Multi-Agent Deep Reinforcement Learning for Autonomous Energy Coordination in Demand Response Methods for Residential Distribution Networks

by

Peter Atrazhev

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science

in

Software Engineering and Intelligent Systems

Department of Electrical and Computer Engineering
University of Alberta

Abstract

In the field of collaborative learning and decision-making, this thesis aims to explore the effects of individual and joint rewards on the performance and coordination of agents in complex environments. The research objectives encompass two main aspects: firstly, to determine the objective superiority of joint rewards over individual rewards in centralized learning decentralized execution (CLDE) algorithms; secondly, to apply CLDE algorithms, both with individual and joint rewards, to evaluate the potential improvement in coordination for district energy management through centralized learning systems.

To achieve these objectives, an empirical analysis was conducted by varying the reward function between individual and joint rewards, as well as adjusting the episode length in the range of 25 to 50, using a selection of CLDE algorithms in the Level Based Foraging (LBF) environment. Subsequently, the same experimental framework was applied to the CityLearn challenge 2022.

The results of the first study reveal that different CLDE algorithms respond unpredictably when transitioning from joint to individual rewards in the LBF environment. Specifically, multi-agent proximal policy optimization (MAPPO) and QMIX demonstrate an ability to leverage the additional variance present in individual rewards, resulting in improved policies. Conversely, value decomposition networks (VDN) and multi-agent synchronous advantage actor critic (MAA2C) experience performance degradation due to increased variance. Notably, it was observed that centralized critic algorithms require a delicate balance, wherein the critic converges slowly enough to find optimal joint policies without being excessively sensitive to variance increases.

Furthermore, value decomposition methods exhibit a need for additional state information to effectively condition agent coordination for optimal policy learning. These findings indicate that the choice of reward function holds significant importance in Multi-Agent Reinforcement Learning (MARL) environments, potentially influencing the emergence of desired behavior.

The results of the second study shed light on the role and effectiveness of various CLDE algorithms and reward structures within the context of the CityLearn task. Comparisons between MAA2C, independent proximal policy optimization (IPPO),independent synchronous advantage actor critic (IA2C), and MAPPO, under individual and joint rewards, reveal substantial impacts on algorithm performance and efficiency. Specifically, MAA2C with individual rewards emerges as the most effective algorithm across multiple key performance indicators (KPIs), surpassing its competitors in peak demand and district ramping, and outperforming the random agent in all district KPIs. While IPPO and IA2C demonstrate strengths under individual rewards, they exhibit deficiencies compared to the random agent in certain areas. Conversely, MAPPO performs better with joint rewards, underscoring the nuanced differences between algorithms and the contextual conditions in which they excel.

This work highlights the significance of reconsidering joint rewards as the default choice for collaborative tasks. The findings suggest that individual rewards can be effectively employed in collaborative settings, with the choice between individual and joint rewards potentially presenting a bias-variance tradeoff. Further research is necessary to fully ascertain the implications of these results and refine the understanding of reward structures in collaborative learning and decision-making environments.

Preface

This thesis is an original work by Peter Atrazhev. The setup of this thesis is a staple thesis in which the core of the work is an accepted journal paper. In addition, this work contains unpublished work applying the results of the journal paper to the CityLearn environment. It's All About Reward: Contrasting Joint Rewards and Individual Reward in Centralized Learning Decentralized Execution Algorithms is co-authored by Peter Atrazhev and Petr Musilek. This work started out as a conference paper titled Investigating Effects of Centralized Learning Decentralized Execution on Team Coordination in the Level Based Foraging Environment as a Sequential Social Dilemma which was accepted at the International Conference on Practical Application of Agents and Multi-Agent Systems 2022 (PAAMS 2022). This initial conference work was then built on and accepted into MDPI Systems special Issue on Frontiers in Practical Applications of Agents, Multi-Agent Systems and Simulating Complex Systems.

The authors contributions were open-sourced code and the data was also open sourced so that the empirical work can be replicated.

"what is this, where am I"

- insert main character here

This thesis is dedicated to my parents, my brother and my partner in crime.

Acknowledgements

I would like to thank my supervisor Dr. Petr Musilek for his continuous and unending support during this degree; without your support this would have not been possible.

I would also like to thank Steven Zhang and Daniel May for their friendship and support during my years of study. You have enriched my university experience in ways that you don't even know. Thanks for being my friends.

All the members of the ENTAIL lab, I'm sorry for all the loud music, the screaming, the almost fires, the spray paint and all the dishes that were dirty for longer than they should have been.

Table of Contents

List of Tables

List of Figures

Abbreviations

AI Artificial intelligence.

AMI Advanced metering infrastructure.

CLDE centralized learning decentralized execution.

CNN Convolutional neural network.

DEC-POMDP Decentralized partially observable markov decision process.

DER Distributed energy resources.

DERMS Distributed energy resource management system.

DHW Domestic Hot Water.

DQN Deep Q-Networks.

DRL Deep Reinforcement learning.

HVAC Heating, ventilation, and air conditioning.

IA2C Independent synchronous advantage actor-critic.

IL Independent Learning.

IPPO Independent proximal policy optimization - used to distinguish between multi agent ppo.

IQL	Independent Deep Q-Learning.
KPI	Key performance indicator.
LBF	Level Based Foraging.
MAA2C	multi-agent independent synchronous advantage actor-critic.
MAPPO	Multi-agent proximal policy optimization.
MDP	Markov decision process.
ML	Machine learning.
PPO	Proximal policy optimization - same as independent.
PV	Photo voltaic.
RA	Random agent.
RBC	Rule based controller.
RNN	Recurrent neural network.
SMAC	Starcraft Multi-Agent Challenge.
SSD	Sequential social dilemma.
TE	Transactive energy.
VDN	Value decomposition networks.

Chapter 1

Introduction

This chapter establishes the motivation, introduces the objective and presents the outline of this thesis.

1.1 Motivation

The growing demand for energy management solutions has led to an increased interest in the application of Artificial Intelligence (AI) in Demand Response (DR) programs. The effective implementation of DR programs requires accurate and timely data, which has become increasingly available due to advances in metering infrastructure and the installation of broadband infrastructure. Moreover, regulatory policies that favor distribution grid efficiencies have created incentives for utilities to explore new technologies to improve their DR programs. In this context, we believe that the deployment of high-frequency Reinforcement Learning (RL) based DR systems at the edge of the grid has the potential to significantly enhance the effectiveness of DR programs. However, several implementation questions need to be addressed to realize this potential. Specifically, our research focuses on exploring the coordination mechanisms in Multi-Agent Reinforcement Learning (MARL) systems and their impact on algorithmic performance. By addressing these challenges, we aim to develop novel solutions that can effectively deploy AI in DR programs, enabling utilities to optimize energy management while addressing emerging concerns and realities associated with the deployment of high-frequency RL-based DR systems.

MARL is a promising field of AI research and over the last couple of years has seen increasingly more pushes to tackle less “toy” problems (full game environments such as ATARI and the Starcraft multi-agent challenge (SMAC) [4]) and instead try to solve complex “real world” problems. Coordination of agents across a large state space is a challenging and multi-faceted problem, with many approaches that can be used to increase coordination. These include communication between agents, both learned and established, parameter sharing and other methods of imparting additional information in function approximators, and increasing levels of centralization.

One paradigm of MARL that aims to increase coordination is called *Centralized Learning Decentralized Execution (CLDE)* [5]. CLDE algorithms train their agent’s policies with additional global information using a centralized mechanism. During execution, the centralized element is removed and the agent’s policy conditions only on local observations. This has been shown to increase the coordination of agents [1]. CLDE algorithms separate into two major categories: centralized policy gradient methods [6–8] and value decomposition methods [9, 10]. Recently, there has been work that put into question the assumption that centralized mechanisms do indeed increase coordination. Lyu et. al. [11] found that in actor-critic systems the use of a centralized critic leads to an increase in variance seen in the final policy learned however, they noted more coordinated agent behavior while training and concluded that the use of a centralized critic should be thought of as a choice that carries with it a bias-variance trade-off.

One aspect of agent coordination that is similarly often taken at face value is the use of a joint reward in cooperative systems that use centralization. The assumption is that joint rewards are necessary for the coordination of systems that rely on centralization. We have not been able to find theoretical backing for this claim. The closest works addressing team rewards in cooperative settings that we could find include works on difference reward which try to measure the impact of an individual agents actions on the full system reward [12]. The high learnability, among other nice properties, makes difference reward attractive but impractical due to the required knowledge of total system state [13–15].

1.2 Thesis Objectives

The objective of this thesis is to assess the effects of varying reward functions between joint reward and individual agent reward when training CLDE MARL systems. This main objective is broken down into several smaller research topics:

1. The Selection and evaluation of MARL environments that can support agents that gain independent rewards will be done.
2. The Selection of existing CLDE MARL algorithms that could be studied and can learn using individual reward will be performed.
3. Once a suitable existing environment and set of algorithms have been selected, a hyperparameter optimization of the algorithms will be performed.
4. Analysis of the effects of training with individual reward when compared to joint reward.
5. Application of this study to the CityLearn environment to showcase the effects of CLDE algorithms on energy management.

1.3 Thesis Contributions

The major contributions of this thesis were as follows:

(a) Our research indicates that individual rewards can be used successfully in collaborative scenarios, and the decision between individual and joint rewards may be a design choice.

(b) Our second findings imply that varying rewards make algorithm and task choice impact performance unpredictably, rendering reward type a hyperparameter needing specific tuning for each algorithm-task combination.

The significance of these contributions is that in Collaborative MARL tasks, it is possible that the type of reward that is given to the team of agents may be considered a hyperparameter.

1.4 Thesis Outline

This thesis is organized into 5 chapters. Chapter 2 provides background information on multi-agent reinforcement learning. Chapter 3 showcases work that is done on the analysis of CLDE mechanisms for coordination in environments that are similar in structure to the Citylearn environment for empirical underpinning. Chapter 4 sees the application of CLDE mechanisms to the CityLearn environment. Conclusions and future works are discussed in chapter 5. Various appendices support this document.

Chapter 2

Background

In this section, underlying ideas and background information will be briefly treated. This section first treats the background information on RL and MARL and culminates in the challenges of MARL. CLDE methods are then introduced as a possible answer to some of the challenges. Individual rewards and joint rewards as well as some more technical definitions arising from changes in reward in different scenarios coming from the game theory literature are introduced in this section as well. Moving away from the literature on machine learning, this section outlines background information on energy and electricity grid topics such as demand-side management techniques and how RL and MARL have been applied to demand-side management, setting the stage for section 4.2. The environments used in this work: LBF and Citylearn are also treated in depth in this section.

2.1 Reinforcement Learning

The forthcoming discussion provides a comprehensive introduction to deep learning (DL) and RL systems. In addition, it incorporates the mathematical definitions of multi-agent systems viewed through a game-theoretic lens, with a specific emphasis on sequential social dilemmas which form the crux of this study. An introduction to the concept of centralized learning with decentralized execution is also incorporated within the context. For those desiring a more in-depth exploration of RL, we recommend the textbook: Reinforcement Learning, An Introduction by Drs. Richard Sutton and Andrew Barto [16].

2.1.1 Intro to Reinforcement Learning

Formal RL Framework

The RL problem is a way of framing the problem of learning from interaction to achieve a goal. *Agents*, the decision maker, interact with the *environment*, which includes everything that is not the agent, to achieve a specific goal. In order to learn the goal, the environment provides the agent a *reward*, an indicator of the performance of the agent's actions. The agent and environment interact sequentially at discrete time steps $t = 0, 1, 2, 3, ..$, allowing the agent to learn from the actions it takes. At each timestep, t, the environment provides the agent with state information $s_t \in S$ where $\mathbb{S}$ is the set of all states that the agent can attain. In addition to this state information, the environment also provides the reward for the previous action $r_t - 1 \in$. The sequential nature of the RL framework creates an experience stream that is in the form $S_0, A_0, R_1, S_1, A_1, R_2, S_2, A_2...$ .The agents learn a policy π from the stream of experience that maps from states to probabilities of selecting a certain action. The above definition describes a very general version of the reinforcement learning problem and many modifications can be added to reflect the nature of more complicated systems. *Environments* come in many different forms but the most general categorization of environments is in terms of information that the agent receives; fully observable or partially observable. Fully observable environments give the agent complete information access to the entire environment allowing it to condition on all available information. Partially observable environments only give the agent information about a region of the environment that it inhabits, termed observations, forcing the agent to only condition on these local observations. *Environments* and by extension problem settings can also vary the number of agents that are interacting in the environment at the same time.

Markov Decision Process Definition:

The RL problem can be written as a Markov decision process is described by the tuple $\langle \mathbb{S}, \mathbb{A}, P, R, \gamma \rangle$

- $\mathbb{S}$: The set of environment states (finite).
- $\mathbb{A}$: The set of actions available to the agent.
- $\mathbb{P}$: The transition function that maps state action pairs to the next state.
- $R \in \mathbb{R}$ this is the reward function that the agents receive their reward from.
- $\gamma \in [0, 1]$ is the discount rate that is applied to the rewards.

RL objective

The overall objective of an RL agent is to maximize the cumulative discounted reward that it receives during its interactions with its environment. This is captured formally by an agent seeking to maximize the expected discounted return which is the discounted sum of all its rewards:

$$G_t = \sum_{t=0}^{n} \gamma^t R_t = R_0 + \gamma R_1 + \gamma^2 R_2 + ...\gamma^n R_n \tag{2.1}$$

Different Components to Learn a Policy

To learn a policy, a reinforcement learning (RL) agent requires several essential components that work in harmony. Firstly, the agent needs a way to interact with the environment, observe states and receive rewards while executing actions based on its current policy. Secondly, the agent must have a learning algorithm, which could be value-based (e.g., Q-learning[17]), policy-based (e.g., REINFORCE [18, 19]), or a hybrid (e.g., actor-critic [20–22]), to update its knowledge and improve its decision-making process over time. This learning algorithm typically leverages a function approximator, such as a neural network, to represent the policy or value function and generalize across state-action space. Finally, the agent needs an exploration strategy, such as ϵ-greedy or entropy-based methods, which guides the agent to balance between exploiting its current knowledge and exploring unknown parts of the environment. By integrating these components, an RL agent can effectively learn a policy to achieve its objectives in a wide range of tasks and environments.

Value-based Methods

Value-based RL methods focus on estimating the value of taking certain actions in particular states, to discover the optimal policy that maximizes the expected cumulative reward. One prominent value-based algorithm is Q-learning [17], which learns an action-value function, known as the Q-function. The Q-function, denoted as $Q(s, a)$, represents the expected cumulative reward of taking action a in state s and following the optimal policy thereafter.

Q-learning operates by iteratively updating the Q-function using the Bellman equation, which relates the value of a state-action pair to the value of its successor state-action pairs. The Q-learning update rule is given by:

$$Q\left(s_t, a_t\right) \leftarrow Q\left(s_t, a_t\right) + \alpha\left[r_t + \gamma \max_{a_{t+1}} Q\left(s_{t+1}, a_{t+1}\right) - Q\left(s_t, a_t\right)\right] \tag{2.2}$$

Here, α is the learning rate, r_t is the reward received after taking action a_t in state s_t, and γ is the discount factor that determines the importance of future rewards. The update rule adjusts the Q-function based on the difference between the observed reward and the current estimate, known as the temporal difference (TD) error. By repeatedly applying this update rule, Q-learning converges to the optimal Q-function, which can then be used to derive the optimal policy by selecting the action with the highest Q-value in each state.

Value-based RL methods like Q-learning have demonstrated their effectiveness in various tasks, especially when combined with function approximation techniques such as deep learning, resulting in powerful algorithms like Deep Q-Networks [23] that can handle large and complex state-action spaces.

Policy Gradient Methods

Policy gradient RL methods directly optimize the policy by following the gradient of the objective function, which aims to maximize the expected cumulative reward. One of the foundational policy gradient algorithms is REINFORCE [18, 19], which adjusts the policy's parameters based on the policy gradient theorem. The policy gradient theorem states that the gradient of the objective function is given by:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta} \left[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta \left(a_t \mid s_t \right) \cdot G_t \right] \tag{2.3}$$

Here, θ represents the policy parameters, $J(\theta)$ is the objective function, $\pi_\theta(a_t|s_t)$ denotes the probability of taking action a_t in state s_t according to the policy, and G_t is the cumulative reward from time step t onwards. The gradient of the objective function indicates the direction in which the policy should be updated to maximize the expected return.

REINFORCE operates by sampling trajectories, calculating the gradient of the objective function using the sampled rewards, and updating the policy accordingly. However, REINFORCE is known for its high variance in gradient estimates, which can lead to slow or unstable learning. To address this issue, actor-critic models have been developed, which combine policy gradient methods with value-based methods, utilizing a critic to estimate the value function and provide more stable updates to the actor's policy, as described in the following section.

Actor Critic Methods

Actor-critic methods are a class of reinforcement learning algorithms that combine the strengths of both policy gradient and value-based methods to achieve more efficient and stable learning [20–22]. In actor-critic methods, there are two separate components: the actor, which is responsible for learning the policy, and the critic, which estimates the value function. The actor takes actions based on the current policy, while the critic evaluates the quality of the actions taken and provides feedback to update the policy.

The policy gradient theorem provides the foundation for updating the policy in actor-critic methods. It states that the gradient of the objective function, which aims to maximize the expected return, can be calculated as follows:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim \pi\theta}\left[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta \left(a_t \mid s_t\right) \cdot A\left(s_t, a_t\right)\right] \tag{2.4}$$

Here $A(s_t, a_t)$ is the advantage function, which quantifies the relative quality of an action compared to the average action in a given state. The advantage function can be estimated using the critic's value function estimates.

$$A(s_t, a_t) = \hat{Q}(s_t, a_t) - \hat{V}(s_t)$$

By following the policy gradient, the actor updates its policy to favor actions that yield higher advantages, gradually improving the policy to maximize the expected return. This collaborative learning process between the actor and critic enables actor-critic methods to tackle complex reinforcement learning tasks with greater efficiency and stability.

Different RL Approaches

Model-free and model-based RL are two fundamental approaches to RL that differ in the way they utilize knowledge of the environment. These approaches can be integrated with both tabular and deep RL methods, resulting in a diverse array of techniques for solving RL problems.

Model-free RL methods, such as Q-learning[17] and SARSA [24, 25], learn a policy or value function directly from the interactions with the environment without explicitly constructing a model of the environment's dynamics. This approach can be combined with tabular methods for small and discrete environments, but when faced with large or continuous action and observation spaces, it can be integrated with deep RL to leverage function approximation techniques, as seen in algorithms like Deep Q-Networks (DQN [17]) and Deep Deterministic Policy Gradients (DDPG)[26].

On the other hand, model-based RL methods aim to learn an explicit model of the environment's dynamics, which can then be used for planning and decision-making. When combined with tabular methods, model-based RL can exploit the learned model to plan optimal actions in small environments using algorithms like Dyna-Q [16]. For more complex problems with larger state spaces or continuous domains, model-based RL can integrate with deep learning techniques, utilizing neural networks to approximate the environment's model. This integration results in advanced algorithms like Neural Network Predictive Control (NNPC) and Model-Based Policy Optimization (MBPO), which demonstrate improved sample efficiency and scalability compared to their model-free counterparts.

In summary, both model-free and model-based RL approaches can be combined with tabular and deep RL techniques, yielding a rich set of algorithms that cater to different problem complexities and requirements. These combinations offer a flexible toolbox for RL practitioners to tackle diverse challenges across various domains.

Tabular and deep RL represent two distinct approaches to solving RL problems, each with its unique set of challenges and advantages. Tabular RL methods rely on creating and maintaining tables for storing state-action values, allowing for exact solutions in small, discrete environments. However, as the size of the action and observation spaces grows, tabular methods quickly become infeasible due to the exponential increase in memory and computational requirements. In contrast, deep RL leverages function approximation techniques, such as neural networks, to estimate state-action values or policies, making it possible to tackle problems with large or continuous action and observation spaces.

Despite the advantages of deep RL, it introduces its own set of challenges, such as the "deadly triad" – the combination of function approximation, off-policy learning, and bootstrapping. The deadly triad can lead to instability and divergence in learning algorithms, necessitating the development of specialized techniques to ensure convergence and stability, such as experience replay and target networks. Furthermore, the use of function approximators in deep RL can result in the loss of theoretical guarantees available in tabular methods, and the optimization of neural networks can be slower and more computationally intensive. Nonetheless, deep RL has proven its potential in tackling complex problems across various domains, thanks to its ability to manage large action and observation spaces through function approximation.

Deep RL

Deep reinforcement learning (DRL) combines the power of deep learning techniques with reinforcement learning to create intelligent agents capable of learning complex tasks through trial and error. Various neural network architectures are employed in DRL systems to enable efficient learning, generalization, and adaptation. Some prominent architectures include feedforward networks, such as deep Q-networks (DQN) for value-based methods; recurrent neural networks (RNN), which capture temporal dependencies and are especially useful in partially observable environments; and convolutional neural networks (CNN) [27], which excel at extracting spatial features from raw data, making them ideal for tasks involving images and grid-based environments. Additionally, more advanced architectures, such as policy gradient methods with actor-critic networks and graph neural networks, have been introduced to tackle more complex tasks that require understanding relational information or managing high-dimensional continuous action spaces. These diverse architectures facilitate the development of robust and versatile DRL systems capable of addressing a wide range of real-world problems.

2.1.2 Introduction to Multi-Agent Reinforcement Learning

Multi-Agent Settings

The multi-agent domain in RL involves tasks where multiple agents, each with their own learning and decision-making processes, interact within a shared environment to achieve their goals. These tasks present unique challenges and opportunities compared to single-agent RL scenarios, as agents must learn to cooperate, compete, or coexist with one another to achieve their respective objectives. Multi-agent reinforcement learning (MARL) has garnered significant interest due to its potential applications in diverse fields, such as robotics, transportation, and finance. In MARL settings, agents must navigate complex dynamics arising from the interactions between themselves and the environment, necessitating the development of advanced algorithms and learning techniques to address issues like coordination, credit assignment, and communication among agents [28].

Formalization of Multi-Agent Problems

The defacto model used to describe a class of problems that appear in multi-agent settings is the DEC-POMDP. The DEC-POMDP [5] is an extension of the MDP introduced in Section 2.1.1 and takes into account not only multiple agents but also the possibility of agents not being able to observe the entire environment space at once, known as *partial observability.* Partial observability adds a whole other layer of complexity to the dynamics of the system from the individual agents perspective as there are elements of the dynamics that may not be observable to the agent but has direct effects on the reward that the agent receives for the actions it performed. This may also change over time as the agent explores its environment. Generally speaking, the introduction of other learning agents into the MDP framework breaks the Markov property and causes increased difficulties in learning for the agents [28].

DEC-POMDP: A Decentralized partially observable Markov Decision Process (Dec-POMDP) can be defined by a tuple $M = (\mathbb{D}, \mathbb{S}, \mathbb{A}, \mathbb{T}, O, \mathbb{O}, R, h, b_0)$ where:

- $\mathbb{D} = 1...., n$ is the set of n agents.
- $\mathbb{S}$ is a set of states (finite).
- $\mathbb{A}$ is the set of joint actions.
- $\mathbb{T}$ is the transition probability function.

- O is the set of joint observations.
- $\mathbb{O}$ is the observation probability function.
- R is the immediate reward function.
- h is the horizon of the problem.
- b_0 is the initial state distribution.

This tuple extends the single-agent POMDP model to the multi-agent setting by including the joint action and state spaces of all agents. These joint spaces are sets that contain observations and actions from all agents.

The shift from single-agent RL to multi-agent RL brings with it several additional parameters by which to define tasks one of which is based on how agents interact while they attempt to accomplish their goals. Tasks can be categorized as cooperative, competitive or mixed tasks. In the first two cases, agents can cooperate to accomplish a goal or be pitted against each other in competition. In the third case, agents may have a joint goal but be competing with each other for resources to accomplish this goal. This creates the necessity for agents to coordinate with each other. Depending on the type of task, different methods of coordination may be used in MARL. This third type of task is the task that was chosen for study in this work and research in this area is currently ongoing.

2.1.3 Challenges in MARL

MARL presents a set of unique challenges that stem from the interactions among multiple agents in a shared environment. Firstly, multi-agent dynamics result in the breaking of the Markov property as the reward and action for an individual agent now depend on not only its own actions but also those of other agents. This increased complexity makes it difficult to achieve optimal decision-making as the same action in the same state no longer guarantees the same reward due to the actions of other agents. Secondly, the curse of dimensionality arises due to an exponential growth in the number of features as the number of agents increases, especially in systems where agents directly observe each other. This feature explosion complicates the learning process and demands more computational power to converge to optimal solutions [28].

Another challenge is action shadowing, which occurs when suboptimal actions appear more favorable than optimal ones due to the interconnected nature of optimal actions and other agents' policies. This phenomenon is particularly prevalent in joint reward settings, where coordination is needed to discover the optimal action but an alternative policy that doesn't rely on coordination is overestimated as superior by the agents. Lastly, multi-agent credit assignment and coordination present difficulties in determining which agents' actions contribute the most to the overall success of the system. This challenge is unique to joint reward settings and necessitates the development of mechanisms to accurately attribute credit and facilitate coordination among agents.

2.1.4 Centralized Learning Decentralized Execution (CLDE)

CLDE is a strong method for performance using MARL on fully cooperative problems. This method attempts to coordinate agents by training agents' individual policies using a centralized system that conditions over all agent's observations [7–9]. During execution, agents only condition on their local observations and rewards. This allows for the creation of collaboration without requiring that agents share any information at run time. CLDE methods fall into two categories: the centralized policy gradient methods [6–8] and the value function decomposition methods [9, 10].

2.1.5 Value based CLDE

Value decomposition methods used were Value Decomposition Networks (VDN) [10] and its more contemporary counterpart QMIX [9]. Value decomposition methods train a factored joint state-action value function using the standard DQN algorithm. VDN decomposes the value function as the sum of individual value functions while QMIX employs a hyper-network conditioned on state information to factor the state-value function.

2.1.6 Centralized Critic Based CLDE

Chosen centralized policy gradient algorithms are multi-agent independent synchronous advantage actor-critic (MAA2C) and multi-agent proximal policy optimization (MAPPO) [7]. MAA2C and MAPPO are both multi-agent versions of their independent counterparts with a joint critic conditioning on all agent observations.

2.1.7 Reward Design

Reward design in RL is critical as the reward signal defines the goal of a RL problem and is the primary basis for altering the agent's policy. In multi-agent scenarios an additional factor is present for reward design: how are agents in the task rewarded. The designer needs to ask themselves "should agents share rewards or not?" "Does the scenario allow for agents to share rewards or not?" Multi-agent scenarios allow for either joint rewards or individual rewards and this has significant consequences to how agents work together to achieve the goal.

Joint reward:

The whole team receives a joint reward value at each timestep taken as the sum of all individual agent rewards $R = R^i = ... = R^N = \sum_{i=1}^{N} R_t^i$. The joint reward has an interesting property that is usually left aside: by being the summation of all agent's rewards if an agent is not participating in a rewarding event they still receive a reward. This creates a small but non-zero probability for all agents to receive rewards in any state and for any action. In addition, in partially observable tasks, these reward events can occur with no context for some of the agents. The advantage of the joint reward is a salient signal across all that can be learned from, as well as additional information about the performance of team members that may or may not be observable.

Individual reward:

Mixed tasks differ from the fully cooperative case only in terms of the reward that is received by agents. Mixed tasks attribute individual rewards to each agent rather than a joint reward, making the R term in the tuple M, $R = R_t^i$ for each agent i. During reward events,a reward is only given to agents that participate in reward events. This reduces the salience of the reward signal during a reward event and can cause increased variance in the reward signal when different agents achieve reward. The advantage of individual rewards is that actions that do not contribute to reward are not artificially encouraged as with joint rewards. The joint reward can create situations where agents whose state and action does not contribute at all to reward events still get rewarded, which changes the value of those states and actions even if they are not favorable to the discovery of the optimal policy. This can lead to sub-optimal policies and even the lazy agent problem, specifically in centralized systems.

2.2 Test environments

2.2.1 Sequential Social Dilemmas

Sequential social dilemmas (SSD) are described as problems in which individual rationality leads to collective irrationality. In the multi-agent setting, sequential social dilemmas are categorized into two broad categories: the provision of public goods and the tragedy of the commons, of which we focus on the latter. The commons problem involves individuals being tempted by an immediate benefit that produces a cost shared by all if all players succumb to temptation [29]. The key to solving the commons problem is for all agents to coordinate their actions to avoid premature exhaustion of a shared resource.

MARL has emerged as an empirical method for studying temporally complex SSDs that feature more than two agents [30–33]. These studies have produced a robust definition of the SSD and its mechanics [30–32]. To analyze a SSD, the joint action set A is split into the set $[C, D]$ where C is the set of collaborative actions and D is the set of defective actions. This categorization allows for the classification of policies to be collaborative or defective. If the SSD is simple or can be decomposed into a simple matrix game then there are analytical methods for proving that the situation is an SSD [31].

2.2.2 MARL Coordination in SSDs

MARL coordination in SSDs has been studied through different mechanisms by which agents can be incentivized to collaborate without necessarily sharing information inspired by psychology and sociology. These mechanisms range from imbuing the reward function with terms that create social pressures on the agents [30], to outright enabling actions that punish other agents, allowing the creation and enforcement of taboo behavior in agent systems [33]. Some of these mechanisms observe the actions and rewards of other agents which may not always be available or favourable for computation in large systems during run time.

Most of these methods study algorithmic effects on agent behavior and focus on investigating changes in agent policy as a result of the imbued mechanism.

We note a lack of investigations that include CLDE algorithms in both mentioned areas of research. This is surprising since CLDE algorithms have been known to increase performance even in very challenging, fully cooperative environments such as the Starcraft Multiagent environment [4]. We were able to find some works that investigate the effects of centralized critics on actor-critic methods, challenging the underlying assumptions that centralized critics make it easier to learn tasks [11].

2.2.3 Level based Foraging Environment

Level-Based Foraging (LBF) is a challenging exploration problem in which multiple agents must work together to collect food items scattered randomly on a grid-world [34]. The environment is highly configurable, allowing for partial observability and the use of cooperative policies only. In LBF, agents and food are assigned random levels, with the maximum food level always being the sum of all agent levels. Agents can take discrete actions, such as moving in a certain direction, loading food, or not taking any action. Agents receive rewards when they successfully load a food item, which is possible only if the sum of all agent levels around the food is equal to or greater than the level of the food item. Agent observations are discrete and include the location and level of all food and agents on the board, including themselves.

The LBF environment is highly configurable starting with gridworld size, number of agents and number of food. Scenarios in the LBF are described using the following nomenclature: *NxM-Ap-Bf* where *N* and *M* define the size of the gridworld, *A* indicates the number of agents and *B* indicates the number of food objectives in the world. A 10 by 10 grid world with three agents and three food would be described by *10x10-3p-3f*. Additionally, partial observability can be configured by adding *Cs-* before the grid size. *C* defines the radius size that agents can observe. For all objects outside the radius, the agent will receive a constant value of -1 in that observation. Finally, the addition of the *-coop* tag after the number of food causes the game to enforce all agents to be present to collect food, thereby forcing cooperative policies to be the only policies that can be learned. As an example, an eight by eight gridworld (N=8, M=8) with two players (A=2) and two food (B=2) which forces cooperative policies while subjecting the agents to partial observability with a radius of two (C=2) would be specified as *2s-8x8-2p-2f-coop*.

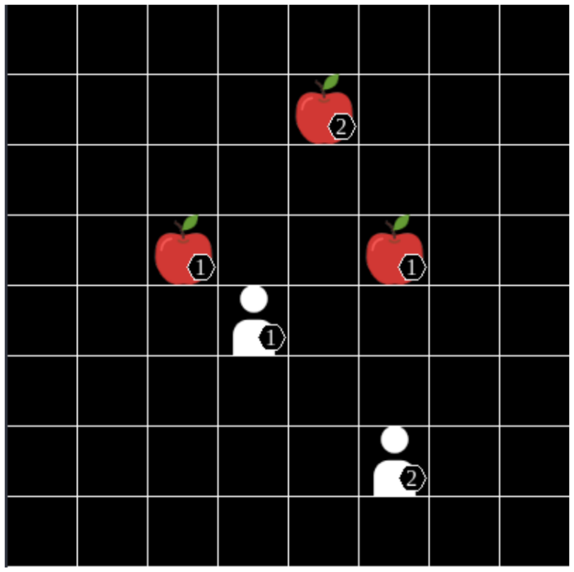

Figure 2.1: LBF Foraging-8x8-2p-3f example gridworld taken from Papoudakis et. al.[1]

2.2.4 Energy and the Grid

Demand side management

The addition of distributed energy resources (DERs) to the grid has brought about a host of challenges for utilities. DERs, especially those located at the grid edge, can be difficult to manage and control, leading to voltage fluctuations, reverse power flow, and degraded power quality [35–37]. As a result, utilities are turning to demand response (DR) techniques to help mitigate these issues.

DR methods can be classified into two categories: direct and indirect. Direct DR involves giving utilities direct control over heavy appliances like HVACs to increase or decrease load as needed. While effective, direct DR is limited in scope and can be invasive, leading to friction between utilities and customers [38–40]. Indirect DR, on the other hand, uses incentive signals like time-of-use pricing, critical peak pricing, real-time pricing, and dynamic pricing to influence customer behavior [38]. However, research has shown that customers tend to prioritize comfort over price signals, making indirect DR less effective when a high level of response is needed [39].

To address these challenges, utilities are turning to advanced metering infrastructure and smart grid technologies. These systems provide greater visibility and control over DERs, enabling more precise load management. For example, advanced metering infrastructure can provide real-time data on energy usage, allowing utilities to offer more accurate pricing signals and incentivize customer behavior that better aligns with grid needs. Smart grid technologies, like distributed energy resource management systems, can help balance the intermittent nature of renewable energy resources and ensure stable grid operation.

In addition to technological solutions, policy changes may also be necessary to encourage greater adoption of DR methods. Regulations that incentivize the installation of DERs with smart inverters that can provide grid support services could help alleviate some of the challenges posed by DERs at the grid edge.

Addressing the challenges of DER integration into the grid will require a multifaceted approach that combines technological innovation, policy changes, and collaboration between utilities and customers. With these solutions in place, we can ensure a stable and reliable energy grid that can meet the needs of both utilities and customers alike. However, due to the limitations of DR, interest in alternative control schemes is increasing.

Application of RL for DR

Over the last decade, there has been an uptake in the application of RL to DR[41]. RL is very well positioned for different kinds of DR due to its model-free nature and its ability to work as a control method, rather than just a predictive one.

2.2.5 Citylearn environment

CityLearn is an RL environment that allows for the study of demand response techniques using single-agent and multi-agent RL [42]. Citylearn is built on the popular OpenAi gym framework with multi-agent capabilities following the style that was set by Lowe et. al. [8]. CityLearn does not require any co-simulations with EnergyPlus or any other building energy simulators. The components of the CityLearn environment are shown in Figure 2.2.

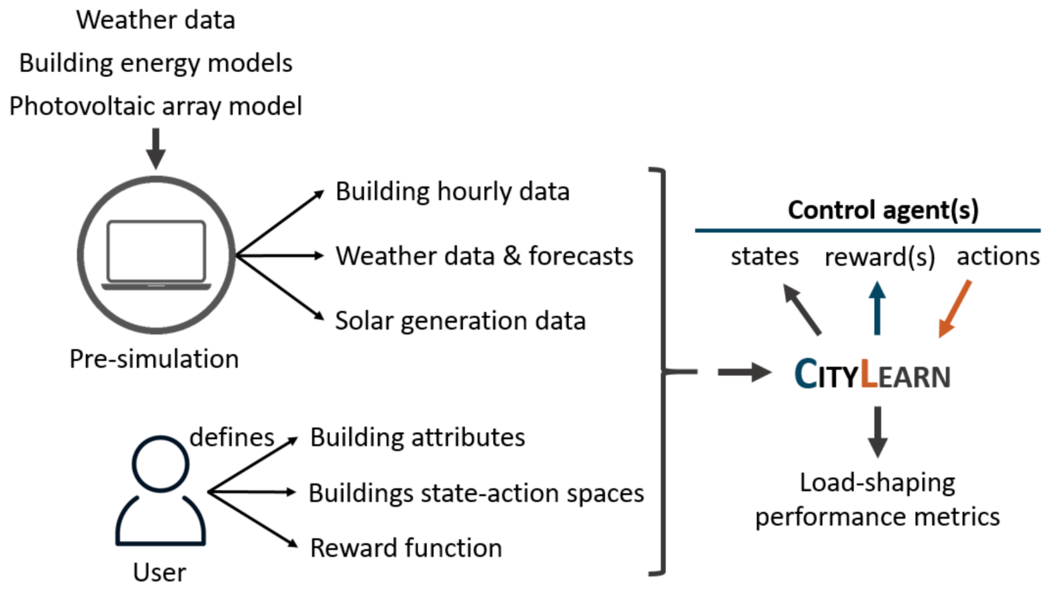

Figure 2.2: Flowchart of the CityLearn environment created by Vazquez-Canteli et. al. [42]

CityLearn contains energy models of a variety of dispatchable (controllable) building systems including air-to-water heat pumps and electric heaters. In addition, CityLearn contains pre-computed energy loads of a set of buildings which include additional building systems such as space cooling, appliances, dehumidification, domestic hot water (DHW) and solar generation. CityLearn operates on a one-hour timestep frequency, allowing agents to select control actions and receive rewards and observations every hour. The environment automatically constrains the actions of the agent to ensure that the energy supply of the devices is large enough to guarantee the building's energy demands are met at every timestep. The RL agents are in charge of cooling/heating storage in the buildings and the dispatch of this energy. CityLearn also contains a backup energy controller that ensures that the systems prioritize the satisfaction of the building's energy demands before storing any additional energy.

This way, CityLearn guarantees that at any point the heating and cooling energy demands of the building are satisfied regardless of the actions of the agent controller that is being studied.

Models in CityLearn are trained using data from 17 zero-net-energy single-family homes in the Sierra Crest zero-net energy community in Fontana, California. This data was collected as part of a study exploring high-penetration solar PV generation and on-site energy storage as a part of the California Solar initiative program.

Environment Dynamics

Observations

CityLearn provides a large number of observations to the agents that come from both inside and outside the building. The observations that are provided to the agents are as follows:

- *Month* - This is the month of the year as an integer that ranges from [1-12]
- *Hour* - This is the hour of the day as an integer that ranges from [1-24]
- *day_type* - EnergyPlus day type. This value ranges from [1-8] as follows (1 = Sunday, 2 = Monday ... , 7 = Saturday , 8 = Holiday)
- *daylight savings status* - This indicates if daylight savings is being used and is an integer in the range [0,1].
- *indoor_average_temperature* This is the average indoor temperature of the building across all thermal zones weighted by the floor area. This is a float and its units are in [°C].
- *average_unmet_cooling_setpoint_difference* - The unmet cooling setpoint difference (UCSD) is the difference between the thermal zone temperature and its setpoint. The average of the building is calculated across all building zones and weighted by their floor area. This is a float value and its units are in [°C].

- *indoor_relative_humidity* - Average indoor relative humidity of the building environmental zones weighted by their floor area. This is a float and its units are a percentage.

- *equipment_electric_power* - This is the electic power consumed by all appliances in the building excluding the HVAC system. This is a float value and its units are [kWh].

- *heating_energy_for_DHW* - Thermal heating energy consumed for DHW. This value is a float and its units are in [kWh].

- *total_cooling_load* - This value is the total thermal energy demand for cooling in the building. It is a float value and its units are [kWh].

These observations are precomputed and are simply sent to the agents as a time series. This makes training and evaluating models in CityLearn fast as there is no need to calculate any building parameters other than those associated with the agent.

Actions

The actions that the agent is able to perform comes in the form of a float value that is bound in the range [-1.0, 1.0]. This value represents the percentage of the storage capacity that is to be discharged (negative values) or charged (positive values) during the hour.

These actions are able to direct the charging and discharging of a few different active thermal energy and electrical energy storage devices. Due to the realities of thermal energy storage, the actions that control thermal energy storage devices are bounded between $-\frac{1}{C}$ and $\frac{1}{C}$ where C is defined as a multiple of the maximum thermal energy consumption by the building in any given hour. Unlike thermal energy storage devices, battery storage devices do not have any load-based limits of operation and are limited only by the battery model itself.

2.2.6 CityLearn Device Models

This section contains outlines for the mathematical models that are used in CityLearn. For more information on the models, please see the work by Vazquez et. al. that explains the environment in full [42].

Heat Pump:

The heat pump model that is implemented in CityLearn is an air-to-water heat pump that is defined by the following coefficients of heating and cooling:

$$COP_c = \eta_{tech} * \frac{T^c_{target}}{T_{outdoorair} - T^C_{target}} \tag{2.5}$$

$$COP_h = \eta_{tech} * \frac{T^h_{target}}{T^h_{target} - T_{outdoorair}} \tag{2.6}$$

The amount of thermal energy that is provided by the heat pump is given by the following equation:

$$Q^{hp}_{t+1} = C * (SOC_{t+1} - SOC_t) + Q^{dem} \tag{2.7}$$

The equation relating the thermal energy generated by the heat pump and the electrical energy needed to do so is:

$$E^{hp}_t = \frac{Q^{hp}_t}{COP_t} \tag{2.8}$$

Electric Heater:

The electric heater consumes electricity E_{heater} to provide heating energy Q_{heater} and this relationship is defined in CityLearn by the following equation:

$$E^{heater}_t = \frac{Q^{heater}_t}{\eta_{eh}} \tag{2.9}$$

Where η_{eh} is the heater efficiency and is usually > 0.9.

Thermal Storage:

CityLearn has two types of thermal storage available to buildings: Domestic hot water (DHW) and chilled water tanks. Both of these thermal storage devices convert electric energy into thermal energy through either heat pumps or electric heaters. These relationships are defined by the following equations:

$$Q_{out}^{sup} := Q_t^{hp}; Q_{out}^{sup} := Q_t^{heater} \tag{2.10}$$

state of charge for the thermal storage is calculated using the following equation:

$$SoC_{t+1} = SoC_t * (1 - e_{loss}) + Q_{in}^{sto} - Q_{out}^{sto} \tag{2.11}$$

where e_{loss} is a thermal loss coefficient over the hour of each timestep. The relationship between the stored thermal energy and the supplied thermal energy relies on the round trip efficiency of the storage device η_{eff} and is given by the following equations:

$$Q_{out}^{sup} = \frac{Q_{in}^{sto}}{\sqrt{\eta_{eff}}} \tag{2.12}$$

$$Q_{out}^{sto} = -\frac{Q^{dem}}{\sqrt{\eta_{eff}}} \tag{2.13}$$

Electrical Storage:

The battery capacity is defined in kWh and its nominal power is defined in kW. The battery model contains a capacity loss coefficient *closs* that is the ratio of the battery capacity lost in each discharge cycle, expressed in the units $c_{loss} = \frac{1}{cycle}$.

The relationship between the initial battery capacity and the current battery capacity is given through the following relationship:

$$C_{new} = d * \#\_of\_cycles * C_0 = c_{loss} * C_0 * \frac{|E_{in|out}|}{2 * C} \tag{2.14}$$

where C_0 is the initial capacity, C is the current capacity and $E_{in|out}$ is the amount of energy that has been charged or discharged.

The battery model also contains a *capacity_power_curve* attribute that allows for the modeling of the relationship between maximum charging power and state of charge.

In addition, the model features a round trip efficiency η_{eff} that can also be set as either a single value or as a function of the charging/discharging rate.

A loss coefficient that represents the ratio of energy loss in the battery while being in stand-by is defined as e_{loss}.

The change in SoC for each timestep can therefor be defined as:

$$SoC_{t+1} = SoC_t * (1 - e_{loss}) + E_{in|out} \tag{2.15}$$

$E_{in|out} < 0$ when the battery is being discharged and $E_{in|out} > 0$ when the battery is being charged.

Solar Photovoltaics (PV):

CityLearn uses pre-simulated datasets that were obtained using SAM and are available in the CityLearn Github repository [43]. Buildings have user-defined PV capacity which is used to scale the pre-simulated values to match.

2.2.7 CityLearn Key Performance Indicators (KPIs)

Agent performance in the CityLearn environment is evaluated with seven different metrics that are to be minimized. These metrics are split into individual building metrics and district-level metrics that take into account all buildings in the district. The building metrics are energy consumption, electricity price and carbon emissions. The district metrics include zero net energy, average daily peak, ramping and load factor. The following sections outline the metrics in more detail:

Energy Consumption

Electricity consumption is defined by the following equation:

$$D = \sum_{h=0}^{n-1} max(0, E_h^{building}) \tag{2.16}$$

The electricity consumption KPI's objective is to minimize the energy consumed by the building from the grid without encouraging profits from the excess generation. As such, it is the sum of the non-negative net building energy consumption. $E_h^{building}$. Net building energy consumption is defined with the following equations:

$$E_h^{building} = E_h^{non-shiftable} = E_h^{main} - E_h^{battery} + E_h^{PV} \tag{2.17}$$

$$E_h = \sum_{m}^{60} \frac{P_m}{60} \tag{2.18}$$

P_m is the power measurement per minute [kW]. $E^{non-shiftable}$ is the load that remains to be satisfied regardless of battery output and PV generation.

Electricity Price

Electricity price is defined by the following equation:

$$C = \sum_{h=0}^{n-1} max(0, E_h^{building} \times T_h) \tag{2.19}$$

The Electricity Price KPI's objective is to minimize the cost of energy used by the building from the grid. As such it is only concerned with the non-zero and non-negative $E_h^{building}$ as this is consumption.

Carbon Emissions

Carbon emissions are defined with the following equation:

$$G = \sum_{h=0}^{n-1} max(0, E_h^{building} \times O_h) \tag{2.20}$$

The carbon emissions KPI's objective is to minimize the building-level carbon emissions $[kg_{CO_2e}]$produced from the electricity consumption of the building. O_h is the carbon intensity for hour h measured in $[kg_{CO_2e}/kWh]$.

Zero Net Energy

Zero net energy is defined with the following equation:

$$Z = \sum_{h=0}^{n-1} E_h \tag{2.21}$$

Zero net energy is the sum of both negative and positive values of $E_h^{buildings}$

Average Daily Peak

The average daily peak is defined with the following equation:

$$P = \frac{\sum_{d=0}^{364} \sum_{h=0}^{23} max(E_{24d+h}^{district}, ..., E_{24d+23}^{district})}{365} \tag{2.22}$$

The objective of the average daily peak KPI is to minimize the mean of the daily maximum energy consumption of all buildings in the district $E_{district}$.

Ramping

The district ramping is defined by the following equation:

$$R = \sum_{h=0}^{n-1} |E_h^{district} - E_{h-1}^{district}| \tag{2.23}$$

The ramping KPIs objective is to lower the absolute difference between consecutive $E_h^{district}$ values. The objective of this KPI is to smooth out any abrupt changes in grid load between timesteps. Ramping is of particular concern with advances in DER technology as a high R-value means that there may be a large and abrupt strain on grid infrastructure which can result in blackouts if there is a supply deficit.

1 - Load Factor

1 - Load factor is a measure of the efficiency of district electricity consumption given by the following equation:

$$1 - L = \left(\sum_{m=0}^{11} 1 - \frac{(\sum_{h=0}^{729} E_{730m+h}^{district}) \div 730}{max(E_{730m}^{district}, ..., E_{730m+729}^{district})} \right) \div 12 \tag{2.24}$$

L is the average ratio of monthly average and peak $E_h^{district}$ and is bounded between 0 (very inefficient) and 1 (highly efficient). This KPI seeks to minimize 1 - L to maximize the efficiency of district energy usage.

Chapter 3

Investigating the Effects of Varying Reward types on CLDE Algorithms

This chapter presents a detailed discussion of two research articles, authored during this degree, investigating the dynamics of agent coordination in cooperative tasks [2, 3].

The primary objective of this study is to evaluate the impact of altering the reward structure in the LBF task - from a joint to an individual reward - on the coordination among agents. The LBF task, originally designed as a cooperative activity, takes on a mixed nature under the individual reward context since agents can opportunistically collect food without waiting for their peers, especially if they hold a higher level.

When the reward is changed from joint reward to individual reward, the LBF task embodies an SSD, specifically an instance of "the tragedy of the commons". The hypothesis that was tested posits that individual rewards might lead to more efficient policies increasing performance at the cost of introducing higher variance during learning, thereby making training harder and possibly longer. Hence, the implementation of CLDE is suggested to enhance coordination during training.

To validate these assumptions, empirical tests were conducted to assess the performance of various CLDE and independent algorithms, both with and without the implementation of a joint reward all while varying episode length. The outcomes of these tests are documented and discussed in the ensuing sections.

3.1 Introductory Investigations

This section outlines the initial research direction that was followed and culminated in the first published work [2]. It details the methods and procedures for the study and then the preliminary findings as well as the reasons requiring further in-depth study that lead to [3].

3.1.1 Experimental Design

The experimental design of the initial study was based on the work of Papoudakis et. al. and the aim was to use their data as comparison values between joint and individual rewards [1]. All the algorithms that were used in this work were also used by Papoudakis et. al. and the hyperparameters were taken directly from Papoudakis et. al.. To identify if any change in performance in algorithms was a result of the interaction of CLDE mechanisms and individual reward, we incorporated the independent version of each CLDE algorithm in the study.

Independent Learning algorithms (IL):

Algorithmically identical to their single-agent counterparts, IL algorithms condition independently on an agent's local information in a decentralized manner, without access to joint information from other agents. In this work, we employ three different IL algorithms. Independent Q Learning (IQL) is based on the popular DQN algorithm [23]. Independent Proximal Policy Optimization (IPPO) based on the successful Proximal policy optimization (PPO) algorithm [44]. Independent Synchronous Advantage Actor-Critic (IA2C) is a synchronous variant of the A3C algorithm [6].

Centralized learning decentralized execution (CLDE)

The CLDE algorithms used in this work include both centralized policy gradient and value function decomposition methods. Chosen centralized policy gradient algorithms are MAA2C and MAPPO [7]. MAA2C and MAPPO are both multi-agent versions of their independent counterparts with a joint critic conditioning on all agent observations. Value decomposition methods used were VDN [10] and its more contemporary counterpart QMIX [9]. Value decomposition methods train a factored joint state-action value function using the standard DQN algorithm. VDN decomposes the value function as the sum of individual value functions while QMIX employs a hyper-network conditioned on state information to factor the state-value function.

LBF Environment Scenarios

Tasks selected for this study are: *8x8-2p-2f-coop*, *2s-8x8-2p-2f-coop*, *10x10-3p-3f* and *2s-10x10-3p-3f*. *2s* indicates an observation radius of 2 squares, allowing the study of partial observability.

These tasks were selected because most algorithms were able to achieve maximum episode return in the joint reward context, and it is reasonable to assume that the same algorithms would also be able to maximize the episode return in the individual reward setting. To evaluate algorithm performance, we calculate the average returns achieved throughout all evaluation windows during training and a 95% confidence interval across five seeds. Hyperparameter tuning was not performed for this experiment to stay consistent with the hyperparameters used for the LBF benchmark [1].

3.1.2 Preliminary Results and Discussions

The preliminary training results from the initial study are shown in Figures 3.1 and 3.2. As a general observation, the change to a cooperative environment decreases the performance of all tested algorithms. Fully observable methods that rely on any form of estimating the action value function seem to improve in the cooperative setting.

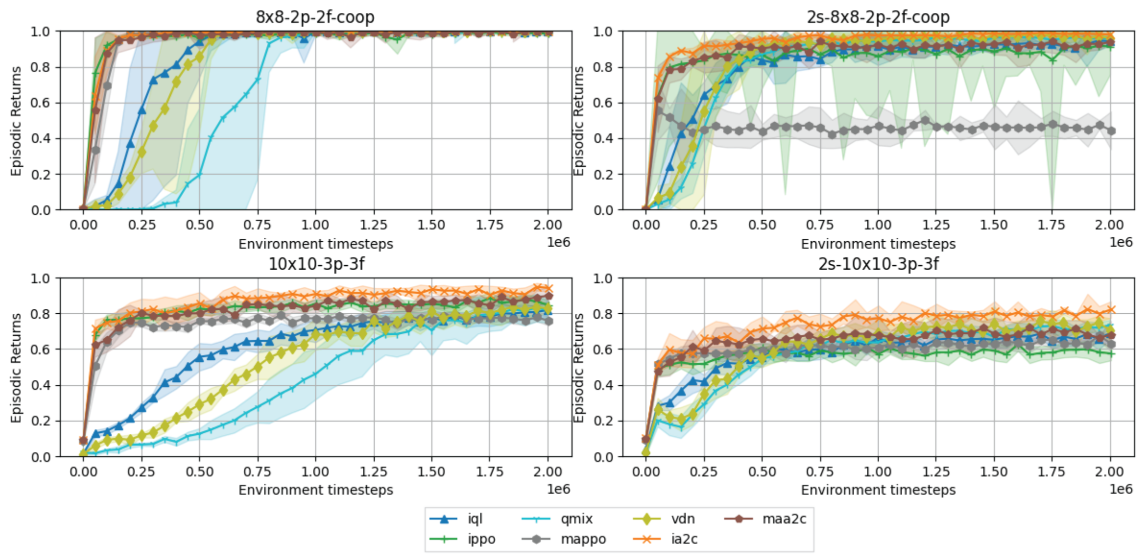

Figure 3.1: Episodic returns of all algorithms on CLBF showing the mean and 95% confidence interval over three different seeds. These results show the performance of algorithms on the LBF environment using individual rewards. Hyperparamters used were taken from Papoudakis et. al. [1].

The algorithms IQL, VDN and QMIX all show an increase in the speed of learning when the reward is individual. While it can be argued that the improvement in the IQL convergence rate is intangible, the improvements in VDN and QMIX are notable. When figures 3.1 and 3.2 are compared, both VDN and QMIX show signs that the training speed is reduced by at least 1/4 with some earlier training instances being reduced by 1/2. This convergence speed-up is best seen in the QMIX algorithm in the *8x8-2p-2f-coop* scenario where convergence is shortened from 2 million timesteps in the fully cooperative setting to just under 1 million timesteps. This increase in convergence rate would suggest that the use of individual rewards over joint rewards has benefits for certain algorithms.

Problems With the Experimental Design

Since hyperparameter optimization was not performed for this experiment, it may be possible that tuned agents could reclaim losses in performance. The lack of hyper-parameter optimizations also makes it unclear if the slight reduction in final mean

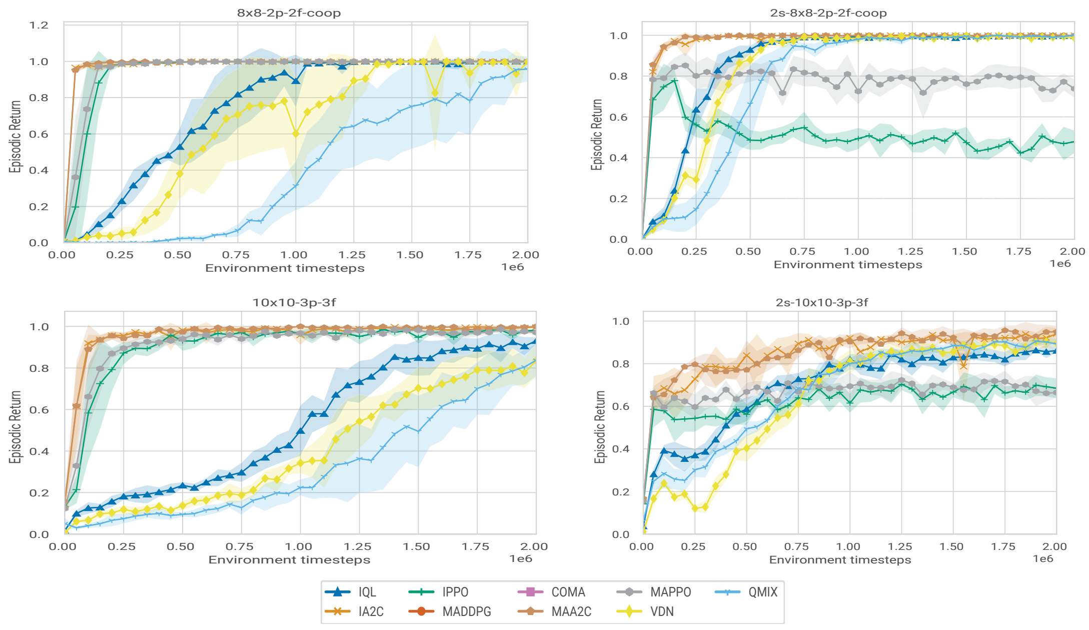

Figure 3.2: Image is taken from Papoudakis et al.[1] COMA and MADDPG traces have been removed and only the relevant scenarios have been kept. These results show the performance of algorithms on the LBF environment with joint reward.

return is due to the change in the reward function or unoptimized hyperparameters. In addition, the data that was used for comparison is taken directly from the publication by Papoudakis et. al. and was not generated through empirical means. Without generating our own episode data using the hyperparameters reported or accessing the data that was used by Papoudakis et. al. it is uncertain if these results are valid.

3.2 In-depth Empirical Evaluation

To address the deficiencies of the initial exploratory design we conduct an in-depth investigation. We varied two variables, reward function and episode length as well as conducting a new hyperparameter sweep using both reward functions to determine optimal hyperparameters for each reward function. Episode length was varied between the reported value of 25 used by Papoudakis et.al [1] and 50 which is the environment's default episode length. We perform two separate hyperparameter tuning, one for each reward type, adhering to the hyperparameter tuning protocol included

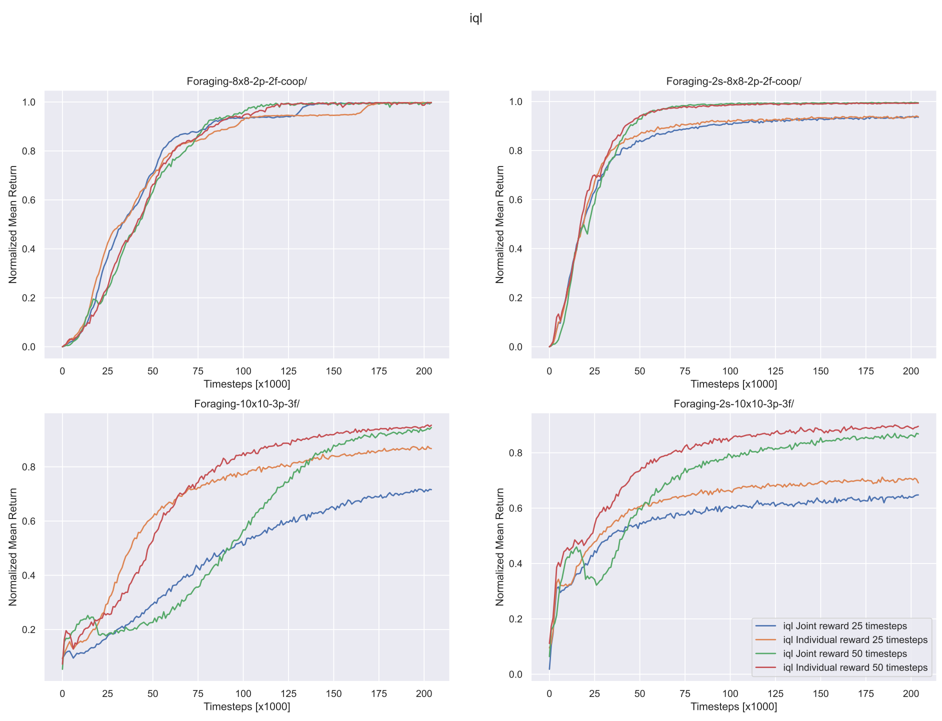

Figure 3.3: Figure comparing mean returns during training for IQL algorithm across both reward types and increased episode horizon.

in Papoudakis et. al. [1].

All other experimental parameters are taken from Papoudakis et. al. [1] and we encourage readers to look into this work for further details.

3.3 Comprehensive Analysis

We compare IQL, IA2C, IPPO, MAA2C, MAPPO, VDN and QMIX and report the mean returns and max returns achieved by algorithms using individual reward in tables 3.2 and 3.1 respectively. The mean returns and max returns of algorithms using joint reward are reported in tables 3.4 and 3.3 respectively. We include tables for the increased episode length (50 timesteps) in the appendix.

We generally observe that in the individual reward case, QMIX can consistently achieve the highest maximal return value in all scenario. In terms of the highest mean returns, QMIX can outperform IPPO in the partially observable scenarios. In the

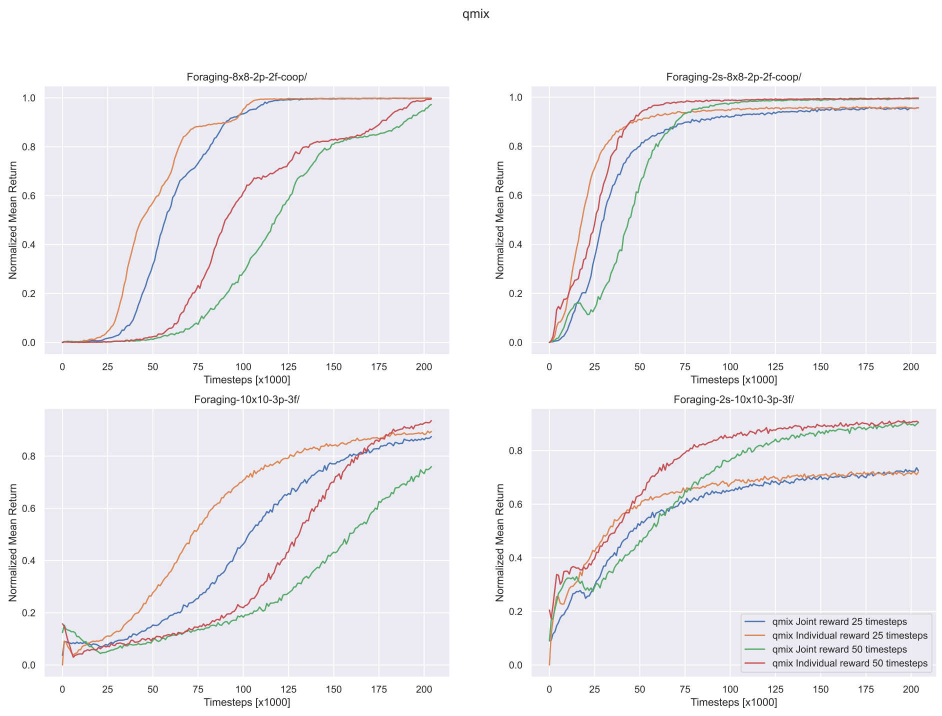

Figure 3.4: Figure comparing mean returns during training for QMIX algorithm across both reward types and increased episode horizon.

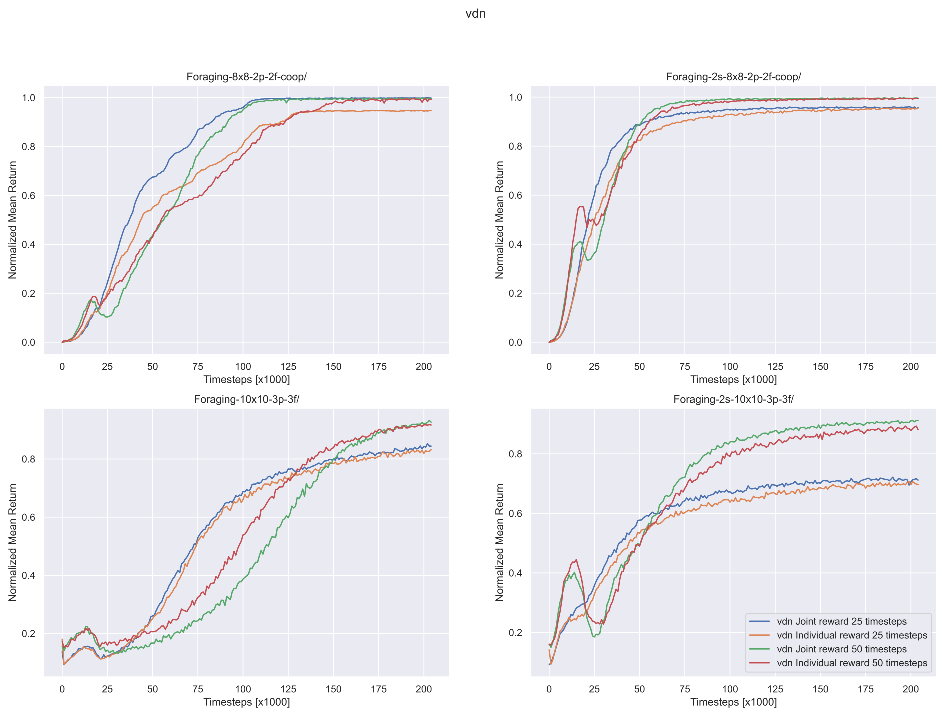

Figure 3.5: Figure comparing mean returns during training for VDN algorithm across both reward types and increased episode horizon.

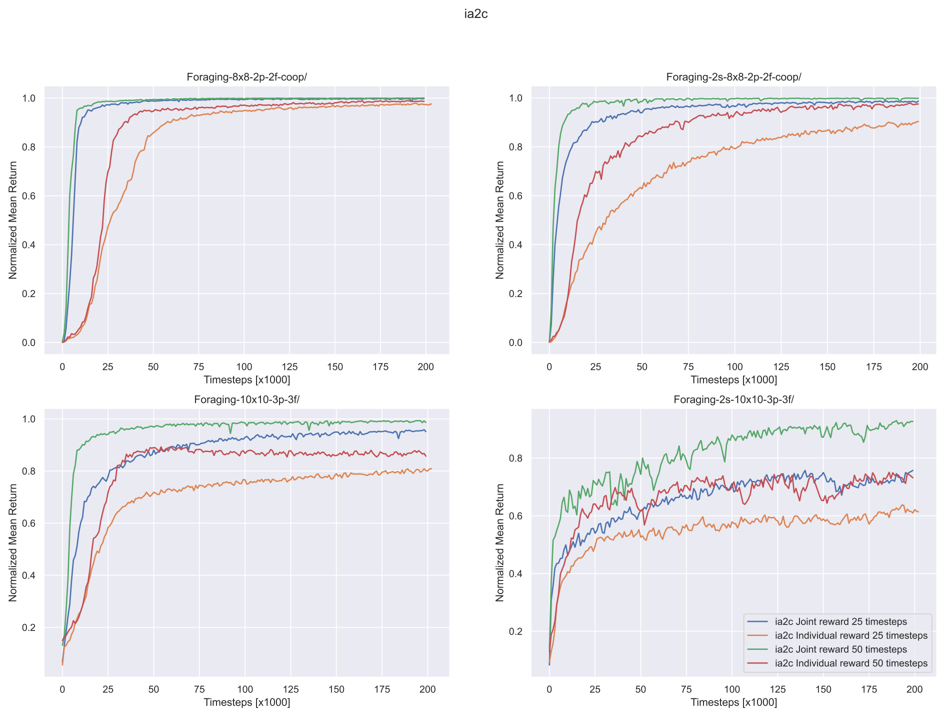

Figure 3.6: Figure comparing mean returns during training for IA2C algorithm across both reward types and increased episode horizon.

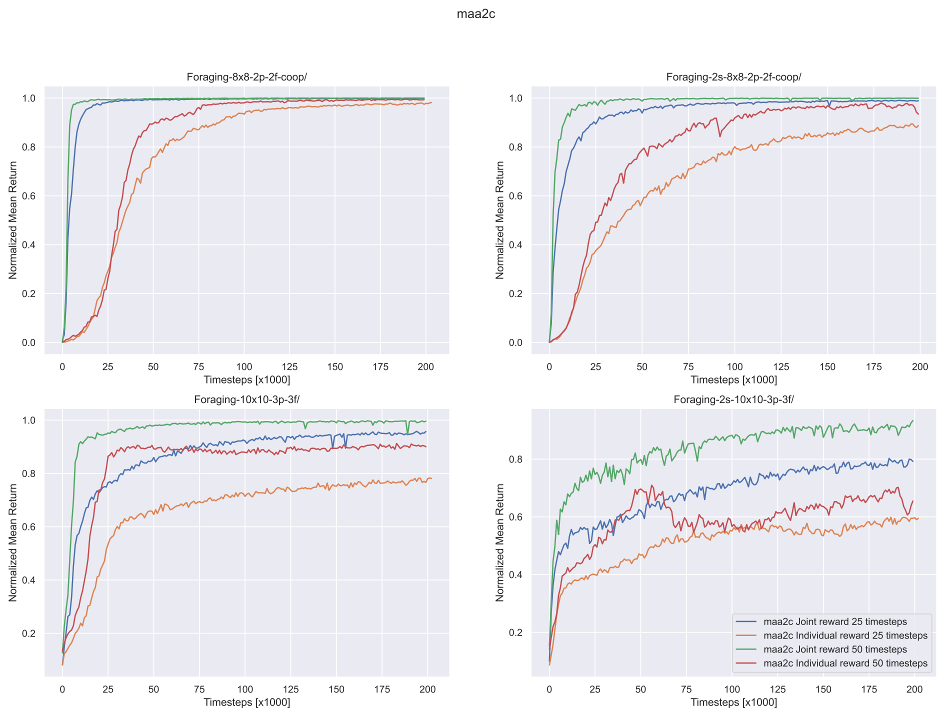

Figure 3.7: Figure comparing mean returns during training for MAA2C algorithm across both reward types and increased episode horizon.

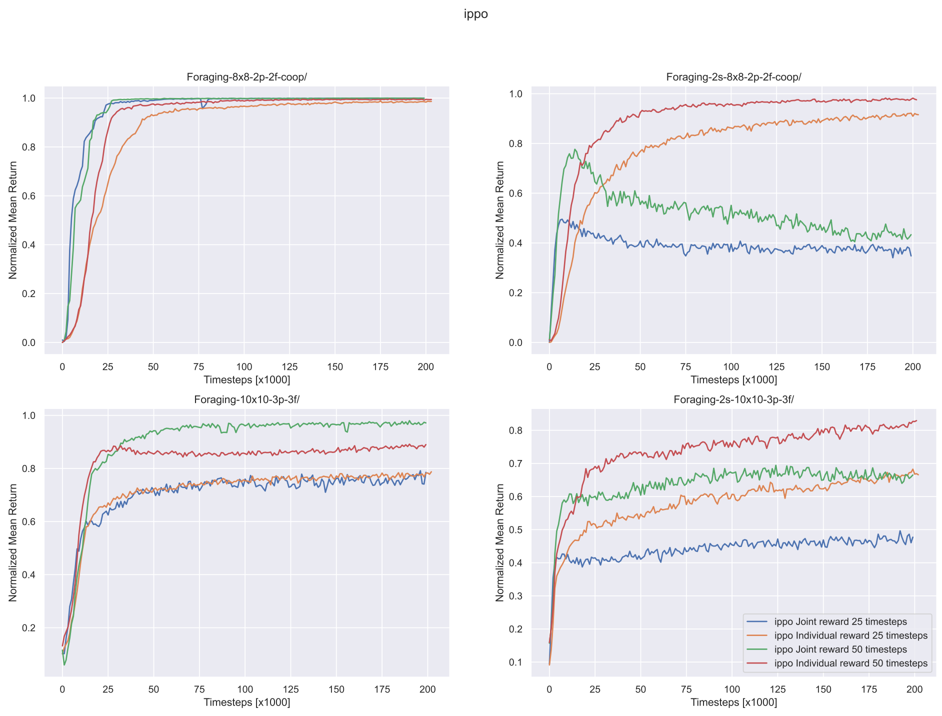

Figure 3.8: Figure comparing mean returns during training for IPPO algorithm across both reward types and increased episode horizon.

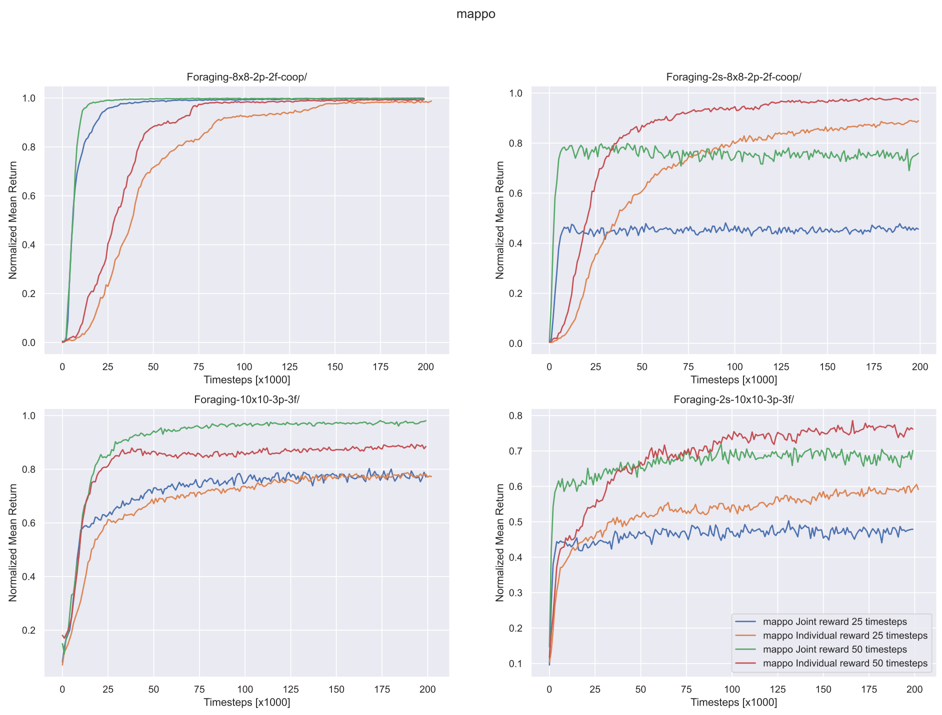

Figure 3.9: Figure comparing mean returns during training for MAPPO algorithm across both reward types and increased episode horizon.

Scenario	IQL	IA2C	IPPO	MAA2C	MAPPO	VDN	QMIX
8x8-2p-2f-c	**1.0±0.0**	**1.0±0.0**	**1.0±0.0**	**1.0±0.0**	**1.0±0.0**	**1.0±0.11**	**1.00±0.00**
8x8-2p-2f-2s-c	0.97 ± 0.0	0.94 ± 0.01	0.95 ± 0.01	0.93 ± 0.01	0.93 ± 0.01	0.97 ± 0.0	**0.98±0.00**
10x10-3p-3f	0.94 ± 0.02	0.86 ± 0.01	0.88 ± 0.04	0.85 ± 0.03	0.86 ± 0.02	0.93 ± 0.04	**0.95±0.02**
10x10-3p-3f-2s	0.75 ± 0.01	0.71 ± 0.02	0.76 ± 0.02	0.7 ± 0.02	0.73 ± 0.07	0.74 ± 0.01	**0.77±0.01**

Table 3.1: Maximum returns and 95% confidence interval of algorithms using individual reward in selected scenarios over 10 seeds after a hyperparameter search was completed. Bolded values indicate the best result in a scenario.

Scenario	IQL	IA2C	IPPO	MAA2C	MAPPO	VDN	QMIX
8x8-2p-2f-c	0.78 ± 0.08	0.82 ± 0.02	**0.84±0.07**	0.78 ± 0.05	0.77 ± 0.07	0.7 ± 0.08	0.75 ± 0.04
8x8-2p-2f-2s-c	0.83 ± 0.01	0.71 ± 0.01	0.77 ± 0.01	0.68 ± 0.01	0.69 ± 0.03	0.81 ± 0.01	**0.86 ± 0.01**
10x10-3p-3f	0.68 ± 0.02	0.7 ± 0.02	**0.72 ± 0.03**	0.66 ± 0.03	0.69 ± 0.02	0.55 ± 0.04	0.58 ± 0.03
10x10-3p-3f-2s	**0.62±0.0**	0.55 ± 0.02	0.58 ± 0.02	0.51 ± 0.02	0.53 ± 0.06	0.57 ± 0.01	**0.62±0.01**

Table 3.2: Mean return values and 95% confidence interval of algorithms using individual reward in selected scenarios over 10 seeds after a hyperparameter search was completed. Bolded values indicate the best result in a scenario.

Scenario	IQL	IA2C	IPPO	MAA2C	MAPPO	VDN	QMIX
8x8-2p-2f-c	**1.0±0.0**	**1.0±0.0**	**1.0±0.0**	**1.0 ±0.0**	**1.0±0.0**	**1.0±0.0**	**1.0±0.0**
8x8-2p-2f-2s-c	0.97 ± 0.01	**1.0±0.0**	0.63 ± 0.02	**1.0±0.0**	0.56 ± 0.02	0.98 ± 0.0	0.97 ± 0.0
10x10-3p-3f	0.89 ± 0.08	**0.99± 0.01**	0.89 ± 0.02	0.98 ± 0.01	0.9 ± 0.24	0.9 ± 0.03	0.91 ± 0.02
10x10-3p-3f-2s	0.7 ± 0.01	**0.84±0.04**	0.56 ± 0.01	**0.85±0.01**	0.58 ± 0.01	0.77 ± 0.01	0.76 ± 0.04

Table 3.3: Maximum returns and 95% confidence interval of algorithms using joint reward in selected scenarios over 10 seeds after a hyperparameter search was completed. Bolded values indicate the best result in a scenario.

Scenario	IQL	IA2C	IPPO	MAA2C	MAPPO	VDN	QMIX
8x8-2p-2f-c	0.77 ± 0.08	0.96 ± 0.01	0.96 ± 0.01	**0.97± 0.01**	0.96 ± 0.02	0.78 ± 0.04	0.69 ± 0.04
8x8-2p-2f-2s-c	0.82 ± 0.01	**0.94±0.01**	0.39 ± 0.02	**0.94±0.0**	0.45 ± 0.02	0.84 ± 0.0	0.79 ± 0.01
10x10-3p-3f	0.47 ± 0.07	**0.88±0.02**	0.71 ± 0.03	0.87 ± 0.02	0.59 ± 0.21	0.56 ± 0.03	0.46 ± 0.04
10x10-3p-3f-2s	0.56 ± 0.01	0.67 ± 0.05	0.44 ± 0.0	**0.69±0.02**	0.46 ± 0.0	0.6 ± 0.01	0.56 ± 0.05

Table 3.4: Mean return values and 95% confidence interval of algorithms using joint reward in selected scenarios over 10 seeds after a hyperparameter search was completed. Bolded values indicate the best result in a scenario.

joint reward case, the majority of the results are in line with those reported in [1] however we note that the average return results for QMIX are much higher with our hyperparameters. We go into more detail regarding these results in Appendix A.1.

Comparing joint reward performance to individual reward performance, we note that the effects of the reward are not easily predictable. Centralized critic algorithms are evenly split in performance, with MAPPO performing better with individual reward while MAA2C's performance suffers. This is paralleled by the independent versions of MAPPO and MAA2C. Value factorization algorithms are also divided, with QMIX performance becoming the top-performing algorithm across the tested scenarios. VDN however, sees an incredible drop in performance when using joint reward. Finally, IQL performance when using individual rewards is relatively unaffected in the simpler 8x8 scenarios but decreases in the larger scenarios.

3.4 Detailed Examination

This section details discussion points regarding the performance of each algorithm when comparing individual and joint rewards.

3.4.1 Independent Algorithms

IQL: Our results show that IQL achieves increased mean return values and max return values when using individual rewards. Our results also show that IQL experienced a reduction in loss variance when using individual rewards. Since IQL is an independent algorithm, the joint reward is the only source of information from other agents. Seeing as IQL does not observe the other agents specifically, our results suggest that the joint reward seems to increase the variance in the loss function by the nonzero probability of agents receiving reward at any timestep, as discussed earlier. The reduction in variance in loss function allows for better policies to be learned by each individual agent, and this is further evidenced by the decrease in variance and simultaneous increase in the mean of the absolute td error that agents have in the

CLBF experiments.

IPPO: IPPO is able to use the individual reward signal to achieve higher mean return and max returns in all scenarios except 8x8-2p-2f-coop. We believe that this is in most part due to the decrease in variance that is observed in the maximum policy values that are learned. Our results show that the TD error that is generated from multiple different individual rewards seems to be higher and more varied than the TD error that is generated from a joint reward. This variance seems to permeate through the loss function allowing the algorithm to continue discovering new higher policies through training. It seems that joint reward causes the TD error to start out strong and quickly the algorithm finds a policy (or set of policies) that has the maximal chances of achieving reward at all timesteps. This is a local minima but the error is too small for policies to escape the minima.

IA2C: IA2C suffers from the increase in variance in individual reward. We note evidence of divergent policy behavior in several metrics most notably the critic and policy gradient loss. The critic is sill able to converge however the policy gradient loss diverges quite a bit more in the individual reward case. It seems that a joint reward is necessary to help coordinate the actor's behavior.

3.4.2 Value Factorization Algorithms

Performance in value factorization algorithms is varied and seems to depend on the underlying algorithm used. In general, Qmix responds very well to individual reward, whereas VDN sees performance drops when using individual rewards. **VDN**: VDN with individual reward has a very quick reduction in loss values. Our data suggest that when using individual rewards, VDN converges prematurely on suboptimal policies, causing the observed reduction in mean and max return. This may be because VDN does not incorporate any state information into the creation of the joint value function. The authors seem to have relied on the information that is contained in the joint reward to help guide the coordination of agents through the learned joint value function. With the use of individual reward, the joint action value function simply optimizes for the first policy that serves to achieve higher returns without any regard to the coordination of agents and without guiding the agents to find optimal policies. When we examined the effects of individual reward on variance in the signal, we found that there was not enough significance in the variance to determine any cause-effect relationship.

QMIX: Our results show that when individual reward is used with Qmix return mean and max return values are increased. When comparing joint reward to independent reward, independent reward shows signs of faster convergence in loss and gradient norms. Qmix's combination of monotonicity constraint and global state information in its hypernetwork seems to be able to find coordinated policies when using individual rewards that achieve higher returns than those found when using joint rewards. By leveraging the global state information during training, Qmix the improvement that Qmix shows is significantly higher in the partially observable scenarios, where the increased information builds stronger coordination between agents. Qmix shows significant improvements in variance in all signals that were tested when using individual rewards. These results are across most of the scenarios tested and were consistent between the different episode lengths. It would seem that for Qmix, the improvement in variance is notable in comparison to the other algorithms.

3.4.3 Centralized Critic

Performance in centralized critics is varied and seems to depend on the underlying algorithm used.

MAA2C The increase in information that is imparted by MAA2C's centralized critic seems to not be enough to counter the increase in variance that is caused by individual rewards. When using joint rewards, the critic can converge and can guide the actor policies to find optimal values relatively quickly which is best demonstrated by the convergence of the TD error. When using individual rewards, there seems to be too much variance for the critic to be able to converge quickly. It has been shown that simply adding a centralized critic to an actor-critic MARL algorithm with the hopes of decreasing variance in the agent learning is not necessarily true and will increase the variance seen by actors [11]. It seems that in MAA2C, using the joint reward to decrease the variance seen by the critic is a good way of increasing performance. We do however note that when we increased the episode length, the individual reward mean and max returns did continue to increase, however it does not show any evidence of rapid convergence. It seems more investigations are required into the effects of increasing the episode length to determine if joint reward has a bias component.

MAPPO

Similar to IPPO, MAPPO performs better when using individual rewards than when using joint rewards. MAPPO's centralized critic does not seem to be able to prevent the critic from converging prematurely. Centralized critics have been shown to increase variance [11] however our results show that the increase in variance of the critic loss is not enough. Just as in IPPO, the critic converges within 100 episodes when using joint rewards. This corresponds to the majority of the gains in return, which seems to indicate some local minima are found by the algorithm.

Chapter 4

Application of CLDE algorithms to CityLearn 2022 challenge

4.1 Motivation

With the results from the works that comprised Chapter 3 showing that individual reward did have positive effects on certain algorithms, we move to a more complicated and topical task: building energy management.

Building energy management is a task that has been gaining popularity as more and more controllable electric devices have been installed in buildings (such as heat pumps and electric hot water tanks) and as energy prices increase or change to incentives individuals to use energy in off-peak hours (Such as time of use). RL and MARL have both been used in many research projects to control different aspects of building energy usage from electric car charging to the scheduling of various house appliances to buy energy on a dynamic energy market [41]. Notably, some of the most recent uses of RL and MARL include [45] where the authors used MARL to try to simultaneously decrease individual building energy needs and contribute positively to overall district grid health. This work used the CityLearn gym environment which was introduced in chapter 2.

4.2 CityLearn 2022 Challenge

CityLearn is used in an annual challenge known as the CityLearn challenge that was started in 2019. CityLearn 2022 focuses on the opportunity brought on by home battery storage devices and photovoltaics and how RL can be applied to these systems.

The challenge utilizes one year of operational electricity demand and PV generation data from 17 single-family buildings in the Sierra Crest home development in Fontana, California, that were studied for Grid integration of zero net energy communities.

Participants developed energy management agent(s) and their reward functions for battery charge and discharge control in each building with the goals of minimizing the monetary cost of electricity drawn from the grid, and the CO_2 emissions when electricity demand is satisfied by the grid.

The CityLearn 2022 challenge was conducted in three phases. In Phase I, participants' submissions were ranked based on a 5/17 buildings training dataset. In Phase II, the leaderboard reflected the ranking of participants' submissions based on an unseen 5/17 buildings validation dataset, as well as the seen 5/17 buildings dataset, with the training and validation dataset scores carrying 40% and 60% weights, respectively. In Phase III, submissions were evaluated on the 5/17 buildings training, 5/17 validation, and remaining 7/17 test datasets, with the training, validation, and test dataset scores carrying 20%, 30%, and 50% weights, respectively.

In addition to the participants of the CityLearn 2022 challenge, some additional publications use the same dataset in CityLearn.

4.2.1 Datasets

The dataset used in the CityLearn 2022 challenge is from a case study of ZNE buildings that covered the time period from August 1, 2016, to July 31, 2017. The data is down-sampled from one minute to one hour and features data from 17 NZE homes equipped with batteries and solar generation.

Time	June-September		October-May	
	Weekday	Weekend	Weekday	Weekend
8 am - 4 pm	0.21	0.21	0.20	0.20
4 pm - 9 pm	0.54	0.40	0.50	0.50
9 pm - 8 am	0.21	0.21	0.20	0.20

Table 4.1: Time of use plan used in the CityLearn challenge 2022, adopted from Southern Californa Edison's TOU-D-PRIME rate.

4.2.2 Battery control task in CityLearn 2022

The CityLearn challenge 2022 focuses specifically on the task of DR through a TE mechanism [46]: Given a time-series input of the observations that are outlined in chapter 2, control the charging and discharging of a household battery to minimize the cost of electricity and the cost of carbon emissions. The costs of energy and emissions are defined as follows:

$$C_{energy} = \sum_{t=1}^{T} C_{energy}(t) = \sum_{t=1}^{T} p_t * (d_t - g_t + b_t)^+ \tag{4.1}$$

$$C_{emissions} = \sum_{t=1}^{T} C_{emission}(t) = \sum_{t=1}^{T} c_t * (d_t - g_t + b_t)^+ \tag{4.2}$$

Where t is the timestep representing an hour of time, $C_{energy}(t)$ is the cost of electricity and $C_{emissions}(t)$ is the cost of emissions. p_t and c_t are the energy and emissions pricing per rate per unit of net energy demand. d_t, b_t and g_t are the non-shiftable household energy demand, battery charging amount (discharging the battery causes this term to be negative) and the generation from the household's solar arrays respectively. The + superscript indicates that negative values are clipped to zero.

It is worth noting that minimizing emissions costs is not the same thing as minimizing emissions as the cost of emissions is not directly linked to the generation profile of energy on the grid at each timestep.

Scoring of algorithms is done through the normalized scoring used in the CityLearn 2022 challenge [46].

$$\hat{C}_{energy} = \frac{C_{energy}}{C^{noop}_{energy}} \tag{4.3}$$

$$\hat{C}_{emissions} = \frac{C_{emissions}}{C^{noop}_{emissions}} \tag{4.4}$$

$$\hat{C} = \frac{\hat{C}_{emissions} + \hat{C}_{energy}}{2} \tag{4.5}$$

These normalized scores normalized the battery control performance, $C_{emissions}$ and C_{energy}, over the performance of the household without battery, $C^{noop}_{emissions}$ and C^{noop}_{energy}. The performance of households without batteries was calculated by setting their b_t component to zero.

4.2.3 Changes to the CityLearn Dataset in CityLearn Challenge 2022

In the CityLearn challenge 2022 dataset, the houses do not include the following observations:

- Indoor Temperature [C]
- Average Unmet Cooling Setpoint Difference [C]
- Indoor Relative Humidity [%]

In addition, buildings 12 and 15 have been removed from the dataset in line with the work by Nweye et. al. and after confirming with the publisher of the dataset [45]. These buildings exhibit very abnormal consumption and generation patterns captured in Figures 4.1, 4.2 and 4.3.

4.3 Methods and Procedure

This section will comprehensively address several distinct topics: the Training Procedure, Hyperparameter Selection, Model Initialization, Reward Design, Observation Design, and Evaluation Metrics used. Each subsection will delve into its respective topic, detailing the methods and principles employed in the research process, thus offering a cohesive understanding of the utilized machine learning framework.

4.3.1 Training Procedure

The research methodology utilized in this study follows a specified framework: taking into consideration that the focal interest resides in the impact of CLDE during the training process, it was determined to confine the use of data to the initial phase of the CityLearn Challenge 2022. The chosen dataset encompasses only the first five houses. It is hypothesized that this delimited data range will suffice in offering initial validation of the potential benefits ascribed to CLDE.

Hyperparameter selection

To keep the comparison to [1] the same hyperparameter search protocol was used, which is outlined in table 4.2.

The hyperparameter search was performed as follows. Random search was used to select a small subset of seeds. In addition to the seeds, the hyperparameters that were found from the previous search on the LBF environment were also tested. This coarse hyperparameter search was used to produce these initial results and a more thorough hyperparameter search should be done for further studies and future work. The hyperparameters search combinations generated by the random search are reported in table 4.3. In addition, table 4.4 also contains the hyperparameters that were taken from the LBF environment.

Hyperparameters that were selected for the work are summarized in table 4.5

Hyperparameters	Values
Hidden dimension	64/128
Learning rate	0.0001/0.0003/0.0005
Reward Standardization	True/False
Network Type	FC/GRU
Evaluation Epsilon	0.0/0.05
Epsilon Anneal	50000/200000
Target Update	200(hard)/0.01(soft)
Entropy Coefficient	0.01/0.001
n-step	5/10

Table 4.2: Hyperparameter search protocol taken from [1]

Model Intialization

DRL model weight initialization is crucial for proper performance on tasks, even more important than for deep learning tasks. The initial model parameters have a very large impact on the initial state visitation. It is recommended that DRL models be initialized using a very small distribution for model weights and it is critical to initialize the final bias layer with care as it will determine how much of the action space is explored by your agent in the initial learning phase. If done properly, the policy will have the largest chance of being unbiased in the initial learning stages.

The model weights were initialized using a uniform distribution with the range [-0.001,0.001]. The final bias layer was initialized to 3 as recommended by a colleague, Daniel May[1].

[1]Daniel May competed in the CityLearn 2022 challenge and his recommendation was to initialize the bias to 3 for use with beta distributions as action encoders.

Hyperparameter	Search 1	Search 2	Search 3	Search 4
Hidden Dimension	64	64	128	128
Learning Rate	0.0005	0.0003	0.0005	0.0005
Reward Standardization	False	True	False	True
Network Type	FC	FC	FC	FC
Evaluation Epsilon	0.0	0.05	0.05	0.0
Epsilon Anneal	200000	50000	50000	50000
Target Update	0.01 (soft)	0.01 (soft)	200 (hard)	200 (hard)
Entropy Coefficient	0.01	0.01	0.01	0.01
N-Step	10	5	10	10

Table 4.3: Hyperparameters generated by random search for use in CityLearn hyperparameter search

Hyperparameter	IA2C	MAA2C	IPPO	MAPPO
Hidden Dimension	128	128	64	64
Learning Rate	0.0005	0.0005	0.0005	0.0005
Reward Standardization	True	True	True	True
Network Type	GRU	GRU	GRU	GRU
Evaluation Epsilon	-	-	-	-
Epsilon Anneal	-	-	-	-
Target Update	0.01 (soft)	200 (hard)	0.01 (soft)	200 (hard)
Entropy Coefficient	0.01	0.001	0.01	0.001
N-Step	5	10	5	10

Table 4.4: Hyperparameters included into CityLearn Hyperparmeter search from LBF experiments.

Hyperparameter	IA2C	MAA2C	IPPO	MAPPO
Hidden Dimension	128	64	64	64
Learning Rate	0.0005	0.0003	0.0005	0.0005
Reward Standardization	True	True	True	True
Network Type	GRU	FC	GRU	GRU
Target Update	0.01 (soft)	0.01(soft)	0.01 (soft)	200 (hard)
Entropy Coefficient	0.01	0.01	0.01	0.001
N-Step	5	5	5	10

Table 4.5: Hyperparameters selected for the CityLearn experiments

4.3.2 Reward Design

The reward used to train the DRL agents is inspired by Zhumabekov et. al. [47]. Several design considerations were taken into account when crafting the reward in order to preserve properties that are similar in nature to the LBF environment reward. These properties are non-zero reward values and sparsity to the reward. The reward used was comprised of two components, an individual component that depends only on the agent and a group component that is based on district performance. The individual component is defined by the following equations:

$$r_{individual}^{energy} = C_{energy}^{noop}(t) - C_{energy}(t) \tag{4.6}$$

$$r_{individual}^{emission} = C_{emission}^{noop}(t) - C_{emission}(t) \tag{4.7}$$

$$R_{individual} = r_{individual}^{energy} + r_{individual}^{emission} \tag{4.8}$$

The group component of the reward is exactly the same as the individual reward except taken over the whole district.

$$r_{district}^{energy} = \sum C_{energy}^{noop}(t) - \sum C_{energy}(t) \tag{4.9}$$

$$r_{district}^{emission} = \sum C_{emission}^{noop}(t) - \sum C_{emission}(t) \tag{4.10}$$

$$R_{district} = r_{district}^{energy} + r_{district}^{emission} \tag{4.11}$$

To preserve the non-zero property, the entire reward term was bound at zero. This also helped decrease the amount of non-zero rewards, increasing sparsity. The final reward function is given by:

$$R_t = max(R_{individual} + R_{district}, 0) \tag{4.12}$$

4.3.3 Observation Design

Zhumabekov et. al. [47] and Nweye et. al. [45] both used observational encoding in their works to make the task easier and the same was done in these experiments. Table 4.6 summarizes the different observations and the transformation for encoding.

Periodic Transformation Encoder

Temporal observations are converted into sine and cosine components to ensure continuity from the last hour, month or year to the next first hour, next month or next year. The encoding therefore takes the single observations of day, month or hour and turns them into 2 observations using the following equations:

$$observation_{sin} = sin(2\pi\frac{M}{N}), \qquad observation_{cos} = cos(2\pi\frac{M}{N}) \tag{4.13}$$

where M is the current integer value of the observation within it's range: month [1,12], Day [1,7], hour [1,24] and N is the largest value in the range: month 12, day 7 and hour 24.

Min-Max Norm Encoder

This encoder is used to transform observations to a value between x_{min} and x_{max}

$$x = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{4.14}$$

Observation	Unit	Transformation	Value Range
Temporal			
Day of Month	-	Periodic transformations	1, 2, ...,31
Month	-	Periodic Transformations	1, 2, ...,12
Hour	-	Periodic Transformations	1, 2, ...,24
Weather			
Outdoor Temp.	°C	Min-Max Norm.	[5.6; 32.2]
Outdoor relative humidity		Min-Max Norm.	[10.0;100.0]
Dir. Solar irradiance	W/m2	Min-Max Norm.	[0.0;953.0]
Ind. Solar irradiance	W/m2	Min-Max Norm.	[0.0;1017.0]
Building			
Solar Generation	kWh	Min-Max Norm.	[0.0; 4.78]
Non-shiftable load	kWh	Min-Max Norm.	[0.0; 8.864]
Net Electricity Consumption	kWh	Min-Max Norm.	-
Battery SOC	-	Min-Max Norm.	[0.0; 1.0]
Carbon Intensity	kg CO2	Min-Max Norm.	[0.0704; 0.2818]
Electricity Priceing	$	Min-Max Norm.	[0.21; 0.54]

Table 4.6: Observation features for DRL agent and the transformations performed on them.

In addition to the use of the min-max norm transformation and the periodic transformation, Zhumabekov et. al. also included additional features that they claim improved their model performance. These were the previous timesteps observations and average energy consumption for that hour, the previous 6 hours and 12 hours. The previous timestep observations were included in the observation space of the agents.

4.3.4 Evaluation Metrics

The metrics by which the algorithms were compared are the metrics that were outlined in chapter 2. All building and district metrics are used to evaluate the success of algorithms with preference being given to district-level metrics as they are a good proxy for any coordination metric that could be implemented. District peak load and district ramping are particularly highly weighted when considering agent coordination as improvements in these metrics involve agents favourably coordinating their behavior.

Normalization of KPIs

To not only evaluate algorithmic performance between algorithms but also compare against the case where there is no battery or AI BMS, each KPI has been turned into a ratio between the KPI under algorithmic control and the KPI with no battery at all.

$$KPI = \frac{KPI_{control}}{KPI_{\text{baseline no battery}}} \tag{4.15}$$

4.3.5 Baseline reference controllers

In addition to comparison with other CLDE algorithms, baseline rule-based controllers and a random controller were also included in the study. The baseline agents consist of those that are included in the CityLearn environment: The basic rule-based controller and the optimized rule-based controller as well as a random agent as a lower

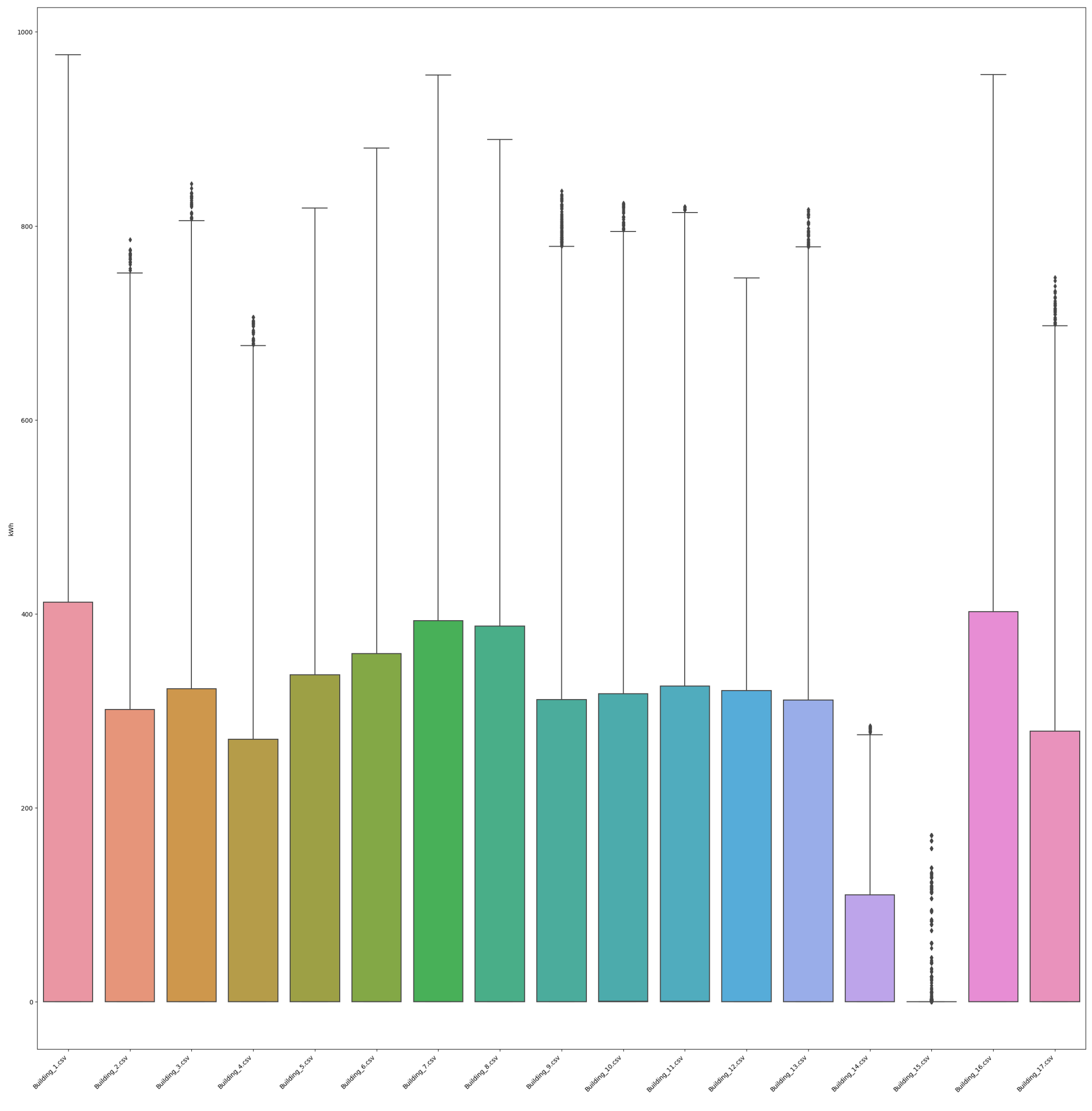

Figure 4.1: Load data histogram from the CityLearn Challenge 2022 Dataset

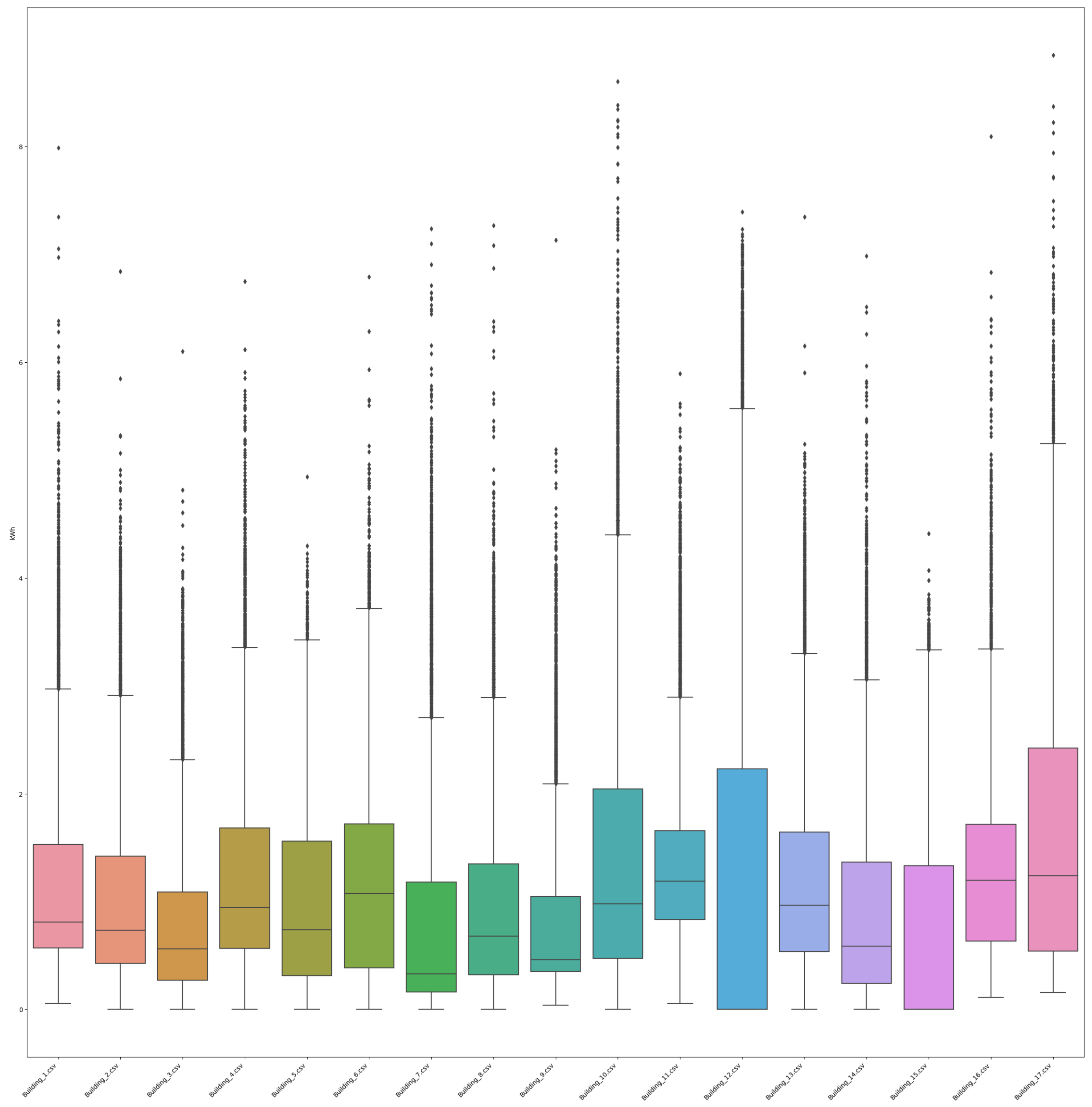

Figure 4.2: Solar Generation data histogram from the CityLearn Challenge 2022 Dataset

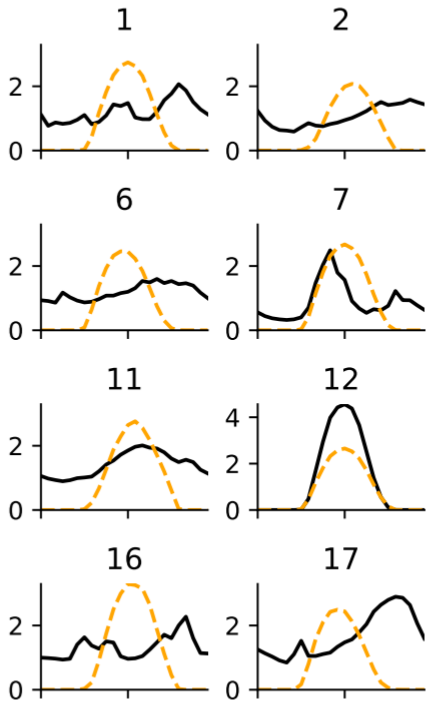

Figure 4.3: Load and generation distributions for all 17 buildings used in the CityLearn challenge 2022 created by Nweyw et. al. [45]

threshold for performance.

Basic Rule based Controller

The rule-based controller design that was used as a baseline agent is the same one that is described by Nweye et. al. and is based on the *time-of-Use Peak Reduction* battery control strategy used by the building batteries in the Sierra Crest Zero Net Energy community [42, 45, 46]. This control strategy is to charge between the hours of 9:00 am and 12:00 PM and then discharge starting at 6:00 pm, all done at a constant 2 kW rate. This controller maintains a 25% SOC at its minimum charge.

Optimized Rule Based Controller

The optimized rule based controller is also described by Nweye et. al. and is based on the *time-of-Use Peak Optimization* battery control strategy used by the building batteries in the Sierra Crest Zero Net Energy community [42, 45, 46]. This control

Algorithm/ KPI	Random Agent	Basic Rule Based Controller	Optimized Rule Based Controller	MAA2C Individual Reward	MAA2C Joint Reward	IA2C Individual Reward	IA2C Joint Reward	MAPPO Individual Reward	MAPPO Joint Reward	IPPO Individual Reward	IPPO Joint Reward
1 - Load Factor	1.028	0.987	0.991	**1.046**	1.071	1.058	1.115	1.072	1.062	1.103	1.127
Average Daily Peak	1.761	1.150	1.018	**1.426**	1.458	1.469	2.361	1.606	1.525	2.07	2.528
Carbon Emissions	1.776	1.186	1.055	1.485	1.464	1.503	1.502	1.522	**1.417**	1.46	**1.417**
Electricity Consumption	1.791	1.249	1.081	1.508	1.474	1.515	1.513	1.541	1.482	1.466	**1.441**
Peak Demand	1.549	1.052	0.992	**1.485**	1.694	1.589	2.356	1.801	1.612	1.82	2.549
Pricing	1.718	1.085	1.041	1.371	1.363	1.384	1.329	1.378	1.351	1.234	**1.196**
Ramping	3.538	1.162	1.081	**2.865**	2.966	3.166	3.567	3.359	3.033	3.225	3.634
Zero Net Energy	1.358	1.148	1.053	1.199	**1.186**	1.2	1.189	1.211	1.193	1.203	1.189

Table 4.7: District metric results for the CityLearn environment. Each Algorithm was evaluated using both individual rewards and joint rewards. Bolded Values indicate the best performing metrics.

strategy is to charge from 6:30 pm at 4.5 kW and then discharge from 12:00 pm. This controller maintains a 25% SOC at its minimum charge.

Random Agent Controller

The Random Agent controller consisted of a simple random sampling of the action space when selecting actions. This controller did not have any neural networks associated with it.

4.4 Results and Discussion

In this study, a detailed comparison of two potent algorithmic candidates, the individual reward MAA2C and joint reward IPPO, unveiled quantifiable performance differences. MAA2C showed an 8.1% increase in energy efficiency (1 - load factor) when operating the battery compared to IPPO. This algorithm also brought about a substantial 110.2% reduction in the average daily peak. Furthermore, MAA2C significantly curtailed peak demand by 110.9% and district ramping by 77% as a result of battery operation when using MAA2C.

Conversely, IPPO performed better in reducing carbon emissions and district electricity consumption, with reductions of 6.8% and 6.7% respectively over MAA2C. Furthermore, IPPO brought about a 17.5% reduction in district pricing and a minor 1% reduction in district zero net energy caused by the operation of the battery.

Given the higher priority assigned to district ramping and peak demand in our eval-

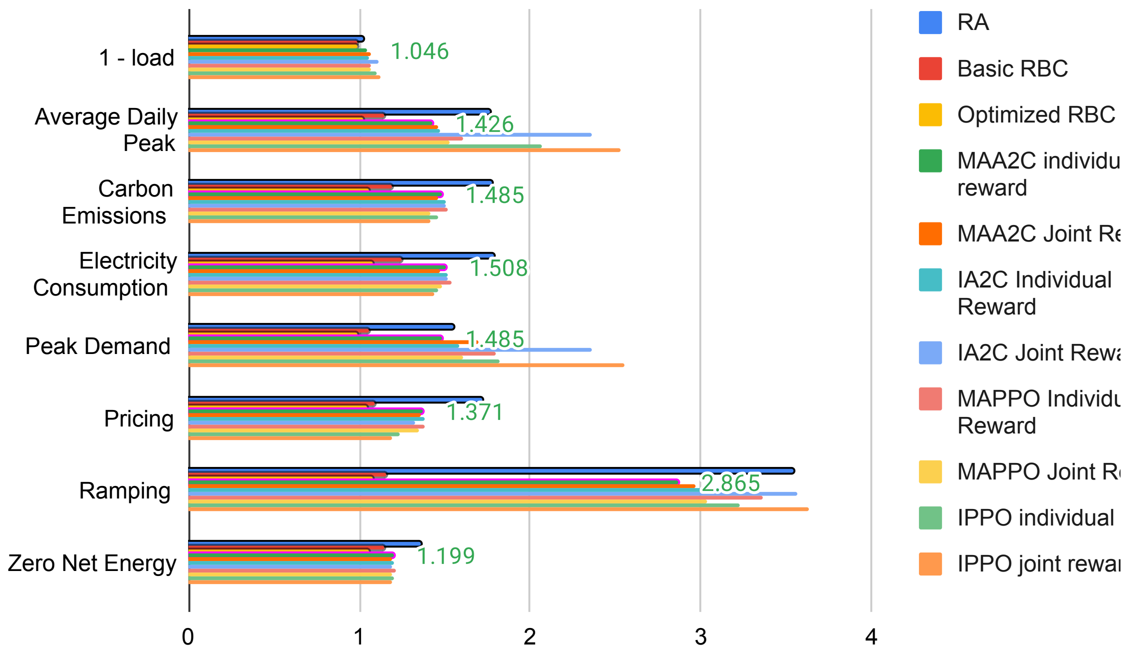

Figure 4.4: Distric KPI values of all evaluated algorithms on the CityLearn 2022 Phase 1 dataset. This figure shows the comparison between individual rewards and joint rewards. The values that are shown in the chart are the KPI values of MAA2C Individual rewards.

uation metrics, these quantitative results lead to the selection of MAA2C as the more favorable algorithm for this task. This choice is informed by MAA2C's exceptional performance in reducing peak demand, district ramping, and the average daily peak, despite IPPO's slightly superior performance in reducing carbon emissions, district electricity consumption, pricing, and zero net energy.

The study, however, encountered ambiguity when contrasting the effectiveness of joint versus individual reward schemes across different CLDE algorithms. Algorithms like IPPO, Independent A2C (IA2C), and MAA2C displayed overall enhancement across most KPIs with individual reward, while MAPPO behaved contradictorily, performing better with joint rewards. This inconsistency blurs the answer to the question, "Is it better to use joint reward or individual reward with CLDE algorithms?" A more comprehensive investigation, focusing on how each algorithm responds to joint versus individual reward, is required to draw definitive conclusions.

It is noteworthy that, while the performance of all algorithms fell short of the random agent (RA), none came close to the performance of either of the Rule-Based Controls (RBCs). This observation points to the possibility that the reward function may not be adequately optimized for shifting the load using the battery.

Interestingly, the outcomes of this study contrast sharply with the findings of our previous journal paper. In that study, MAPPO benefited most from individual rewards, while MAA2C was most negatively impacted by the shift from joint to individual rewards. This discrepancy underscores the importance of further research to clarify the complex dynamics at play in this domain.

An in-depth analysis of IPPO's performance when utilizing individual rewards, as compared to joint rewards, reveals significant differences in operational outcomes. Notably, implementing an individual reward scheme with IPPO results in a 45% reduction in the average daily peak caused by battery operation, a 41% reduction in ramping, and a substantial 79% reduction in peak demand. These outcomes underscore the potent influence of reward structuring on the IPPO algorithm's performance.

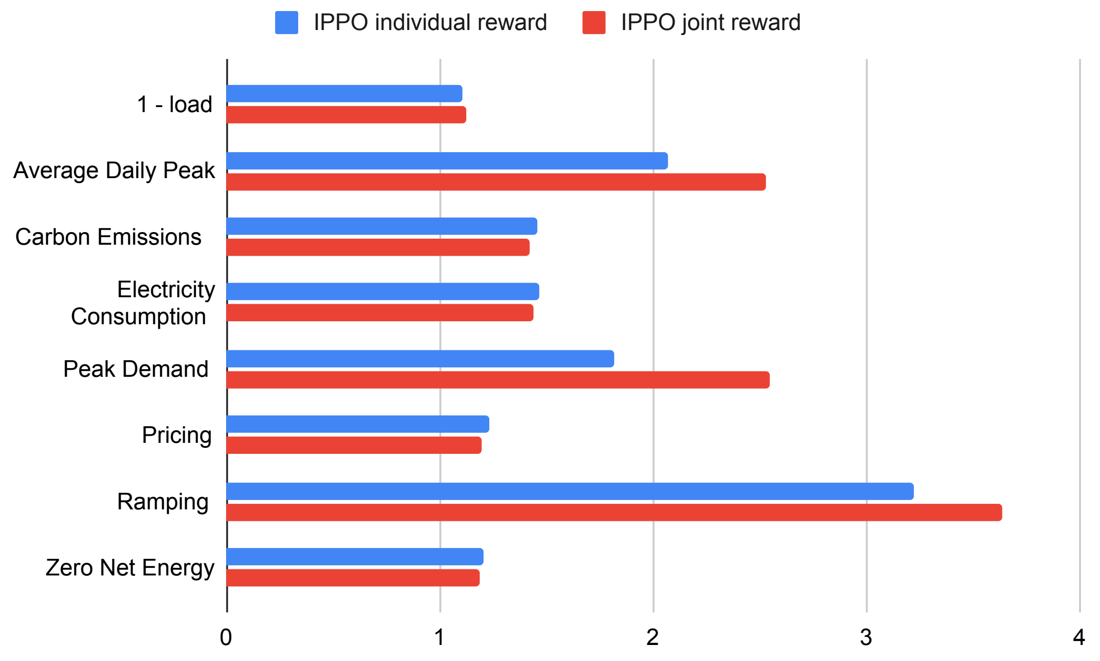

Figure 4.5: Comparison between district KPIs of IPPO independent reward and joint reward.

However, it is crucial to note that, regardless of the reward type used, IPPO underperforms compared to the random agent in specific metrics. Specifically, it performs 27% worse in peak demand, 7.5% worse in 1-load factor, and 30% worse in the average daily peak. These observations spotlight the current inadequacies in IPPO's performance, hinting at a need for more cautious and thoughtful considerations in the structuring of rewards. By revealing the critical importance of reward structures in influencing the effectiveness of an algorithm, these findings further emphasize the necessity for precision and strategic forethought in reward design. A comprehensive evaluation of the MAPPO algorithm reveals noteworthy results. Contrary to IPPO, MAPPO exhibits superior performance when employing joint rewards in comparison to individual rewards across all assessed categories. Specifically, employing joint rewards results in a 19% decrease in peak demand and a 33% reduction in ramping caused by battery operation. These findings signify the distinctive advantage of joint rewards in the functioning of the MAPPO algorithm.

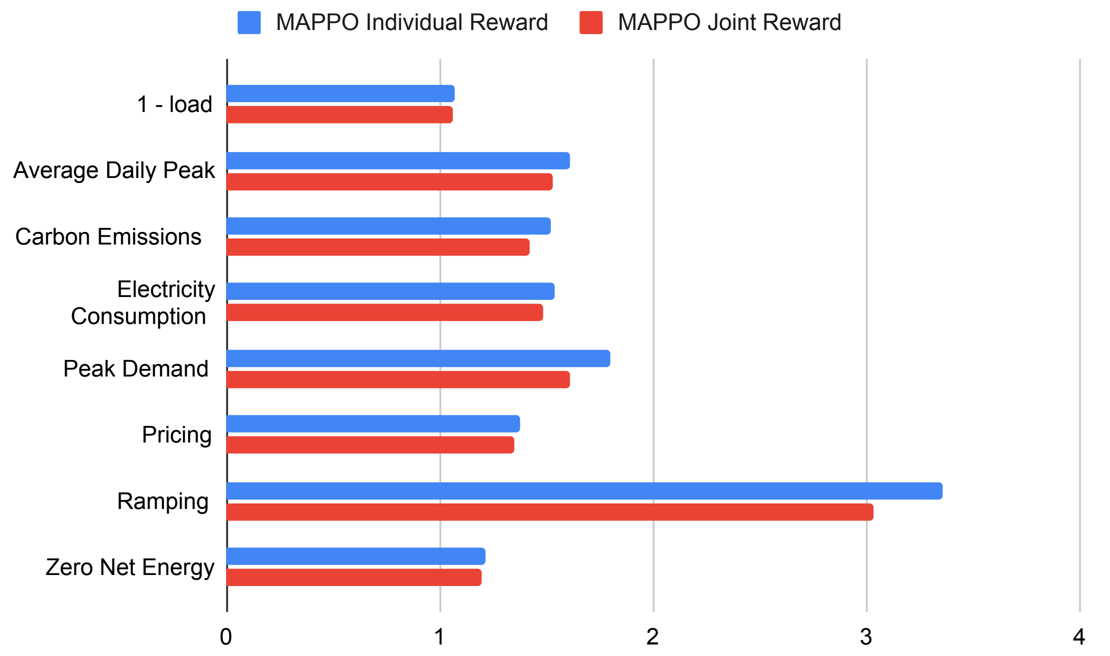

Figure 4.6: Comparison between district KPIs of MAPPO independent reward and joint reward.

Notably, irrespective of the reward scheme employed, MAPPO outperforms the random agent across all metrics, with the exceptions of 1-load factor and peak demand. In these particular areas, MAPPO falls short by 3.4% and 6.3% respectively when compared to the random agent. These results hint at areas for potential improvement in the performance of the MAPPO algorithm, while simultaneously underscoring its overall effectiveness in comparison to a random agent. The observed performance differences further highlight the integral role that the choice of reward structure plays in shaping algorithm performance outcomes. An intricate analysis of the IA2C algorithm presents noteworthy performance differences, particularly when comparing IA2C with individual rewards to IA2C with joint rewards. Predominantly, IA2C with individual rewards demonstrates superior outcomes. The performance of joint rewards only surpasses that of individual rewards in terms of district pricing, with a minor increase of 5.5%. However, this rise in cost is substantially overshadowed by the considerable reductions in other key areas, including an 89% decrease

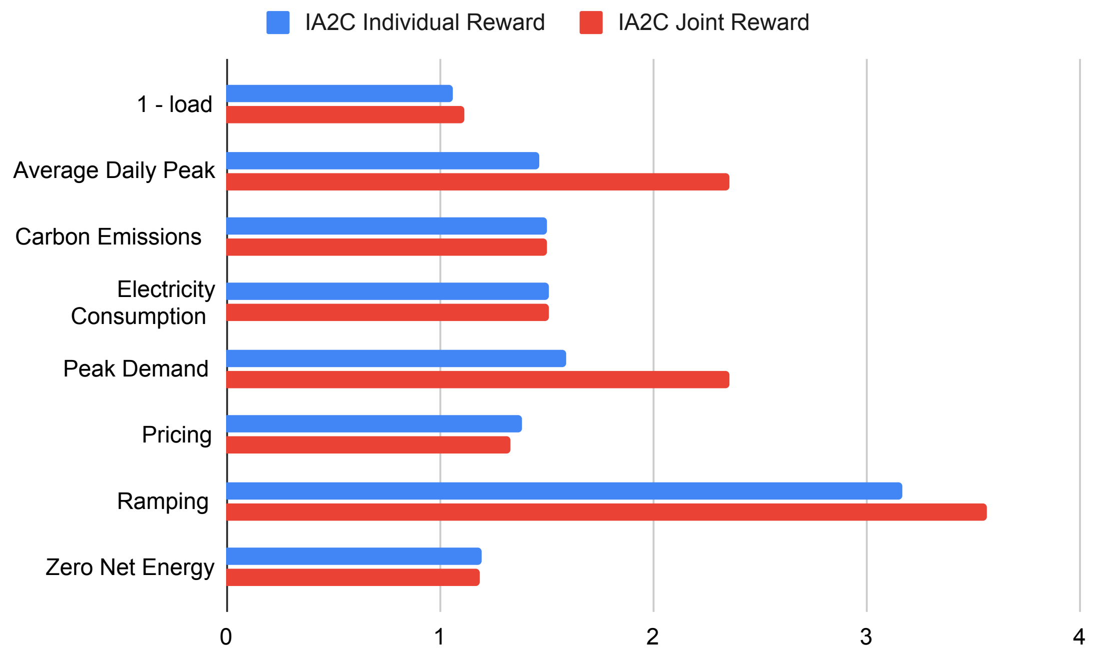

Figure 4.7: Comparison between district KPIs of IA2C independent reward and joint reward.

in average daily peak, a 76.1% reduction in peak demand, and a 40% diminution in ramping. The trade-off here clearly highlights the effectiveness of individual rewards, despite a slight increase in district pricing.

When compared to the random agent, IA2C with individual rewards further displays its proficiency with a 3% reduction in district 1-load factor and a 4% decrease in district peak demand. These results highlight the capabilities of the IA2C algorithm when individual rewards are utilized, providing a compelling case for its use over a random agent for battery control tasks.

A meticulous evaluation of the MAA2C algorithm exhibits intriguing insights, particularly when individual rewards are compared with joint rewards. MAA2C with individual rewards notably outperforms its joint reward counterpart in four of the KPIs, including the crucial peak demand and ramping KPIs. Specifically, a 10% reduction in district ramping and a 20% reduction in peak demand is observed when utilizing individual rewards.

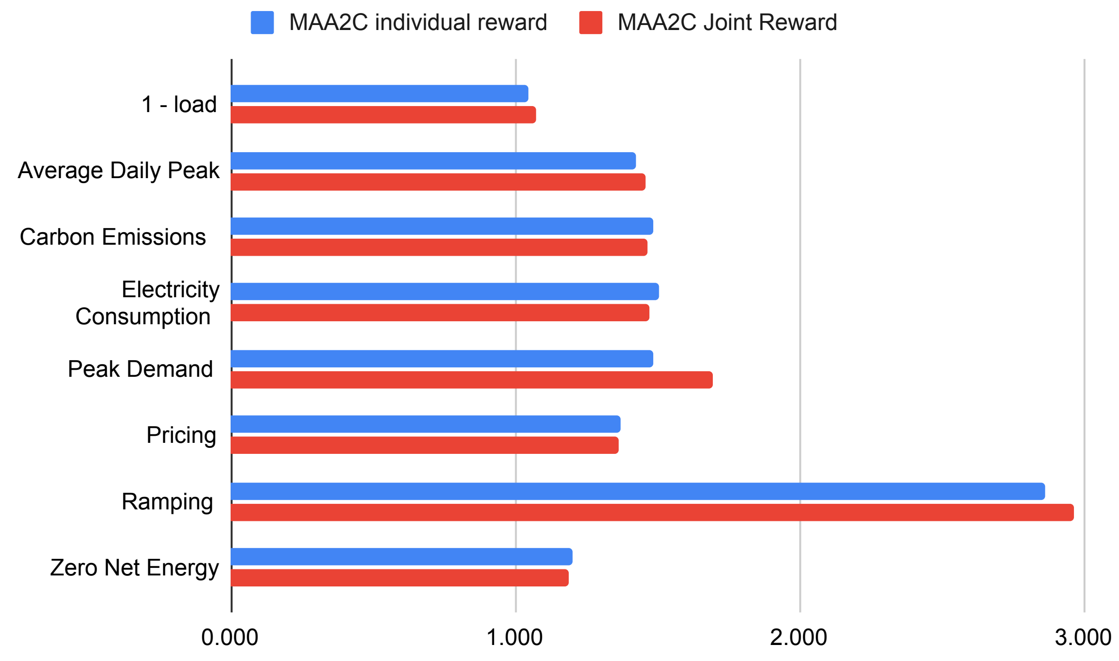

Figure 4.8: Comparison between district KPIs of MAA2C independent reward and joint reward.

Interestingly, MAA2C's performance excels regardless of the reward scheme, as it surpasses the random agent in all district KPIs. This consistent performance across all measured metrics strengthens the case for MAA2C as an ideal candidate algorithm for this task.

When juxtaposing the performance of MAA2C with individual rewards against that of the random agent, MAA2C boasts the highest performance improvement among all algorithms examined, manifesting a significant reduction of 67% in ramping. This achievement, in particular, underscores the efficiency of the MAA2C algorithm and its potential effectiveness in the CityLearn 2022 challenge task.

Chapter 5

Contributions & Future Work

This chapter summarizes the contributions of this thesis as well as outlining possible future research directions.

5.1 Contributions

The purpose of this chapter is to shed light on the significant insights unearthed from the studies conducted throughout this research project. From an overarching perspective, a critical finding that underpins our exploration of CLDE algorithms lies in the differential performance of these algorithms based on reward structures.

Beginning with our investigation in the LBF environment, we uncovered a surprising response range from the different CLDE algorithms when the reward structure was transitioned from joint to individual. The MAPPO and QMIX algorithms demonstrated resilience, harnessing the increased variance in the individual reward to aid in discovering optimal policies. In stark contrast, VDN and MAA2C algorithms struggled under these conditions, their performance deteriorating with the increased variance of the individual reward.

Our findings highlight an intriguing aspect of the CLDE algorithms - the delicate equilibrium required in the convergence rate of the centralized critic. The critic should converge at a rate that is leisurely enough to detect the optimal joint policy while avoiding entrapment in a local minimum. Moreover, we found that the critic should not be overly sensitive to an increase in variance to prevent divergence and subsequent inability to locate the optimal policy, as evidenced in the case of MAA2C. An additional revelation in our study was the apparent need for value decomposition methods to have supplementary state information for efficient agent coordination and optimal policy learning as evidenced by their increased performance with joint reward.

The crucial role of agent collaboration in influencing emergent behavior in MARL systems led us to surmise the potential importance of reward function selection in MARL environments, possibly more than in single-agent ones. Our initial findings suggest a possible bias-variance trade-off in relation to joint and individual rewards, a conjecture that deserves further exploration.

Transitioning to the second phase of our investigation, the CityLearn task, our focus was directed toward understanding the impact of reward structures on CLDE algorithm performance in another environment with similar dynamics to LBF. In addition, we sought to evaluate the CLDE algorithms on several KPIs that were not directly encoded in the reward function and relied on the agents all coordinating together. The comparative study of MAA2C, IPPO, IA2C, and MAPPO operating under both individual and joint rewards brought to light the profound effect of these factors on algorithm performance and efficiency.

A key highlight from this phase of the study was the superior performance of MAA2C with individual rewards across multiple Key Performance Indicators (KPIs). This algorithm significantly outperformed competitors in peak demand and district ramping and surpassed the random agent across all district KPIs. Notably, while IPPO and IA2C showcased strengths under individual rewards, they were found wanting in certain areas compared to the random agent. MAPPO, interestingly, displayed superior performance with joint rewards, underscoring the intricate subtleties inherent in algorithm performance across different contexts.

Through these insights, we wish to emphasize the pivotal role of careful reward structuring in optimizing algorithm performance. Additionally, our findings reinforce the importance of a strategic, considerate approach to the selection and application of reinforcement learning algorithms, considering the task's specific demands and objectives.

Despite the promising findings from our studies, the journey to perfect these algorithms and their corresponding reward structures is far from over. The vast scope for further exploration to build upon these initial findings opens up exciting prospects. Enhancing our understanding of these complex systems can drive more efficient, superior solutions in district energy management.

5.2 Future Work

Our research journey has revealed valuable insights into the complex realm of MARL algorithms and CLDE algorithms. However, like all scientific endeavors, it has also highlighted new questions and potential areas for further exploration. In this section, we outline the promising avenues for future work, drawing on the findings and limitations of our current studies.

An intriguing conjecture that has surfaced proposes that choosing between joint and individual rewards when training MARL algorithms may be a manifestation of a bias-variance trade-off. This theory needs further validation. A first step would be to study simpler scenarios, enabling analytical verification of the proposition that individual rewards lead to an increase in variance. These scenarios must retain the characteristic sparse positive reward observed in the LBF environment.

Upon the successful establishment of this theoretical foundation, one could then begin to dismantle the constraints of sparsity and positive reward, evaluating whether the theory continues to hold true. These systematic investigations aim to conclusively shed light on the impact of varying reward functions in cooperative MARL systems.

Several recommendations for improvement and further exploration have been identified within the context of the CityLearn environment.

The simplicity of the reward function utilized raises questions about its ability to effectively elicit desired behaviors. Future investigations into the design and impact of more sophisticated reward functions are thus warranted. These explorations should also move beyond solely positive, sparse rewards to consider the effects of negative and frequent rewards. The opportunity for more precise tuning exists, given that the present work only involved a coarse hyperparameter search. An exhaustive exploration of the hyperparameter space in future studies can unearth optimal configurations, ultimately enhancing the performance of the applied algorithms. At present, agents are trained on a limited dataset comprised of the first five houses. A larger, more diverse dataset, including all available houses, could significantly enrich the learning environment. This modification will better simulate the complexity and variability of real-world energy systems, leading to the development of more robust and generalizable models. While the current frequency of data collection stands at a 1-hour interval, more granular data – such as data collected every minute – could provide a finer understanding of energy consumption patterns and enable more timely decision-making. This inclusion of higher-frequency data in future CityLearn experiments may improve agent performance.

Opportunities also exist to broaden the analysis of individual rewards in CLDE algorithms outside the CityLearn environment. It is recommended to extend investigations into different reward conditions, including negative rewards and rewards at frequent intervals, beyond those of positive and sparse rewards currently explored. A broader examination of these conditions will offer a more comprehensive understanding of the performance of CLDE algorithms in diverse scenarios.

To summarize, these future directions offer the potential to further our understanding and refinement of MARL and CLDE algorithms. By pursuing advancements in reward function design, hyperparameter tuning, training dataset expansion, data frequency, and broadening CLDE algorithm analysis, the field can move closer to realizing more efficient and intelligent agent-based energy management systems.

Bibliography

[4] M. Samvelyan, T. Rashid, C. S. de Witt, G. Farquhar, N. Nardelli, T. G. J. Rudner, C.-M. Hung, P. H. S. Torr, J. Foerster, and S. Whiteson, "The StarCraft Multi-Agent Challenge," *CoRR*, vol. abs/1902.04043, 2019.

[5] F. A. Oliehoek and C. Amato, "The decentralized pomdp framework," in *A Concise Introduction to Decentralized POMDPs*. Cham: Springer International Publishing, 2016, pp. 11–32.

[1] G. Papoudakis, F. Christianos, L. Schäfer, and S. V. Albrecht, "Benchmarking multi-agent deep reinforcement learning algorithms in cooperative tasks," *CoRR*, vol. abs/2006.07869, 2020. arXiv: 2006.07869. [Online]. Available: https://arxiv.org/abs/2006.07869.

[6] 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 *International conference on machine learning*, PMLR, 2016, pp. 1928–1937.

[7] C. Yu, A. Velu, E. Vinitsky, Y. Wang, A. M. Bayen, and Y. Wu, "The surprising effectiveness of MAPPO in cooperative, multi-agent games," *CoRR*, vol. abs/2103.01955, 2021. arXiv: 2103.01955. [Online]. Available: https://arxiv.org/abs/2103.01955.

[8] R. Lowe, Y. I. Wu, A. Tamar, J. Harb, O. Pieter Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," *Advances in neural information processing systems*, vol. 30, 2017.

[9] 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," *CoRR*, vol. abs/1803.11485, 2018. arXiv: 1803.11485. [Online]. Available: http://arxiv.org/abs/1803.11485.

[10] P. Sunehag, G. Lever, A. Gruslys, W. M. Czarnecki, V. Zambaldi, M. Jaderberg, M. Lanctot, N. Sonnerat, J. Z. Leibo, K. Tuyls, *et al.*, "Value-decomposition networks for cooperative multi-agent learning," *arXiv preprint arXiv:1706.05296*, 2017.

[11] X. Lyu, Y. Xiao, B. Daley, and C. Amato, "Contrasting centralized and decentralized critics in multi-agent reinforcement learning," *CoRR*, vol. abs/2102.04402, 2021. arXiv: 2102.04402. [Online]. Available: https://arxiv.org/abs/2102.04402.

[12] K. Tumer and D. Wolpert, "A survey of collectives," *Collectives and the design of complex systems*, pp. 1–42, 2004.

[13] M. Colby, T. Duchow-Pressley, J. J. Chung, and K. Tumer, "Local approximation of difference evaluation functions," in *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*, 2016, pp. 521–529.

[14] A. K. Agogino and K. Tumer, "Analyzing and visualizing multiagent rewards in dynamic and stochastic domains," *Autonomous Agents and Multi-Agent Systems*, vol. 17, pp. 320–338, 2008.

[15] S. Proper and K. Tumer, "Modeling difference rewards for multiagent learning.," in *AAMAS*, 2012, pp. 1397–1398.

[16] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction.* MIT press, 2018.

[17] C. J. C. H. Watkins, "Learning from delayed rewards," 1989.

[18] R. J. Williams, *Reinforcement-learning connectionist systems.* College of Computer Science, Northeastern University, 1987.

[19] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Reinforcement learning*, pp. 5–32, 1992.

[20] I. H. Witten, "An adaptive optimal controller for discrete-time markov environments," *Information and control*, vol. 34, no. 4, pp. 286–295, 1977.

[21] A. Barto, R. Sutton, and C. Anderson, "Neuron like elementsthat can solve difficult learning control problems," *IEEE Trans. Systems, Man, and Cybernetics*, vol. 13, no. 5.1983, p. 835,

[22] R. S. Sutton, *Temporal credit assignment in reinforcement learning.* University of Massachusetts Amherst, 1984.

[23] 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*, vol. 518, no. 7540, pp. 529–533, 2015.

[24] G. A. Rummery and M. Niranjan, *On-line Q-learning using connectionist systems.* University of Cambridge, Department of Engineering Cambridge, UK, 1994, vol. 37.

[25] R. S. Sutton, "Generalization in reinforcement learning: Successful examples using sparse coarse coding," *Advances in neural information processing systems*, vol. 8, 1995.

[26] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[27] Y. LeCun *et al.*, "Generalization and network design strategies," *Connectionism in perspective*, vol. 19, no. 143-155, p. 18, 1989.

[28] P. Hernandez-Leal, B. Kartal, and M. E. Taylor, "A survey and critique of multiagent deep reinforcement learning," *Autonomous Agents and Multi-Agent Systems*, vol. 33, no. 6, pp. 750–797, 2019.

[29] P. Kollock, "Social dilemmas: The anatomy of cooperation," *Annual review of sociology*, vol. 24, no. 1, pp. 183–214, 1998.

[30] E. Hughes, J. Z. Leibo, M. G. Philips, K. Tuyls, E. A. Duéñez-Guzmán, A. G. Castañeda, I. Dunning, T. Zhu, K. R. McKee, R. Koster, H. Roff, and T. Graepel, "Inequity aversion resolves intertemporal social dilemmas," *CoRR*, vol. abs/1803.08884, 2018. arXiv: 1803.08884. [Online]. Available: http://arxiv.org/abs/1803.08884.

[31] J. Z. Leibo, V. F. Zambaldi, M. Lanctot, J. Marecki, and T. Graepel, "Multi-agent reinforcement learning in sequential social dilemmas," *CoRR*, vol. abs/1702.03037, 2017. arXiv: 1702.03037. [Online]. Available: http://arxiv.org/abs/1702.03037.

[32] J. Z. Leibo, E. A. Dueñez-Guzman, A. 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," in *International Conference on Machine Learning*, PMLR, 2021, pp. 6187–6199.

[33] R. Köster, K. R. McKee, R. Everett, L. Weidinger, W. S. Isaac, E. Hughes, E. A. Duéñez-Guzmán, T. Graepel, M. Botvinick, and J. Z. Leibo, "Model-free conventions in multi-agent reinforcement learning with heterogeneous preferences," *arXiv preprint arXiv:2010.09054*, 2020.

[34] F. Christianos, L. Schäfer, and S. V. Albrecht, "Shared experience actor-critic for multi-agent reinforcement learning," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[35] S. M. Ismael, S. H. A. Aleem], A. Y. Abdelaziz, and A. F. Zobaa, "State-of-the-art of hosting capacity in modern power systems with distributed generation," *Renewable Energy*, vol. 130, pp. 1002 –1020, 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0960148118307936.

[36] T. O. Olowu, A. Sundararajan, M. Moghaddami, and A. I. Sarwat, "Future challenges and mitigation methods for high photovoltaic penetration: A survey," *Energies*, vol. 11, no. 7, 2018.

[37] C Chapelsky, K Gerasimov, and P Musilek, "DER impacts to urban utilities study summary," 2019. [Online]. Available: https://www.epcor.com/products-services/power/Documents/micro-generation-research-solar-energy-electricity-grid-2019.pdf.

[38] M. D. A.Fattahi Meyabadi, "A review of demand-side management: Reconsidering theoretical framework," *Renewable and Sustainable Energy Reviews*, vol. 80, pp. 367–379, 2017.

[39] S. Chen and C.-C. Liu, "From demand response to transactive energy: State of the art," *Journal of Modern Power Systems and Clean Energy*, vol. 5, no. 1, pp. 10–19, 2017.

[40] O. Abrishambaf, F. Lezama, P. Faria, and Z. Vale, "Towards transactive energy systems: An analysis on current trends," *Energy Strategy Reviews*, vol. 26, p. 100 418, 2019.

[41] J. R. Vázquez-Canteli and Z. Nagy, "Reinforcement learning for demand response: A review of algorithms and modeling techniques," *Applied energy*, vol. 235, pp. 1072–1089, 2019.

[42] J. R. Vázquez-Canteli, J. Kämpf, G. Henze, and Z. Nagy, "Citylearn v1. 0: An openai gym environment for demand response with deep reinforcement learning," in *Proceedings of the 6th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*, 2019, pp. 356–357.

[43] *System advisor model.* [Online]. Available: https://sam.nrel.gov/.

[2] P. Atrazhev and P. Musilek, "Investigating effects of centralized learning decentralized execution on team coordination in the level based foraging environment as a sequential social dilemma," in *International Conference on Practical Applications of Agents and Multi-Agent Systems*, Springer, 2022, pp. 15–23.

[3] P. Atrazhev and P. Musilek, "It's all about reward: Contrasting joint rewards and individual reward in centralized learning decentralized execution algorithms," *Systems*, vol. 11, no. 4, p. 180, 2023.

[44] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *CoRR*, vol. abs/1707.06347, 2017. arXiv: 1707.06347. [Online]. Available: http://arxiv.org/abs/1707.06347.

[45] K. Nweye, S. Sankaranarayanan, and Z. Nagy, "Merlin: Multi-agent offline and transfer learning for occupant-centric energy flexible operation of grid-interactive communities using smart meter data and citylearn," *arXiv preprint arXiv:2301.01148*, 2022.

[46] *Neurips 2022: Citylearn challenge: Challenges.* [Online]. Available: https://www.aicrowd.com/challenges/neurips-2022-citylearn-challenge.

[47] A. Zhumabekov, D. May, T. Zhang, A. K. GS, O. Ardakanian, and M. E. Taylor, "Ensembling diverse policies improves generalizability of reinforcement learning algorithms in continuous control tasks,"

Appendix A: Supporting Materials for Publications

This appendix contains the supporting information for the publications. The first appendix contains information on the replication of the results from Papoudakis et. al. [1]. The second appendix contains the statistical underpinnings of the differences in algorithmic performance between joint reward and indiviual reward in the LBF environment that was included in [3].

A.1 Replication Study Papoudakis et. al.

As part of our work on analysis of algorithmic performance we replicated the work that was done as part of [1] on the LBF environment. This section includes the data that was collected from our repeated experiments. We used the hyperparameters that were reported in the appendix section of [1] and ran 10 runs for each hyperparameter configuration. The selected hyperparameters were those for parameter sharing and parameter sharing was used for the data collection to keep in line with the results in [1].

We found discrepancies between the reported data in [1] for VDN and QMIX, and these discrepancies also seem to explain some of the results we reported in [2]. Notably, we found that the convergence of the value factorization methods was not reported properly in [1] and these convergence values are in line with the increase in convergence rates that we found in [2].

Tasks/Algs	IQL	IA2C	IPPO	MAA2C	MAPPO	VDN	QMIX
8x8-2p-2f-c	1.0 ± 0.0	1.0 ± 0.00	1.0 ± 0.00	1.0 ± 0.00	1.0 ± 0.0	1.0 ± 0.00	1.0 ± 0.00
8x8-2p-2f-2s-c	0.97 ± 0.01	1.0 ± 0.0	0.63 ± 0.02	1.0 ± 0.0	0.56 ± 0.02	0.98 ± 0.00	0.97 ± 0.0
10x10-3p-3f	0.89 ± 0.08	0.99 ± 0.01	0.89 ± 0.02	0.98 ± 0.01	0.9 ± 0.24	0.9 ± 0.03	0.91 ± 0.02
10x10-3p-3f-2s	0.7 ± 0.01	0.84 ± 0.04	0.56 ± 0.01	0.85 ± 0.01	0.58 ± 0.01	0.77 ± 0.01	0.76 ± 0.04

Table A.1: Maximum returns and 95% confidence interval of hyperparameter configurations taken from [1]. Bolded values are those that differ significantly from [1].

Tasks/Algs	IQL	IA2C	IPPO	MAA2C	MAPPO	VDN	QMIX
8x8-2p-2f-c	0.77 ± 0.08	0.96 ± 0.01	0.96 ± 0.01	0.97 ± 0.01	0.96 ± 0.02	0.78 ± 0.04	$\mathbf{0.69 \pm 0.04}$
8x8-2p-2f-2s-c	0.82 ± 0.01	0.94 ± 0.01	0.39 ± 0.02	0.94 ± 0.0	0.45 ± 0.02	0.84 ± 0.0	0.79 ± 0.01
10x10-3p-3f	0.47 ± 0.07	0.88 ± 0.02	0.71 ± 0.03	0.87 ± 0.02	0.59 ± 0.21	0.56 ± 0.03	0.46 ± 0.04
10x10-3p-3f-2s	0.56 ± 0.01	0.67 ± 0.05	0.44 ± 0.0	0.69 ± 0.02	0.46 ± 0.0	0.6 ± 0.01	0.56 ± 0.05

Table A.2: Average returns and 95% confidence interval of hyperparameter configurations taken from [1]. Bolded values are those that differ significantly from [1].

A.2 Variance analysis between joint reward, individual reward with varying timesteps on LBF

This section of the appendix contains all the statistical data analysis that was used during the empirical variance analysis in section 3.3. The statistical analysis used Bartlett's test in order to determin if the variance of two means are the same. The α value used to determine statistical significance is $\alpha = 0.05$. Bartlett's test tests the null hypothesis h_0 that the variances of each data distribution tested are identical. If the p-value is below that of the selected α, then the null hypothesis is rejected and the variances of the data tested are not the same. In our analysis, the data collected for each run was averaged over and then the set of 10 replicates were used in Bartlett's test. The nan value indicates that there was no variation at all because the algorithm was able to solve the scenario perfectly in the 25 timestep scenarios for both individual and joint reward.

IQL

Below are the statistics that were gathered on the IQL algorithm. The results aspects of the algorithm that were compared include: *loss*, *grad norm*, *mean of selected q values*, *means of return*,*max of returns* , and *target network mean q values for the selected action.* Variances are evaluated between joint reward and independent reward. Bolded p-values reject the null hypothesis, indicating that the variances between 25 step and 50 step runs are different.

	Foraging-8x8-2p-2f-coop/	Foraging-2s-8x8-2p-2f-coop/	Foraging-10x10-3p-3f/	Foraging-2s-10x10-3p-3f/
25	0.88	0.15	**0.048**	0.35
50	**0.066**	0.63	0.18	**0.044**

Table A.3: P-values of Bartlett's test for homogeneity of variances for gradient norm values of IQL between 25 timesteps and 50 timesteps grouped by scenario.

	Foraging-8x8-2p-2f-coop/	Foraging-2s-8x8-2p-2f-coop/	Foraging-10x10-3p-3f/	Foraging-2s-10x10-3p-3f/
25	**0.014**	0.095	**1.45e-3**	0.71
50	0.21	0.069	**6.42e-4**	0.67

Table A.4: P-values of Bartlett's test for homogeneity of variances for loss values of IQL between 25 timesteps and 50 timesteps.

	Foraging-8x8-2p-2f-coop/	Foraging-2s-8x8-2p-2f-coop/	Foraging-10x10-3p-3f/	Foraging-2s-10x10-3p-3f/
25	0.68	0.50	0.074	**0.002**
50	0.56	0.49	0.059	0.64

Table A.5: P-values of Bartlett's test for homogeneity of variances for the mean q value of selected actions of IQL between 25 timesteps and 50 timesteps grouped by scenario.

IPPO

Below are the statistics that were gathered on the IPPO algorithm. The statistics that were tested include: *Mean return*, *Max return*, *Agent grad norms*, *critic grad*

	Foraging-8x8-2p-2f-coop/	Foraging-2s-8x8-2p-2f-coop/	Foraging-10x10-3p-3f/	Foraging-2s-10x10-3p-3f/
25	0.70	0.47	0.080	**2.08e-3**
50	0.56	0.47	0.061	0.63

Table A.6: P-values of Bartlett's test for homogeneity of variances for the target value of selected actions of IQL between 25 timesteps and 50 timesteps grouped by scenario.

	Foraging-8x8-2p-2f-coop/	Foraging-2s-8x8-2p-2f-coop/	Foraging-10x10-3p-3f/	Foraging-2s-10x10-3p-3f/
25	0.99	0.36	**2.80e-3**	**7.73e-3**
50	0.73	0.48	**0.023**	0.57

Table A.7: P-values of Bartlett's test for homogeneity of variances for the mean return values of IQL between 25 timesteps and 50 timesteps grouped by scenario.

	Foraging-8x8-2p-2f-coop/	Foraging-2s-8x8-2p-2f-coop/	Foraging-10x10-3p-3f/	Foraging-2s-10x10-3p-3f/
25	**1.84e-6**	0.77	**5.28e-5**	0.41
50	nan	**9.22e-3**	**1.60e-3**	0.47

Table A.8: P-values of Bartlett's test for homogeneity of variances for the max return values of IQL between 25 timesteps and 50 timesteps grouped by scenario.

norms, *Critic loss*, *Policy gradient loss*, *Maximum Pi values of the actor* and *Advantage means*. Variances are evaluated between joint reward and independent reward. Bolded p-values reject the null hypothesis, indicating that the variances between 25 step and 50 step runs are different.

	Foraging-8x8-2p-2f-coop/	Foraging-2s-8x8-2p-2f-coop/	Foraging-10x10-3p-3f/	Foraging-2s-10x10-3p-3f/
25	**4.52e-5**	**0.074**	0.54	**1.14e-5**
50	0.16	**3.39e-6**	0.75	**1.37e-4**

Table A.9: P-values of Bartlett's test for homogeneity of variance for mean returns of IPPO varying episode length between 25 timesteps and 50 timesteps and comparing reward functions.

	Foraging-8x8-2p-2f-coop/	Foraging-2s-8x8-2p-2f-coop/	Foraging-10x10-3p-3f/	Foraging-2s-10x10-3p-3f/
25	**0.0**	**0.013**	**0.049**	**9.04e-4**
50	**0.0**	**8.66e-7**	**8.26e-6**	0.34

Table A.10: P-values of Bartlett's test for homogeneity of variance for max returns of IPPO varying episode length between 25 timesteps and 50 timesteps and comparing reward functions.

	Foraging-8x8-2p-2f-coop/	Foraging-2s-8x8-2p-2f-coop/	Foraging-10x10-3p-3f/	Foraging-2s-10x10-3p-3f/
25	**8.27e-17**	**9.97e-4**	**1.17e-12**	**3.23e-14**
50	**9.14e-16**	**5.11e-8**	**8.82e-13**	**5.99e-18**

Table A.11: P-values of Bartlett's test for homogeneity of variance for agent grad norms returns of IPPO varying episode length between 25 timesteps and 50 timesteps and comparing reward functions.

	Foraging-8x8-2p-2f-coop/	Foraging-2s-8x8-2p-2f-coop/	Foraging-10x10-3p-3f/	Foraging-2s-10x10-3p-3f/
25	**1.98e-21**	**1.15e-13**	**2.98e-22**	**2.27e-15**
50	**4.39e-23**	**2.60e-15**	**9.49e-20**	**5.75e-18**

Table A.12: P-values of Bartlett's test for homogeneity of variance for critic grad norms returns of IPPO varying episode length between 25 timesteps and 50 timesteps and comparing reward functions.

	Foraging-8x8-2p-2f-coop/	Foraging-2s-8x8-2p-2f-coop/	Foraging-10x10-3p-3f/	Foraging-2s-10x10-3p-3f/
25	**1.51e-25**	**2.56e-10**	**2.33e-22**	**2.43e-17**
50	**1.69e-28**	**1.79e-11**	**1.01e-18**	**2.90e-18**

Table A.13: P-values of Bartlett's test for homogeneity of variance for critic loss of IPPO varying episode length between 25 timesteps and 50 timesteps and comparing reward functions.

	Foraging-8x8-2p-2f-coop/	Foraging-2s-8x8-2p-2f-coop/	Foraging-10x10-3p-3f/	Foraging-2s-10x10-3p-3f/
25	**1.86e-17**	**2.06e-9**	**5.25e-14**	**4.11e-12**
50	**1.22e-22**	**9.84e-17**	**7.55e-14**	**4.60e-14**

Table A.14: P-values of Bartlett's test for homogeneity of variance for policy gradient loss of IPPO varying episode length between 25 timesteps and 50 timesteps and comparing reward functions.

	Foraging-8x8-2p-2f-coop/	Foraging-2s-8x8-2p-2f-coop/	Foraging-10x10-3p-3f/	Foraging-2s-10x10-3p-3f/
25	**1.39e-5**	**1.28e-3**	0.19	0.18
50	0.64	**6.63e-4**	**1.88e-4**	**0.011**

Table A.15: P-values of Bartlett's test for homogeneity of variance for maximum policy values of IPPO varying episode length between 25 timesteps and 50 timesteps and comparing reward functions.

	Foraging-8x8-2p-2f-coop/	Foraging-2s-8x8-2p-2f-coop/	Foraging-10x10-3p-3f/	Foraging-2s-10x10-3p-3f/
25	**2.05e-17**	**3.98e-10**	**9.86e-15**	**2.59e-12**
50	**1.12e-21**	**1.38e-16**	**5.35e-14**	**8.63e-14**

Table A.16: P-values of Bartlett's test for homogeneity of variance for advantage means of IPPO varying episode length between 25 timesteps and 50 timesteps and comparing reward functions.

IA2C

Below are the statistics that were gathered on the IA2C algorithm. T The statistics that were tested include: *Mean return*, *Max return*, *Agent grad norms*, *critic grad norms*, *Critic loss*, *Policy gradient loss*, *Maximum Pi values of the actor* and *Advantage means*. Variances are evaluated between joint reward and independent reward. Bolded p-values reject the null hypothesis, indicating that the variances between 25 step and 50 step runs are different.

	Foraging-8x8-2p-2f-coop/	Foraging-2s-8x8-2p-2f-coop/	Foraging-10x10-3p-3f/	Foraging-2s-10x10-3p-3f/
25	**0.016**	0.35	0.63	**6.69e-3**
50	**0.003**	0.38	0.12	0.063

Table A.17: P-values of Bartlett's test for homogeneity of variances for mean return values of IA2C between 25 timesteps and 50 timesteps.

VDN

Below are the statistics that were gathered on the VDN algorithm. The results aspects of the algorithm that were compared include: *loss*, *grad norm*, *mean of selected q values*, *means of return*, *max of returns*,and *target network mean q values for the se-*

	Foraging-8x8-2p-2f-coop/	Foraging-2s-8x8-2p-2f-coop/	Foraging-10x10-3p-3f/	Foraging-2s-10x10-3p-3f/
25	**0.0**	**4.27e-4**	**0.071**	0.29
50	**0.0**	**0.0**	**2.07e-9**	**0.016**

Table A.18: P-values of Bartlett's test for homogeneity of variances for max return values of IA2C between 25 timesteps and 50 timesteps.

	Foraging-8x8-2p-2f-coop/	Foraging-2s-8x8-2p-2f-coop/	Foraging-10x10-3p-3f/	Foraging-2s-10x10-3p-3f/
25	0.25	0.24	0.33	**0.005**
50	0.13	0.31	**0.019**	**0.010**

Table A.19: P-values of Bartlett's test for homogeneity of variances for critic grad norm of IA2C between 25 timesteps and 50 timesteps.

	Foraging-8x8-2p-2f-coop/	Foraging-2s-8x8-2p-2f-coop/	Foraging-10x10-3p-3f/	Foraging-2s-10x10-3p-3f/
25	0.60	0.19	**0.011**	**4.12e-4**
50	0.81	0.33	**0.045**	0.17

Table A.20: P-values of Bartlett's test for homogeneity of variances for critic loss of IA2C between 25 timesteps and 50 timesteps.

	Foraging-8x8-2p-2f-coop/	Foraging-2s-8x8-2p-2f-coop/	Foraging-10x10-3p-3f/	Foraging-2s-10x10-3p-3f/
25	0.25	0.24	0.33	**4.89e-3**
50	0.13	0.31	**0.019**	**9.87e-3**

Table A.21: P-values of Bartlett's test for homogeneity of variances for PG loss of IA2C between 25 timesteps and 50 timesteps.

	Foraging-8x8-2p-2f-coop/	Foraging-2s-8x8-2p-2f-coop/	Foraging-10x10-3p-3f/	Foraging-2s-10x10-3p-3f/
25	0.20	0.17	**3.13e-3**	**0.033**
50	**0.029**	0.13	0.15	**3.15e-3**

Table A.22: P-values of Bartlett's test for homogeneity of variances for advantage mean of IA2C between 25 timesteps and 50 timesteps.

lected action. Variances are evaluated between joint reward and independent reward. Bolded p-values reject the null hypothesis, indicating that the variances between 25 step and 50 step runs are different.

	Foraging-8x8-2p-2f-coop/	Foraging-2s-8x8-2p-2f-coop/	Foraging-10x10-3p-3f/	Foraging-2s-10x10-3p-3f/
25	0.30	0.41	1.00	0.13
50	0.45	**0.011**	0.55	**0.005**

Table A.23: P-values of Bartlett's test for homogeneity of variances for gradient norm values of VDN between 25 timesteps and 50 timesteps grouped by scenario.

	Foraging-8x8-2p-2f-coop/	Foraging-2s-8x8-2p-2f-coop/	Foraging-10x10-3p-3f/	Foraging-2s-10x10-3p-3f/
25	0.17	0.40	**0.016**	0.10
50	0.87	0.58	0.33	0.076

Table A.24: P-values of Bartlett's test for homogeneity of variances for loss values of VDN between 25 timesteps and 50 timesteps.

	Foraging-8x8-2p-2f-coop/	Foraging-2s-8x8-2p-2f-coop/	Foraging-10x10-3p-3f/	Foraging-2s-10x10-3p-3f/
25	**0.034**	0.099	0.77	0.052
50	**0.021**	0.87	0.83	0.20

Table A.25: P-values of Bartlett's test for homogeneity of variances for the mean q value of selected actions of VDN between 25 timesteps and 50 timesteps grouped by scenario.

	Foraging-8x8-2p-2f-coop/	Foraging-2s-8x8-2p-2f-coop/	Foraging-10x10-3p-3f/	Foraging-2s-10x10-3p-3f/
25	**0.034**	0.11	0.78	**0.038**
50	**0.020**	0.86	0.81	0.18

Table A.26: P-values of Bartlett's test for homogeneity of variances for the target network mean q values of selected actions of VDN between 25 timesteps and 50 timesteps grouped by scenario.

	Foraging-8x8-2p-2f-coop/	Foraging-2s-8x8-2p-2f-coop/	Foraging-10x10-3p-3f/	Foraging-2s-10x10-3p-3f/
25	**0.038**	**0.002**	0.27	0.11
50	0.36	0.75	0.71	0.37

Table A.27: P-values of Bartlett's test for homogeneity of variances for the mean return values VDN between 25 timesteps and 50 timesteps grouped by scenario.

QMIX

Below are the statistics that were gathered on the QMIX algorithm. The results aspects of the algorithm that were compared include: *loss*, *grad norm*, *mean of selected*

	Foraging-8x8-2p-2f-coop/	Foraging-2s-8x8-2p-2f-coop/	Foraging-10x10-3p-3f/	Foraging-2s-10x10-3p-3f/
25	**0.0**	0.50	0.31	0.089
50	**0.0**	0.26	0.26	**0.003**

Table A.28: P-values of Bartlett's test for homogeneity of variances for the max return values VDN between 25 timesteps and 50 timesteps grouped by scenario.

q values, *means of return*, *max of returns* ,and *target network mean q values for the selected action.* Variances are evaluated between joint reward and independent reward. Bolded p-values reject the null hypothesis, indicating that the variances between 25 step and 50 step runs are different.

	Foraging-8x8-2p-2f-coop/	Foraging-2s-8x8-2p-2f-coop/	Foraging-10x10-3p-3f/	Foraging-2s-10x10-3p-3f/
25	**8.18e-6**	0.13	**7.12e-5**	**5.66e-8**
50	**9.17e-32**	**1.84e-10**	0.25	**3.55e-4**

Table A.29: P-values of Bartlett's test for homogeneity of variances for loss values of Qmix between 25 timesteps and 50 timesteps.

	Foraging-8x8-2p-2f-coop/	Foraging-2s-8x8-2p-2f-coop/	Foraging-10x10-3p-3f/	Foraging-2s-10x10-3p-3f/
25	**2.92e-8**	0.20	**6.36e-7**	**7.19e-10**
50	**1.09e-17**	**1.53e-11**	**1.06e-3**	**9.73e-7**

Table A.30: P-values of Bartlett's test for homogeneity of variances for gradient norm values of Qmix between 25 timesteps and 50 timesteps grouped by scenario.

	Foraging-8x8-2p-2f-coop/	Foraging-2s-8x8-2p-2f-coop/	Foraging-10x10-3p-3f/	Foraging-2s-10x10-3p-3f/
25	**0.04**	0.55	**0.01**	**1.65e-8**
50	**0.04**	**2.86e-8**	**0.03**	0.08

Table A.31: P-values of Bartlett's test for homogeneity of variances for the mean q value of selected actions of Qmix between 25 timesteps and 50 timesteps grouped by scenario.

	Foraging-8x8-2p-2f-coop/	Foraging-2s-8x8-2p-2f-coop/	Foraging-10x10-3p-3f/	Foraging-2s-10x10-3p-3f/
25	**0.04**	0.52	**6.18e-3**	**1.21e-8**
50	**0.03**	**2.66e-8**	**0.03**	0.07

Table A.32: P-values of Bartlett's test for homogeneity of variances for the target network mean q values of selected actions of Qmix between 25 timesteps and 50 timesteps grouped by scenario.

	Foraging-8x8-2p-2f-coop/	Foraging-2s-8x8-2p-2f-coop/	Foraging-10x10-3p-3f/	Foraging-2s-10x10-3p-3f/
25	*Nan*	0.90	0.76	**2.73e-4**
50	**4.65e-6**	0.83	**3.90e-4**	0.21

Table A.33: P-values of Bartlett's test for homogeneity of variances for the max return values Qmix between 25 timesteps and 50 timesteps grouped by scenario.

	Foraging-8x8-2p-2f-coop/	Foraging-2s-8x8-2p-2f-coop/	Foraging-10x10-3p-3f/	Foraging-2s-10x10-3p-3f/
25	0.87	0.61	0.40	**1.24e-5**
50	**7.55e-3**	**2.10e-2**	0.51	0.25

Table A.34: P-values of Bartlett's test for homogeneity of variances for the mean return values Qmix between 25 timesteps and 50 timesteps grouped by scenario.

MAA2C

Below are the statistics that were gathered on the MAA2C algorithm. The statistics that were tested include: *Mean return*, *Max return*, *Agent grad norms*, *critic grad norms*, *Critic loss*, *Policy gradient loss*, *Maximum Pi values of the actor* and *Advantage means*. Bolded p-values reject the null hypothesis, indicating that the variances between 25 step and 50 step runs are different.

	Foraging-8x8-2p-2f-coop/	Foraging-2s-8x8-2p-2f-coop/	Foraging-10x10-3p-3f/	Foraging-2s-10x10-3p-3f/
25	**1.46e-7**	**9.93e-5**	0.12	0.22
50	**9.03e-8**	**7.12e-12**	**1.59e-7**	0.89

Table A.35: P-values of Bartlett's test for homogeneity of variance for mean returns of MAA2C varying episode length between 25 timesteps and 50 timesteps and comparing reward functions.

	Foraging-8x8-2p-2f-coop/	Foraging-2s-8x8-2p-2f-coop/	Foraging-10x10-3p-3f/	Foraging-2s-10x10-3p-3f/
25	0.0	9.83e-10	1.09e-4	0.011
50	0.0	0.0	9.83e-2	0.28

Table A.36: P-values of Bartlett's test for homogeneity of variance for Max Returns of MAA2C varying episode length between 25 timesteps and 50 timesteps and comparing reward functions.

	Foraging-8x8-2p-2f-coop/	Foraging-2s-8x8-2p-2f-coop/	Foraging-10x10-3p-3f/	Foraging-2s-10x10-3p-3f/
25	0.90	**7.12e-2**	0.43	**4.62e-3**
50	**1.76e-2**	0.50	0.80	**4.54e-2**

Table A.37: P-values of Bartlett's test for homogeneity of variance for agent grad norms of MAA2C varying episode length between 25 timesteps and 50 timesteps and comparing reward functions.

	Foraging-8x8-2p-2f-coop/	Foraging-2s-8x8-2p-2f-coop/	Foraging-10x10-3p-3f/	Foraging-2s-10x10-3p-3f/
25	0.33	**0.020**	**2.81e-5**	0.74
50	0.13	**0.005**	0.13	**0.012**

Table A.38: P-values of Bartlett's test for homogeneity of variance for critic grad norms of MAA2C varying episode length between 25 timesteps and 50 timesteps and comparing reward functions.

	Foraging-8x8-2p-2f-coop/	Foraging-2s-8x8-2p-2f-coop/	Foraging-10x10-3p-3f/	Foraging-2s-10x10-3p-3f/
25	**0.027**	**2.52e-4**	**0.004**	0.69
50	**0.029**	0.11	**1.20e-5**	0.59

Table A.39: P-values of Bartlett's test for homogeneity of variance for critic loss of MAA2C varying episode length between 25 timesteps and 50 timesteps and comparing reward functions.

	Foraging-8x8-2p-2f-coop/	Foraging-2s-8x8-2p-2f-coop/	Foraging-10x10-3p-3f/	Foraging-2s-10x10-3p-3f/
25	**3.20e-5**	**1.32e-5**	**2.22e-3**	**0.029**
50	**0.034**	**1.17e-6**	**0.014**	**0.025**

Table A.40: P-values of Bartlett's test for homogeneity of variance for pg loss of MAA2C varying episode length between 25 timesteps and 50 timesteps and comparing reward functions.

	Foraging-8x8-2p-2f-coop/	Foraging-2s-8x8-2p-2f-coop/	Foraging-10x10-3p-3f/	Foraging-2s-10x10-3p-3f/
25	**3.28e-7**	**6.16e-4**	0.18	**0.019**
50	**3.31e-4**	**1.64e-3**	**1.94e-8**	0.061

Table A.41: P-values of Bartlett's test for homogeneity of variance for max policy values of MAA2C varying episode length between 25 timesteps and 50 timesteps and comparing reward functions.

	Foraging-8x8-2p-2f-coop/	Foraging-2s-8x8-2p-2f-coop/	Foraging-10x10-3p-3f/	Foraging-2s-10x10-3p-3f/
25	**1.51e-3**	**0.025**	**1.11e-3**	0.071
50	0.38	**3.53e-4**	0.90	**1.07e-4**

Table A.42: P-values of Bartlett's test for homogeneity of variance for advantage mean values of MAA2C varying episode length between 25 timesteps and 50 timesteps and comparing reward functions.

MAPPO

Below are the statistics that were gathered on the MAPPO algorithm. The statistics that were tested include: *Mean return*, *Max return*, *Agent grad norms*, *critic grad norms*, *Critic loss*, *Policy gradient loss*, *Maximum Pi values of the actor* and *Advantage means*. Bolded p-values reject the null hypothesis, indicating that the variances between 25 step and 50 step runs are different.

	Foraging-8x8-2p-2f-coop/	Foraging-2s-8x8-2p-2f-coop/	Foraging-10x10-3p-3f/	Foraging-2s-10x10-3p-3f/
25	**3.21e-4**	0.12	**1.08e-6**	**2.85e-6**
50	**2.76e-6**	0.84	0.92	**4.05e-6**

Table A.43: P-values of Bartlett's test for homogeneity of variance for return means of MAPPO varying episode length between 25 timesteps and 50 timesteps and comparing reward functions.

	Foraging-8x8-2p-2f-coop/	Foraging-2s-8x8-2p-2f-coop/	Foraging-10x10-3p-3f/	Foraging-2s-10x10-3p-3f/
25	**3.28e-5**	0.90	**3.71e-8**	**3.70e-4**
50	**0.00**	**1.86e-5**	**1.66e-6**	**1.74e-3**

Table A.44: P-values of Bartlett's test for homogeneity of variance for return maxes of MAPPO varying episode length between 25 timesteps and 50 timesteps and comparing reward functions.

	Foraging-8x8-2p-2f-coop/	Foraging-2s-8x8-2p-2f-coop/	Foraging-10x10-3p-3f/	Foraging-2s-10x10-3p-3f/
25	**1.73e-12**	**1.85e-5**	**2.49e-10**	**3.24e-9**
50	**6.17e-16**	**2.60e-8**	**3.29e-13**	**4.52e-18**

Table A.45: P-values of Bartlett's test for homogeneity of variance for agent grad norms of MAPPO varying episode length between 25 timesteps and 50 timesteps and comparing reward functions.

	Foraging-8x8-2p-2f-coop/	Foraging-2s-8x8-2p-2f-coop/	Foraging-10x10-3p-3f/	Foraging-2s-10x10-3p-3f/
25	**1.29e-17**	**2.35e-11**	**6.32e-11**	**4.66e-12**
50	**1.53e-20**	**1.51e-19**	**9.26e-22**	**8.30e-20**

Table A.46: P-values of Bartlett's test for homogeneity of variance for critic grad norm of MAPPO varying episode length between 25 timesteps and 50 timesteps and comparing reward functions.

	Foraging-8x8-2p-2f-coop/	Foraging-2s-8x8-2p-2f-coop/	Foraging-10x10-3p-3f/	Foraging-2s-10x10-3p-3f/
25	**3.46e-18**	**6.95e-7**	**1.15e-7**	**6.13e-8**
50	**3.70e-22**	**1.11e-17**	**1.14e-14**	**3.87e-16**

Table A.47: P-values of Bartlett's test for homogeneity of variance for policy gradient loss of MAPPO varying episode length between 25 timesteps and 50 timesteps and comparing reward functions.

	Foraging-8x8-2p-2f-coop/	Foraging-2s-8x8-2p-2f-coop/	Foraging-10x10-3p-3f/	Foraging-2s-10x10-3p-3f/
25	**6.89e-3**	0.83	**1.44e-6**	0.22
50	**0.015**	0.15	0.40	0.10

Table A.48: P-values of Bartlett's test for homogeneity of variance for max policy values of MAPPO varying episode length between 25 timesteps and 50 timesteps and comparing reward functions.

	Foraging-8x8-2p-2f-coop/	Foraging-2s-8x8-2p-2f-coop/	Foraging-10x10-3p-3f/	Foraging-2s-10x10-3p-3f/
25	**2.81e-18**	**4.19e-7**	**3.13e-7**	**5.83e-8**
50	**6.95e-22**	**1.27e-17**	**1.62e-14**	**4.02e-16**

Table A.49: P-values of Bartlett's test for homogeneity of variance for advantage mean values of MAPPO varying episode length between 25 timesteps and 50 timesteps and comparing reward functions.