for complex multi-agent environments.

5.3.4 Equivalence Analysis

In this section, we endeavor to validate the proposition that any multi-agent learning framework can be effectively transformed into a unified single-agent learning process. Our objective is to establish that the data employed in optimizing the target entity is consistently the same across various methodologies. To this end, we will examine the well-recognized PyMARL and EPyMARL frameworks, noted for their efficiency. These frameworks are built around three core processes: data sampling, centralized data aggregation, and one-step training. We will juxtapose these frameworks with MARLlib's methodology, which is distinguished by its unique sampling method, sharing mechanism (post-processing), and agent-specific training. This comparative analysis will focus on three distinct learning methodologies as outlined below.

Independent Learning: Figures 5.4 and 5.5 display the independent learning strategies of (E)PyMARL and MARLlib. In (E)PyMARL, agents gather data collectively and store it in a central buffer prior to training. During training, agents from (E)PyMARL retrieve necessary data from this central repository, whereas MARLlib agents utilize

their own individual buffers. Ultimately, the same dataset is employed to refine each agent's policy, demonstrating that both approaches are fundamentally equivalent.

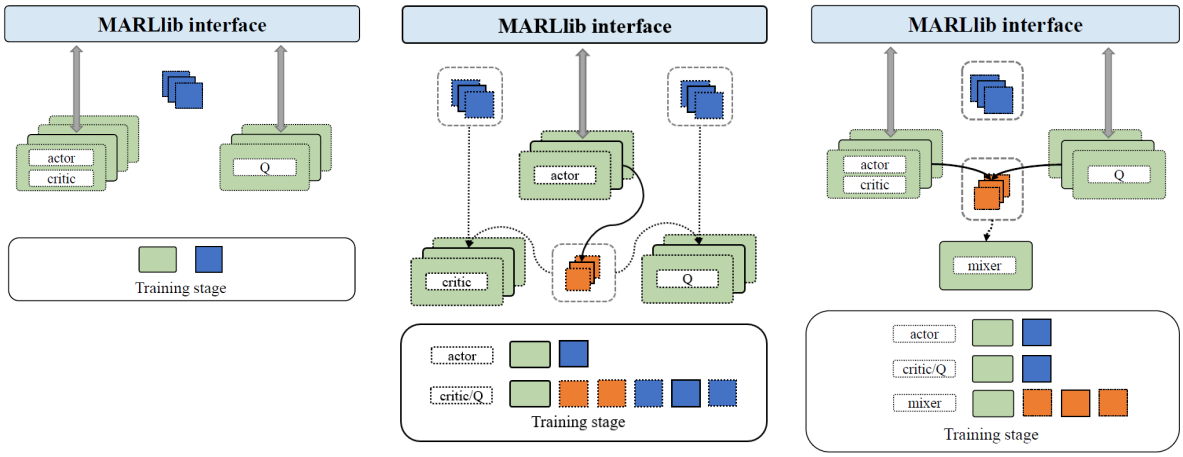


Independent Learning | Centralized Critic | Value Decomposition

Figure 5.4: Illustration of the learning processes for independent learning, centralized critic, and value decomposition in (E)PyMARL. Note that only one agent is shown in the training phase for clarity, although in practice, (E)PyMARL trains all agents concurrently.

Centralized Critic: The comparison of centralized critic models between (E)PyMARL and MARLlib is depicted in the middle sections of Figures 5.4 and 5.5. In (E)PyMARL, all gathered data is fed to the agents during training, where non-essential data is filtered out to optimize critical components, such as the critic/Q function. Conversely, in MARLlib, all pertinent data for different model components is prepared during the postprocessing phase before training. This preprocessing facilitates a smoother optimization process, adaptable to various learning styles (e.g., using the same objective for independent PPO and MAPPO). Ultimately, the identical dataset optimizes each agent, ensuring that the learning pipelines of (E)PyMARL and MARLlib are equivalent.

Value Decomposition: The right sections of Figures 5.4 and 5.5 illustrate the value decomposition models of PyMARL and MARLlib. Similar to the centralized critic scenario, in PyMARL, agents receive all collected data, select the necessary portions, and optimize various model components. In contrast, MARLlib's framework shares essential data, including Q/critic values, in the postprocessing stage. Each agent then optimizes its model independently using the maintained data, without the need to coordinate with others. This demonstrates that both learning methodologies employ the same data for policy updates, thus affirming their equivalence in the value decomposition approach.



Independent Learning | Centralized Critic | Value Decomposition

Figure 5.5: Learning pipelines of MARLlib for independent learning, centralized critic, and value decomposition.

5.3.5 Usage and Extensibility

```
from marllib import marl

# prepare env
env = marl.make_env(environment_name="mpe", map_name="simple_spread")
# initialize algorithm with appointed hyper-parameters
mappo = marl.algos.mappo(hyperparam_source="mpe")
# build agent model based on env + algorithms + user preference
model = marl.build_model(env, mappo, {"core_arch": "mlp", "encode_layer": "128-256"})
# start training
mappo.fit(
  env, model,
  stop={"timesteps_total": 1000000},
  checkpoint_freq=100,
  share_policy="group"
)
# rendering
mappo.render(
  env, model,
  local_mode=True,
  restore_path={'params_path': "checkpoint_000010/params.json",
                'model_path': "checkpoint_000010/checkpoint-10"}
)
```

MARLlib offers an API that combines user-friendliness with flexibility, simplifying library usage while maintaining its adaptability for user-specific modifications. This

design enables researchers to concentrate on their scientific inquiries without delving into complex implementation details.

The flexibility of MARLlib is supported by its modular architecture, comprising five main components: configuration, training script, algorithm, model, and environments. Each component is thoroughly documented with corresponding APIs, methods, or instructions, which facilitate straightforward customization and enhancement. We detail how MARLlib's flexibility can be exploited in various aspects of MARL research:

- To extend an algorithm's applicability to a broader range of task modes, such as transitioning from purely cooperative to mixed scenarios, researchers can modify scripts in the `marl/algo/scripts` directory. Adapting to new task modes may involve changing the policy mapping function within these scripts.
- To tailor an algorithm for complex or partially observable task structures, attention should be directed to the `marl/models` directory. For example, handling advanced tasks like Neural-MMO, which offers both 3D and 1D state observations, would necessitate modifications here.
- To develop a new algorithm, researchers can utilize the foundational elements found in the `marl/algos/core` and `marl/utils` directories.
- To establish baselines for novel multi-agent tasks, the `envs` directory should be considered. This directory facilitates the seamless integration of new tasks through the MARLlib agent-environment interface, enabling the testing of all existing MARLlib algorithms on these new scenarios.

The extensive adaptability of MARLlib supports a wide range of MARL experiments, marking it as a valuable resource for MARL research. Further capabilities are showcased in the `examples` directory of the MARLlib repository.

5.3.6 Linking with RLlib

MARLlib builds upon the robust foundation of RLlib to enhance and extend its functionalities specifically for (MARL. By leveraging RLlib's multi-agent task interface, MARLlib develops a cohesive and compatible agent-environment interface tailored for MARL experiments, enabling both researchers and developers to utilize the comprehensive functionalities of RLlib while benefiting from MARLlib's specialized optimizations for MARL tasks.

Challenges Presented by RLlib's Multi-Agent Functionality Despite RLlib's solid infrastructure for reinforcement learning, its multi-agent functionality poses several challenges, which is crucial for advancing research in MARL:

1. **Lack of a standardized unified agent-environment interface:** The complexity of MARL involving multiple agents within an environment, requires a well-defined agent-environment interface, which RLlib does not currently provide.
2. **Complexity and barrier to entry for newcomers:** The multi-agent aspect of RLlib can be daunting for those new to the field, necessitating a deep understanding of its core functionalities.
3. **Absence of a central integration point for diverse algorithms:** Without a centralized framework, RLlib struggles to provide a cohesive environment for comparing and integrating different MARL algorithms.

MARLlib Enhancements Over RLlib MARLlib not only builds on but also significantly refines RLlib's multi-agent capabilities, introducing key improvements to enhance its utility in MARL research. The relationship between MARLlib and RLlib is analogous to that between TensorFlow [1] and Keras [14], where Keras provides a high-level API that simplifies the use of TensorFlow's powerful backend. Similarly, MARLlib uses RLlib's infrastructure to facilitate the development of sophisticated multi-agent systems through an accessible, user-friendly interface. Key contributions of MARLlib include:

1. **Unified and Compatible Agent-Environment Interface:** MARLlib standardizes agent-environment interactions, streamlining data handling across various multi-agent tasks, thereby enhancing research efficiency.
2. **Simplified Abstraction and Algorithm Categorization:** MARLlib refines the information sharing stage, promoting effective data exchange and algorithm compatibility.
3. **Enhanced Accessibility for Newcomers:** The user-friendly API of MARLlib eases entry for new researchers into MARL.

Additionally, MARLlib introduces innovative features such as pipeline auto-adaptation and compatibility testing for training setups, complemented by detailed documentation to aid users in navigating the complexities of MARL. These enhancements empower users to customize their experiments extensively, fostering a more adaptable and inclusive MARL research environment.

Table 5.2: Comparative Analysis of MARLlib and RLlib

Features	**MARLlib**	**RLlib**
Data Handling	Structured	Broad & Flexible
Support for Multi-Agent Algorithms	Enhanced CTDE	Extended Single-Agent RL
Policy Mapping & Implementation	Automated	Requires Manual Effort
System Scalability and Integration	Derived	✓
User Friendliness	High	Moderate
Adaptability & Testing for Compatibility	✓	×
Research and Performance Evaluation	✓	Constrained
Support Resources	Extensive	Basic

5.4 Results

This section evaluates the performance of seventeen algorithms across twenty-three tasks from five prominent MARL test beds: SMAC [79], MPE [56], GRF [46], MAMuJoCo [71], and MAgent [113]. These test beds were selected due to their prominence in MARL research and their diversity in task modes, observation shapes, additional information provided, action spaces, reward density (sparse versus dense), and agent homogeneity versus heterogeneity. We present the average returns from these experiments, conducted using four random seeds, totaling over one thousand experiments. The results are depicted in Table 5.3 and Figure 5.6, validating the quality of implementation and providing a detailed analysis.

To demonstrate the correctness of MARLlib, we compare its performance on SMAC against that reported by EPyMARL, maintaining consistency in important hyperparameters. EPyMARL's results utilize 40 million steps for on-policy algorithms and four million for off-policy algorithms. MARLlib, in contrast, requires only half these steps for training convergence. Despite fewer training steps, MARLlib matches most of the performances reported by EPyMARL, as illustrated in Table 5.3. For all performance comparisons available, MARLlib achieves similar results in 63% of cases (with total reward differences under 1.0), superior results in 25% of cases, and inferior outcomes in the remaining 12%. The consistent expected performances across algorithms, without resorting to task-specific optimizations, underscore the implementation's correctness. This table also introduces, for the first time, the performances of five algorithms on SMAC and MPE, twelve on GRF, and ten on MAMuJoCo, serving as a reference for the community.

Table 5.3: Algorithmic performances for cooperative tasks across a range of control environments, including both discrete (SMAC, GRF, MPE) and continuous (MAMuJoCo) control tasks. Specifically, SMAC environments are highlighted with dual-row entries for each scenario. The initial row, styled in italics, reports performance metrics from EPyMARL, whereas the subsequent row provides data from experiments using MARLlib. For the remaining environments, performance metrics are exclusively reported for MARLlib. The symbol **-** denotes the absence of reported data, and cells with darker shading represent the top two performances in each scenario.

Env	Scenario	Independent Learning					Centralized Critic				Value Decomposition			
		IQL	IPG	IA2C	ITRPO	IPPO	MAA2C	COMA	MATRPO	MAPPO	VDN	QMIX	VDA2C	VDPPO
SMAC	2s_vs_1sc	*16.72*	-	*20.24*	-	*20.24*	*20.20*	*11.04*	-	*20.25*	*18.04*	*19.01*	-	-
		16.09	20.07	20.07	20.16	20.18	20.09	10.32	20.23	20.21	16.3	17.25	15.61	20.24
	3s5z	*16.44*	-	*18.56*	-	*13.36*	*19.95*	*18.90*	-	*19.91*	*19.57*	*19.66*	-	-
		16.73	10.78	13.49	10.04	14.3	15.21	9.78	12.1	19.52	19.38	19.32	8.58	13.15
	MMM2	*13.69*	-	*10.70*	-	*11.37*	*10.37*	*6.95*	-	*17.78*	*18.49*	*18.40*	-	-
		12.08	9.21	10.17	8.04	10.37	16.08	6.7	7.62	16.86	19.31	18.34	2.72	9.31
	3s_vs_5z	*21.15*	-	*4.42*	-	*19.36*	*6.68*	*3.23*	-	*18.17*	*19.03*	*16.04*	-	-
		16.78	5.6	10.79	3.39	7.95	12.14	4.79	13.32	17.24	18.55	19.84	9.6	14.61
MPE	simple_spread	-197.61	-63.83	-63.16	-78.16	-65.74	-63.37	-71.64	-77.63	-66.26	-190.5	-189.27	-190.66	-213.99
	simple_speaker_listener	-44.07	-261.65	-29.06	-50.17	-38.29	-27.76	-67.6	-44.01	-34.41	-35.26	-25.68	-54.37	-64.61
	simple_reference	-75.36	-36.3	-35.95	-57.79	-50.92	-35.05	-56.5	-47.71	-37.89	-70.56	-31.53	-69.35	-73.82
GRF	pass_and_shoot	-0.17	0.6	-0.03	0.6	0.5	-0.02	-0.01	0.48	0.74	-0.06	-0.24	0.05	0.01
	run_pass_and_shoot	-0.15	0.07	-0.07	-0.05	-0.07	-0.05	-0.03	-0.02	-0.03	-0.24	-0.11	-0.09	-0.13
	3_vs_1_with_keeper	0.02	0.33	0.01	0.37	0.05	0	0.03	0.13	0.45	-0.08	-0.06	0	0

		IPG	IA2C	IDDPG	ITRPO	IPPO	MAA2C	MADDPG	MAPPO	HAPPO	FACMAC	VDA2C	VDPPO
MAMuJoCo	2AgentAnt	143.22	-268.02	44.60	527.10	-153.46	730.8	18.53	-57.02	330.12	-1224.6	449.19	-98.74
	2AgentHalfCheetah	-133.06	-457.11	-197.85	1652.49	-644.89	-493.3	-313.95	-357.78	153.2	-433.61	-423	-644.53
	2AgentWalker	50.67	114.1	95.76	272.41	8.71	103.65	153.93	-4.12	164.45	-7.88	125.49	-3.76
	4AgentAnt	584.75	49.36	-971.28	750.96	-127.43	-1005.30	-419.93	149.1	151.85	-457.68	-338.21	-164.72
	6AgentHalfCheetah	-140.96	-302.99	-196.46	1492.24	-653.78	-257.76	-207.49	-529.43	442.48	-151.95	-588.66	-544.29

Performance Inheritance in Single-Agent RL Our empirical analysis indicates that basing MARL algorithms on robust single-agent RL algorithms is beneficial. For instance, PPO, which outperforms both vanilla PG and A2C in single-agent settings, yields more effective MARL versions-MAPPO and VDPPO show superior performance compared to MAA2C and VDA2C in various configurations. This advantage is supported by the effectiveness of value iteration methods, which exhibit less sensitivity to hyper-parameter variations and demonstrate greater sample efficiency than policy-gradient approaches. The multi-agent adaptations of Q learning, such as IQL, VDN, and QMIX, also benefit from these advantages, displaying strong performance in environments like SMAC and MPE.

Effectiveness Across MARL Algorithms Table 5.3 illustrates that different algorithms excel in specific tasks that align with their strategic designs. Independent learning proves advantageous in scenarios where centralized information is unnecessary. Despite theoretical concerns about its suboptimality in MARL contexts, research [19] shows that independent approaches can outperform centralized critics in tasks like `simple_spread` and `pass_and_shoot`, where similar agent behaviors do not require centralized data

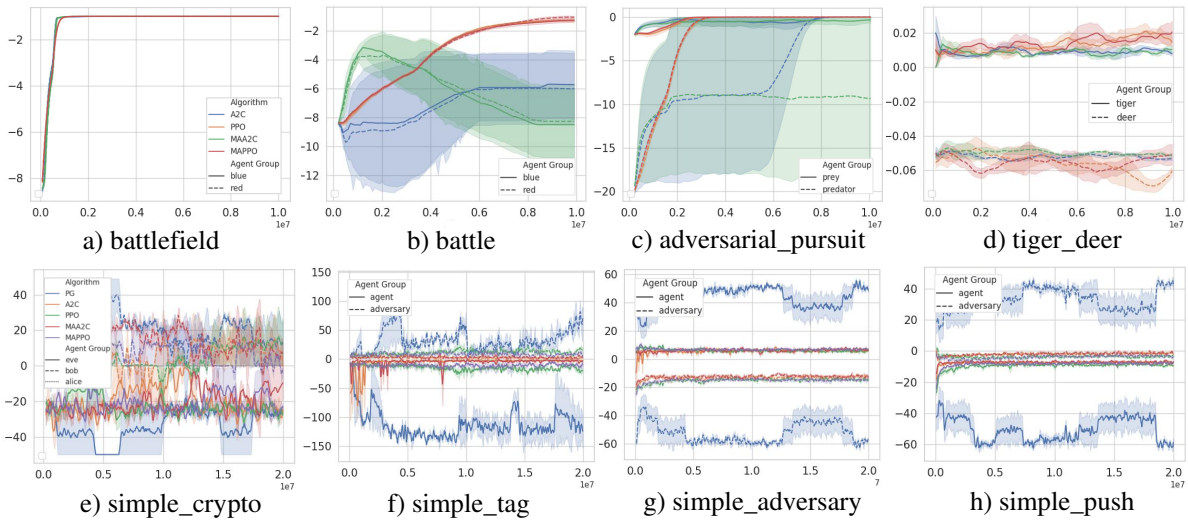


Figure 5.6: Return curves for eight mixed scenarios (agents compete in groups) in MAgent (a-d) and MPE (e-h) are displayed. Various styles of curves represent different groups of agents. These return curves depict a dynamic equilibrium throughout the learning process, with the equilibrium point being influenced by both the algorithms used and the specific tasks. For enhanced clarity, zoom in on the curves.

for policy optimization. Conversely, independent learning faces challenges in coordination tasks such as `simple_speaker_listener` and `simple_reference`, where a global perspective is crucial. Centralized critic strategies excel in tasks requiring diverse yet coordinated actions. These approaches, exemplified by MAPPO and its heterogeneous variant HAPPO, effectively leverage both local observations and global information, achieving robust results in cooperative environments across SMAC, MPE, and GRF. These algorithms set strong benchmarks for tasks where agent roles are distinct and varied. Value decomposition methods dominate in most cooperative benchmarks but fall short in two specific areas: continuous control tasks, where algorithms like VDN and QMIX are less effective, and scenarios requiring long-term planning with sparse rewards, such as those found in GRF. Here, the performance of value decomposition algorithms lags behind other strategies like ITRPO and MAPPO, primarily due to their preference for dense rewards and difficulties in decomposing near-zero Q values. Aside from these exceptions, value decomposition approaches generally deliver robust performance with optimal sample efficiency.

Evaluating Algorithms in Mixed Scenarios Assessing algorithms in mixed tasks, where agents must cooperate and compete simultaneously, presents unique challenges.

Determining the superiority of an algorithm from reward metrics alone is difficult, as the effectiveness of one policy often negatively impacts the performance of competing policies. In such mixed environments, algorithms are evaluated based on the total reward accrued by all involved policies. An optimized policy prompts competitors to enhance their strategies, thereby increasing the collective reward. This metric, depicted in Figure 5.6[a-d], serves as the primary indicator of an algorithm's effectiveness. However, exceptions occur, as shown in Figure 5.6[e-h], where the total reward remains constant, and the learning curves of competing policies mirror each other, reaching equilibrium swiftly. Identifying a consistent and fair evaluation criterion for these constant-sum tasks remains an ongoing area of research.

5.5 Conclusion

The introduction of MARLlib marks a significant advancement in MARL research. By addressing the prevalent challenges in current libraries, MARLlib offers a comprehensive, unified platform that eases the integration, testing, and deployment of MARL algorithms across diverse environments and task scenarios.

MARLlib introduces several innovative features: agent-level distributed dataflow, a unified multi-agent environment interface, efficient policy mapping, and thorough performance evaluations. These features establish MARLlib as an essential tool for both newcomers and seasoned researchers in the field. It effectively bridges the gap between the complexity of multi-agent interactions and the need for a modular, scalable framework that supports robust experimentation and development.

The refined interface and structured algorithm integration approach of MARLlib enable a more systematic exploration of MARL algorithms. This allows researchers to extend the limits of what is possible in scenarios ranging from strictly cooperative to highly competitive. Additionally, the adaptability of the library and its comprehensive documentation allow users to customize the framework to their specific needs with minimal overhead, fostering innovation and creative solutions to intricate challenges.

Moreover, the performance of MARLlib, demonstrated through extensive testing across various settings, highlights its efficiency and dependability. By achieving comparable or superior results to existing benchmarks while reducing computational demands, MARLlib redefines the standards of feasibility in MARL research.

As MARL gains relevance in diverse fields such as autonomous vehicles, robotics, finance, and healthcare, MARLlib emerges as a pivotal resource. It will enable researchers

and developers to pioneer new concepts, collaborate more efficiently, and realize groundbreaking achievements in AI and machine learning. The continued evolution of MARLlib is set to significantly impact the trajectory of MARL research, promoting a deeper understanding and more innovative applications of these complex systems.

In conclusion, MARLlib not only meets the current needs of the MARL community but also lays the groundwork for future innovations. Its comprehensive strategy for addressing the complexities of multi-agent environments and algorithms will serve as a foundational element for the next generation of MARL research. This will lead to more durable, scalable, and effective solutions applicable to a wide array of real-world challenges.

CHAPTER 6

CONCLUSION

This thesis has made significant strides towards developing a unified framework for Multi-Agent Reinforcement Learning (MARL), addressing the complexities and challenges associated with the coordination and learning of multiple agents within diverse and dynamic environments.

UPDeT A cornerstone of this unified framework is the introduction of UPDeT (Universal Policy Decoupling Transformer), a novel transformer-based model that significantly advances the capabilities of MARL systems. By leveraging the self-attention mechanism, UPDeT enhances the adaptability, scalability, and efficiency of learning in environments characterized by partial observability and sophisticated multi-agent coordination requirements. Its superior performance has been validated in complex simulation environments such as StarCraft II, highlighting its potential for broad real-world applications.

Role Diversity The examination of role diversity is pivotal in our pursuit of a unified MARL framework. This exploration offers a robust method for both evaluating and refining policies. By identifying and leveraging variations in role diversity, our framework promotes the development of more tailored and effective training strategies. This approach not only enhances the performance of MARL systems but also bolsters their robustness, contributing significantly towards the unification of MARL practices and methodologies.

MARLlib The development of MARLlib, a comprehensive library, supports the framework's goal of unification by providing tools that simplify the integration, testing, and deployment of various multi-agent tasks and MARL algorithms. MARLlib bridges the gap between theoretical research and practical application, enabling researchers to easily explore and deploy advanced MARL strategies in a range of environments and scenarios.

This thesis lay a solid foundation for future advancements in MARL, aligning closely with the goal of creating a unified, scalable, and highly adaptable MARL framework. Future research can further this goal by expanding the application domains of UPDeT, refining the metrics of role diversity for even deeper insights, and continuing to enhance the capabilities of MARLlib to support a wider array of MARL algorithms and environments. In essence, the progress made in this thesis not only pushes the boundaries of current MARL technologies but also sets a pathway for achieving a truly unified framework that could revolutionize how multi-agent systems are designed and implemented across various domains.

BIBLIOGRAPHY

[1] M. ABADI, P. BARHAM, J. CHEN, Z. CHEN, A. DAVIS, J. DEAN, M. DEVIN, S. GHEMAWAT, G. IRVING, M. ISARD, ET AL., *Tensorflow: a system for large-scale machine learning.*, in OSDI, 2016.

[2] J. ACHIAM, *Spinning Up in Deep Reinforcement Learning*, (2018).

[3] G. ARSLAN AND S. YÜKSEL, *Decentralized q-learning for stochastic teams and games*, IEEE Transactions on Automatic Control, 62 (2016), pp. 1545–1558.

[4] B. BAKER, I. KANITSCHEIDER, T. M. MARKOV, Y. WU, G. POWELL, B. MCGREW, AND I. MORDATCH, *Emergent tool use from multi-agent autocurricula*, in ICLR, 2020.

[5] N. BARD, J. N. FOERSTER, S. CHANDAR, N. BURCH, M. LANCTOT, H. F. SONG, E. PARISOTTO, V. DUMOULIN, S. MOITRA, E. HUGHES, ET AL., *The hanabi challenge: A new frontier for ai research*, Artificial Intelligence, (2020).

[6] R. BELLMAN, *Dynamic programming*, science, 153 (1966), pp. 34–37.

[7] A. BLUM AND Y. MONSOUR, *Learning, regret minimization, and equilibria*, (2007).

[8] M. BOWLING, *Convergence and no-regret in multiagent learning*, Advances in neural information processing systems, 17 (2004).

[9] M. BOWLING AND M. VELOSO, *Multiagent learning using a variable learning rate*, Artificial intelligence, 136 (2002), pp. 215–250.

[10] G. BROCKMAN, V. CHEUNG, L. PETTERSSON, J. SCHNEIDER, J. SCHULMAN, J. TANG, AND W. ZAREMBA, *Openai gym*, 2016.

[11] N. BROWN AND T. SANDHOLM, *Superhuman ai for multiplayer poker*, Science, 365 (2019), pp. 885–890.

[12] L. BUSONIU, R. BABUSKA, AND B. DE SCHUTTER, *A comprehensive survey of multiagent reinforcement learning*, IEEE TSMC, (2008).

[13] P. S. CASTRO, S. MOITRA, C. GELADA, S. KUMAR, AND M. G. BELLEMARE, *Dopamine: A Research Framework for Deep Reinforcement Learning*, (2018).

[14] F. CHOLLET ET AL., *Keras*.
`https://github.com/fchollet/keras`, 2015.

[15] F. CHRISTIANOS, G. PAPOUDAKIS, M. A. RAHMAN, AND S. V. ALBRECHT, *Scaling multi-agent reinforcement learning with selective parameter sharing*, in ICML, 2021.

[16] J. CHUNG, C. GULCEHRE, K. CHO, AND Y. BENGIO, *Empirical evaluation of gated recurrent neural networks on sequence modeling*, arXiv preprint arXiv:1412.3555, (2014).

[17] C. CLAUS AND C. BOUTILIER, *The dynamics of reinforcement learning in cooperative multiagent systems*, AAAI/IAAI, 1998 (1998), p. 2.

[18] M. CRAWSHAW, *Multi-task learning with deep neural networks: A survey*, arXiv preprint arXiv:2009.09796, (2020).

[19] C. S. DE WITT, T. GUPTA, D. MAKOVIICHUK, V. MAKOVIYCHUK, P. H. TORR, M. SUN, AND S. WHITESON, *Is independent learning all you need in the starcraft multi-agent challenge?*, arXiv preprint arXiv:2011.09533, (2020).

[20] J. DENG, W. DONG, R. SOCHER, L.-J. LI, K. LI, AND L. FEI-FEI, *Imagenet: A large-scale hierarchical image database*, in CVPR, 2009.

[21] P. DHARIWAL, C. HESSE, O. KLIMOV, A. NICHOL, M. PLAPPERT, A. RADFORD, J. SCHULMAN, S. SIDOR, Y. WU, AND P. ZHOKHOV, *Openai baselines*, 2017.

[22] Y. DU, L. HAN, M. FANG, J. LIU, T. DAI, AND D. TAO, *Liir: Learning individual intrinsic reward in multi-agent reinforcement learning*, in NeurIPS, 2019.

[23] J. FOERSTER, N. NARDELLI, G. FARQUHAR, T. AFOURAS, P. H. TORR, P. KOHLI, AND S. WHITESON, *Stabilising experience replay for deep multi-agent reinforcement learning*, in International conference on machine learning, PMLR, 2017, pp. 1146–1155.

[24] J. N. FOERSTER, G. FARQUHAR, T. AFOURAS, N. NARDELLI, AND S. WHITESON, *Counterfactual multi-agent policy gradients*, in AAAI, 2018.

[25] D. FUDENBERG AND J. TIROLE, *Game theory*, MIT press, 1991.

[26] S. GU, E. HOLLY, T. LILLICRAP, AND S. LEVINE, *Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates*, in 2017 IEEE international conference on robotics and automation (ICRA), IEEE, 2017, pp. 3389–3396.

[27] GYMNASIUM.
`https://gymnasium.farama.org`, 2016.

[28] M. HAUSKNECHT AND P. STONE, *Deep recurrent q-learning for partially observable mdps*, arXiv preprint arXiv:1507.06527, (2015).

[29] A. HILL, A. RAFFIN, M. ERNESTUS, A. GLEAVE, A. KANERVISTO, R. TRAORE, P. DHARIWAL, C. HESSE, O. KLIMOV, A. NICHOL, M. PLAPPERT, A. RADFORD, J. SCHULMAN, S. SIDOR, AND Y. WU, *Stable baselines*, 2018.

[30] S. HOCHREITER AND J. SCHMIDHUBER, *Long short-term memory*, Neural computation, 9 (1997), pp. 1735–1780.

[31] W. J. K. D. E. HOSTALLERO, K. SON, D. KIM, AND Y. Y. QTRAN, *Learning to factorize with transformation for cooperative multi-agent reinforcement learning*, in ICML, 2019.

[32] J. HU, S. WANG, S. JIANG, AND W. WANG, *Rethinking the implementation tricks and monotonicity constraint in cooperative multi-agent reinforcement learning*, in ICLR Blogposts, 2023.

[33] S. HU, C. XIE, X. LIANG, AND X. CHANG, *Policy diagnosis via measuring role diversity in cooperative multi-agent rl*, in ICML, 2022.

[34] S. HU, Y. ZHONG, M. GAO, W. WANG, H. DONG, X. LIANG, Z. LI, X. CHANG, AND Y. YANG, *Marllib: A scalable and efficient multi-agent reinforcement learning library*, Journal of Machine Learning Research, (2023).

[35] S. HU, F. ZHU, X. CHANG, AND X. LIANG, *Updet: Universal multi-agent rl via policy decoupling with transformers*, in ICLR, 2020.

[36] S. IQBAL, C. A. S. DE WITT, B. PENG, W. BÖHMER, S. WHITESON, AND F. SHA, *Randomized entity-wise factorization for multi-agent reinforcement learning*, in ICML, 2021.

[37] M. JADERBERG, W. M. CZARNECKI, I. DUNNING, L. MARRIS, G. LEVER, A. G. CASTANEDA, C. BEATTIE, N. C. RABINOWITZ, A. S. MORCOS, A. RUDERMAN, ET AL., *Human-level performance in 3d multiplayer games with population-based reinforcement learning*, Science, 364 (2019), pp. 859–865.

[38] J. JIANG AND Z. LU, *Learning attentional communication for multi-agent cooperation*, in NeurIPS, 2018.

[39] H. JIANYE, X. HAO, H. MAO, W. WANG, Y. YANG, D. LI, Y. ZHENG, AND Z. WANG, *Boosting multiagent reinforcement learning via permutation invariant and permutation equivariant networks*, in The Eleventh International Conference on Learning Representations, 2022.

[40] L. P. KAELBLING, M. L. LITTMAN, AND A. W. MOORE, *Reinforcement learning: A survey*, Journal of artificial intelligence research, 4 (1996), pp. 237–285.

[41] D. KIM, S. MOON, D. HOSTALLERO, W. J. KANG, T. LEE, K. SON, AND Y. YI, *Learning to schedule communication in multi-agent reinforcement learning*, in ICLR, 2019.

[42] M. KIM, J. OH, Y. LEE, J. KIM, S. KIM, S. CHONG, AND S. YUN, *The starcraft multi-agent exploration challenges: Learning multi-stage tasks and environmental factors without precise reward functions*, IEEE Access, 11 (2023), pp. 37854–37868.

[43] J. KOBER, J. A. BAGNELL, AND J. PETERS, *Reinforcement learning in robotics: A survey*, The International Journal of Robotics Research, 32 (2013), pp. 1238–1274.

[44] I. KOSTRIKOVK. `https://github.com/ikostrikov/pytorch-a2c-ppo-acktr-gail`, 2018.

[45] J. G. KUBA, R. CHEN, M. WEN, Y. WEN, F. SUN, J. WANG, AND Y. YANG, *Trust region policy optimisation in multi-agent reinforcement learning*, in ICLR, 2022.

[46] K. KURACH, A. RAICHUK, P. STANCZYK, M. ZAJAC, O. BACHEM, L. ESPEHOLT, C. RIQUELME, D. VINCENT, M. MICHALSKI, O. BOUSQUET, AND S. GELLY, *Google research football: A novel reinforcement learning environment*, in AAAI, 2020.

[47] M. LANCTOT, E. LOCKHART, J.-B. LESPIAU, V. ZAMBALDI, S. UPADHYAY, J. PÉROLAT, S. SRINIVASAN, F. TIMBERS, K. TUYLS, S. OMIDSHAFIEI, ET AL., *Openspiel: A framework for reinforcement learning in games*, arXiv preprint arXiv:1908.09453, (2019).

[48] M. LANCTOT, V. ZAMBALDI, A. GRUSLYS, A. LAZARIDOU, K. TUYLS, J. PÉROLAT, D. SILVER, AND T. GRAEPEL, *A unified game-theoretic approach to multiagent reinforcement learning*, NIPS, (2017).

[49] A. LAZARIDOU AND M. BARONI, *Emergent multi-agent communication in the deep learning era*, arXiv preprint arXiv:2006.02419, (2020).

[50] H. M. LE, Y. YUE, P. CARR, AND P. LUCEY, *Coordinated multi-agent imitation learning*, in ICML, 2017.

[51] N. LE, V. S. RATHOUR, K. YAMAZAKI, K. LUU, AND M. SAVVIDES, *Deep reinforcement learning in computer vision: a comprehensive survey*, Artificial Intelligence Review, (2022), pp. 1–87.

[52] J. Z. LEIBO, E. D. NEZ GUZMÁN, A. S. VEZHNEVETS, J. P. AGAPIOU, P. SUNEHAG, R. KOSTER, J. MATYAS, C. BEATTIE, I. MORDATCH, AND T. GRAEPEL, *Scalable evaluation of multi-agent reinforcement learning with melting pot*, PMLR, 2021.

[53] J. LI, S. KOYAMADA, Q. YE, G. LIU, C. WANG, R. YANG, L. ZHAO, T. QIN, T.-Y. LIU, AND H.-W. HON, *Suphx: Mastering mahjong with deep reinforcement learning*, arXiv preprint arXiv:2003.13590, (2020).

[54] E. LIANG, R. LIAW, R. NISHIHARA, P. MORITZ, R. FOX, K. GOLDBERG, J. GONZALEZ, M. JORDAN, AND I. STOICA, *Rllib: Abstractions for distributed reinforcement learning*, in ICML, 2018.

[55] B. LIU, Q. LIU, P. STONE, A. GARG, Y. ZHU, AND A. ANANDKUMAR, *Coach-player multi-agent reinforcement learning for dynamic team composition*, in International Conference on Machine Learning, PMLR, 2021, pp. 6860–6870.

[56] R. LOWE, Y. WU, A. TAMAR, J. HARB, P. ABBEEL, AND I. MORDATCH, *Multi-agent actor-critic for mixed cooperative-competitive environments*, in NIPS, 2017.

[57] A. MAHAJAN, T. RASHID, M. SAMVELYAN, AND S. WHITESON, *Maven: Multi-agent variational exploration*, in NeurIPS, 2019.

[58] MARL-ALGORITHMS. `https://github.com/starry-sky6688/MARL-Algorithms`, 2019.

[59] L. MATIGNON, G. J. LAURENT, AND N. LE FORT-PIAT, *Independent reinforcement learners in cooperative markov games: a survey regarding coordination problems*, The Knowledge Engineering Review, 27 (2012), pp. 1–31.

[60] L. MENG, M. WEN, C. LE, X. LI, D. XING, W. ZHANG, Y. WEN, H. ZHANG, J. WANG, Y. YANG, ET AL., *Offline pre-trained multi-agent decision transformer*, Machine Intelligence Research, 20 (2023), pp. 233–248.

[61] V. MNIH, A. P. BADIA, M. MIRZA, A. GRAVES, T. LILLICRAP, T. HARLEY, D. SILVER, AND K. KAVUKCUOGLU, *Asynchronous methods for deep reinforcement learning*, in ICML, 2016.

[62] V. MNIH, K. KAVUKCUOGLU, D. SILVER, A. GRAVES, I. ANTONOGLOU, D. WIERSTRA, AND M. RIEDMILLER, *Playing atari with deep reinforcement learning*, arXiv preprint arXiv:1312.5602, (2013).

[63] V. MNIH, K. KAVUKCUOGLU, D. SILVER, A. A. RUSU, J. VENESS, M. G. BELLEMARE, A. GRAVES, M. RIEDMILLER, A. K. FIDJELAND, G. OSTROVSKI, ET AL., *Human-level control through deep reinforcement learning*, Nature, (2015).

[64] P. MORITZ, R. NISHIHARA, S. WANG, A. TUMANOV, R. LIAW, E. LIANG, M. ELIBOL, Z. YANG, W. PAUL, M. I. JORDAN, ET AL., *Ray: A distributed framework for emerging {AI} applications*, in OSDI, 2018.

[65] Z. NING AND L. XIE, *A survey on multi-agent reinforcement learning and its application*, Journal of Automation and Intelligence, (2024).

[66] F. A. OLIEHOEK AND C. AMATO, *A concise introduction to decentralized POMDPs*, Springer, 2016.

[67] M. J. OSBORNE ET AL., *An introduction to game theory*, vol. 3, Oxford university press New York, 2004.

[68] G. OWEN, *Game theory*, Emerald Group Publishing, 2013.

[69] X. PAN, M. LIU, F. ZHONG, Y. YANG, S.-C. ZHU, AND Y. WANG, *MATE: Benchmarking multi-agent reinforcement learning in distributed target coverage control*, in Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track, 2022.

[70] G. PAPOUDAKIS, F. CHRISTIANOS, L. SCHAFER, AND S. V. ALBRECHT, *Benchmarking multi-agent deep reinforcement learning algorithms in cooperative tasks*, in NIPS Track on Datasets and Benchmarks, 2021.

[71] B. PENG, T. RASHID, C. SCHROEDER DE WITT, P.-A. KAMIENNY, P. TORR, W. BÖHMER, AND S. WHITESON, *Facmac: Factored multi-agent centralised policy gradients*, NIPS, (2021).

[72] P. POUPART, *Partially Observable Markov Decision Processes*, Springer US, 2010.

[73] PYMARL.
`https://github.com/oxwhirl/pymarl`, 2019.

[74] A. RAFFIN, A. HILL, A. GLEAVE, A. KANERVISTO, M. ERNESTUS, AND N. DORMANN, *Stable-baselines3: Reliable reinforcement learning implementations*, JMLR, (2021).

[75] T. RASHID, G. FARQUHAR, B. PENG, AND S. WHITESON, *Weighted qmix: Expanding monotonic value function factorisation for deep multi-agent reinforcement learning*, NIPS, (2020).

[76] T. RASHID, M. SAMVELYAN, C. S. DE WITT, G. FARQUHAR, J. N. FOERSTER, AND S. WHITESON, *QMIX: monotonic value function factorisation for deep multi-agent reinforcement learning*, in ICML, 2018.

[77] J. RENAULT, *A tutorial on zero-sum stochastic games*, arXiv preprint arXiv:1905.06577, (2019).

[78] C. RESNICK, W. ELDRIDGE, D. HA, D. BRITZ, J. N. FOERSTER, J. TOGELIUS, K. CHO, AND J. BRUNA, *Pommerman: A multi-agent playground*, in AAAI, J. Zhu, ed., 2018.

[79] M. SAMVELYAN, T. RASHID, C. S. DE WITT, G. FARQUHAR, N. NARDELLI, T. G. J. RUDNER, C. HUNG, P. H. S. TORR, J. N. FOERSTER, AND S. WHITESON, *The starcraft multi-agent challenge*, in AAMAS, 2019.

[80] J. SCHULMAN, F. WOLSKI, P. DHARIWAL, A. RADFORD, AND O. KLIMOV, *Proximal policy optimization algorithms*, arXiv preprint arXiv:1707.06347, (2017).

[81] R. SELTEN AND R. S. BIELEFELD, *Reexamination of the perfectness concept for equilibrium points in extensive games*, Springer, 1988.

[82] J. SHAO, Z. LOU, H. ZHANG, Y. JIANG, S. HE, AND X. JI, *Self-organized group for cooperative multi-agent reinforcement learning*, Advances in Neural Information Processing Systems, 35 (2022), pp. 5711–5723.

[83] Y. SHOHAM AND K. LEYTON-BROWN, *Multiagent systems: Algorithmic, game-theoretic, and logical foundations*, Cambridge University Press, 2008.

[84] D. SILVER, A. HUANG, C. J. MADDISON, A. GUEZ, L. SIFRE, G. VAN DEN DRIESSCHE, J. SCHRITTWIESER, I. ANTONOGLOU, V. PANNEERSHELVAM, M. LANCTOT, ET AL., *Mastering the game of go with deep neural networks and tree search*, nature, 529 (2016), pp. 484–489.

[85] D. SILVER, T. HUBERT, J. SCHRITTWIESER, I. ANTONOGLOU, M. LAI, A. GUEZ, M. LANCTOT, L. SIFRE, D. KUMARAN, T. GRAEPEL, T. LILLICRAP, K. SIMONYAN, AND D. HASSABIS, *A general reinforcement learning algorithm that masters chess, shogi, and go through self-play*, Science, (2018).

[86] A. SINGH, T. JAIN, AND S. SUKHBAATAR, *Learning when to communicate at scale in multiagent cooperative and competitive tasks*, in ICLR, 2019.

[87] J. SU, S. ADAMS, AND P. A. BELING, *Value-decomposition multi-agent actor-critics*, in AAAI, 2021.

[88] J. SUAREZ, Y. DU, C. ZHU, I. MORDATCH, AND P. ISOLA, *The neural MMO platform for massively multiagent research*, NIPS, (2021).

[89] S. SUKHBAATAR, A. SZLAM, AND R. FERGUS, *Learning multiagent communication with backpropagation*, in NIPS, 2016.

[90] P. SUNEHAG, G. LEVER, A. GRUSLYS, W. M. CZARNECKI, V. F. ZAMBALDI, M. JADERBERG, M. LANCTOT, N. SONNERAT, J. Z. LEIBO, K. TUYLS, AND

T. GRAEPEL, *Value-decomposition networks for cooperative multi-agent learning based on team reward*, in AAMAS, 2018.

[91] R. S. SUTTON AND A. G. BARTO, *Reinforcement learning: An introduction*, MIT press, 2018.

[92] M. TAN, *Multi-agent reinforcement learning: Independent vs. cooperative agents*, in ICML, 1993.

[93] J. K. TERRY, B. BLACK, N. GRAMMEL, M. JAYAKUMAR, A. HARI, R. SULLIVAN, L. S. SANTOS, C. DIEFFENDAHL, C. HORSCH, R. PEREZ-VICENTE, N. L. WILLIAMS, Y. LOKESH, AND P. RAVI, *Pettingzoo: Gym for multi-agent reinforcement learning*, in NIPS, M. Ranzato, A. Beygelzimer, Y. N. Dauphin, P. Liang, and J. W. Vaughan, eds., 2021.

[94] J. K. TERRY, N. GRAMMEL, A. HARI, L. SANTOS, AND B. BLACK, *Revisiting parameter sharing in multi-agent deep reinforcement learning*, arXiv preprint arXiv:2005.13625, (2020).

[95] E. TODOROV, T. EREZ, AND Y. TASSA, *Mujoco: A physics engine for model-based control*, in 2012 IEEE/RSJ international conference on intelligent robots and systems, IEEE, 2012, pp. 5026–5033.

[96] K. TUYLS AND G. WEISS, *Multiagent learning: Basics, challenges, and prospects*, Ai Magazine, 33 (2012), pp. 41–41.

[97] V. UC-CETINA, N. NAVARRO-GUERRERO, A. MARTIN-GONZALEZ, C. WEBER, AND S. WERMTER, *Survey on reinforcement learning for language processing*, Artificial Intelligence Review, (2023).

[98] H. VAN HASSELT, A. GUEZ, AND D. SILVER, *Deep reinforcement learning with double q-learning*, in Proceedings of the AAAI conference on artificial intelligence, vol. 30, 2016.

[99] S. VANDENHENDE, S. GEORGOULIS, W. VAN GANSBEKE, M. PROESMANS, D. DAI, AND L. VAN GOOL, *Multi-task learning for dense prediction tasks: A survey*, IEEE transactions on pattern analysis and machine intelligence, (2021).

[100] A. VASWANI, N. SHAZEER, N. PARMAR, J. USZKOREIT, L. JONES, A. N. GOMEZ, Ł. KAISER, AND I. POLOSUKHIN, *Attention is all you need*, in Advances in neural information processing systems, 2017, pp. 5998–6008.

[101] T. WANG, H. DONG, V. R. LESSER, AND C. ZHANG, *ROMA: multi-agent reinforcement learning with emergent roles*, in ICML, 2020.

[102] T. WANG, T. GUPTA, A. MAHAJAN, B. PENG, S. WHITESON, AND C. ZHANG, *RODE: learning roles to decompose multi-agent tasks*, in ICLR, 2021.

[103] W. WANG, T. YANG, Y. LIU, J. HAO, X. HAO, Y. HU, Y. CHEN, C. FAN, AND Y. GAO, *From few to more: Large-scale dynamic multiagent curriculum learning.*, in AAAI, 2020.

[104] Z. WANG, T. SCHAUL, M. HESSEL, H. HASSELT, M. LANCTOT, AND N. FREITAS, *Dueling network architectures for deep reinforcement learning*, in International conference on machine learning, PMLR, 2016, pp. 1995–2003.

[105] C. J. WATKINS AND P. DAYAN, *Q-learning*, Machine learning, 8 (1992), pp. 279–292.

[106] M. WEN, J. KUBA, R. LIN, W. ZHANG, Y. WEN, J. WANG, AND Y. YANG, *Multi-agent reinforcement learning is a sequence modeling problem*, Advances in Neural Information Processing Systems, 35 (2022), pp. 16509–16521.

[107] J. WENG, H. CHEN, D. YAN, K. YOU, A. DUBURCQ, M. ZHANG, Y. SU, H. SU, AND J. ZHU, *Tianshou: A highly modularized deep reinforcement learning library*, JMLR, (2022).

[108] Y. YANG AND J. WANG, *An overview of multi-agent reinforcement learning from game theoretical perspective*, CoRR, abs/2011.00583 (2020).

[109] C. YU, A. VELU, E. VINITSKY, J. GAO, Y. WANG, A. BAYEN, AND Y. WU, *The surprising effectiveness of PPO in cooperative multi-agent games*, in NIPS Datasets and Benchmarks Track, 2022.

[110] K. ZHANG, Z. YANG, AND T. BAŞAR, *Multi-agent reinforcement learning: A selective overview of theories and algorithms*, Handbook of Reinforcement Learning and Control, (2021).

[111] M. ZHANG, S. ZHANG, Z. YANG, L. CHEN, J. ZHENG, C. YANG, C. LI, H. ZHOU, Y. NIU, AND Y. LIU, *Gobigger: A scalable platform for cooperative-competitive multi-agent interactive simulation*, in International Conference on Learning Representations, 2023.

[112] Y. ZHANG AND Q. YANG, *A survey on multi-task learning*, IEEE Transactions on Knowledge and Data Engineering, 34 (2021), pp. 5586–5609.

[113] L. ZHENG, J. YANG, H. CAI, M. ZHOU, W. ZHANG, J. WANG, AND Y. YU, *Magent: A many-agent reinforcement learning platform for artificial collective intelligence*, in AAAI, 2018.

[114] Y. ZHONG, J. G. KUBA, X. FENG, S. HU, J. JI, AND Y. YANG, *Heterogeneous-agent reinforcement learning*, Journal of Machine Learning Research, (2024).

[115] M. ZHOU, Z. LIU, P. SUI, Y. LI, AND Y. Y. CHUNG, *Learning implicit credit assignment for cooperative multi-agent reinforcement learning*, Advances in Neural Information Processing Systems, 33 (2020), pp. 11853–11864.

[116] M. ZHOU, J. LUO, J. VILLELLA, Y. YANG, D. RUSU, J. MIAO, W. ZHANG, M. ALBAN, I. FADAKAR, Z. CHEN, A. C. HUANG, Y. WEN, K. HASSANZADEH, D. GRAVES, D. CHEN, Z. ZHU, N. NGUYEN, M. ELSAYED, K. SHAO, S. AHILAN, B. ZHANG, J. WU, Z. FU, K. REZAEE, P. YADMELLAT, M. ROHANI, N. P. NIEVES, Y. NI, S. BANIJAMALI, A. C. RIVERS, Z. TIAN, D. PALENICEK, H. BOU AMMAR, H. ZHANG, W. LIU, J. HAO, AND J. WANG, *Smarts: Scalable multi-agent reinforcement learning training school for autonomous driving*, 2020.

[117] M. ZHOU, Z. WAN, H. WANG, M. WEN, R. WU, Y. WEN, Y. YANG, Y. YU, J. WANG, AND W. ZHANG, *Malib: A parallel framework for population-based multi-agent reinforcement learning*, JMLR, (2023).

[118] T. ZHOU, F. ZHANG, K. SHAO, K. LI, W. HUANG, J. LUO, W. WANG, Y. YANG, H. MAO, B. WANG, ET AL., *Cooperative multi-agent transfer learning with level-adaptive credit assignment*, arXiv preprint arXiv:2106.00517, (2021).