Overcooked: Exploring Collaborative MARL

Kirsten Odendaal
kodendaal3@gatech.edu
College of Computing
Georgia Institute of Technology

1 Introduction

Multi-Agent Reinforcement Learning (MARL) has shown promise for tasks requiring precise coordination, such as the *Overcooked* environment, where agents jointly prepare and deliver soups by completing interdependent objectives. This study compares baseline Independent Double Deep Q-Network (IQL-DDQN) agents against Value Decomposition Network (VDN-DDQN) agents and clearly highlights how joint-value decomposition promotes effective collaboration while considering curriculum methods. Furthermore, an ablation analysis explores critical algorithmic components, including symmetric replay buffers, exploration robustness, and DDQN architectures which help in identifying factors that significantly influence performance in collaborative MARL scenarios.

2 Technical Background

This section briefly reviews fundamental concepts in MARL, highlighting key principles that form the theoretical basis for the upcoming empirical evaluations.

2.1 Multi-Agent Reinforcement Learning

MARL extends Single-Agent Reinforcement Learning (RL) to scenarios where multiple agents concurrently interact with and influence a shared environment (Albrecht et al., 2024). MARL introduces unique challenges, including:

- Non-stationarity: Each agent's environment continuously evolves as other agent policies change during learning, which has the potential to greatly complicate convergence properties.
- Moving Target Problem: Bootstrapped Q-value updates depend on changing subsequent state-value estimates, creating training instability. Target networks can help to avoid this by providing more stable updates.

To address these complexities, Centralized Training - Decentralized Execution (CTDE) is commonly considered. Under CTDE, agents utilize global information during centralized training to optimize joint objectives, allowing for sophisticated coordination strategies (Albrecht et al., 2024). However, at execution time, agents rely only on local observations. This promotes scalability and practical deployment for real-world scenarios in which such information cannot be known. Despite the effectiveness of CTDE, simpler decentralized approaches such as Independent Q-Learning (IQL) often serve as a valuable baseline due to their conceptual simplicity and ease of implementation.

2.2 Independent Q-Learning

IQL is a decentralized approach that decomposes a multi-agent scenario into multiple simultaneous single-agent problems (Albrecht et al., 2024) working in a shared local environment. Although it doesn't explicitly handle non-stationarity and thus lacks convergence guarantees, it remains a practical baseline due to ease of implementation (Rashid et al., 2018).

2.3 Value Decomposition Network

Value decomposition networks (VDN) aim to address coordination by learning a joint action-value function decomposed into individual agent value functions. Formally:

$$Q_{\text{tot}}(\tau, \mathbf{u}) = \sum_{i=1}^{N} Q_i(\tau_i, u_i)$$

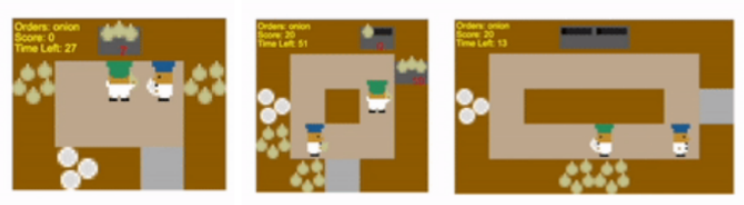


Figure 1: Overcooked layouts (left-to-right): Cramped Room, Coordination Ring, Counter Circuit

where τ and $\mathbf{u}$ represent joint observation-action histories and joint actions, respectively, and $Q_i(\tau_i, u_i)$ is the individual agent utility based on local observation-action history (Sunehag et al., 2017; Rashid et al., 2018). By decomposing the joint value into additive individual utilities, VDN can capture the contributions of individual agents toward collective outcomes, allowing for decentralized execution and improved coordination. While effective, VDN's linear additive decomposition limits its ability to capture complex, non-linear interactions among agents. Environments which require complex inter-agent dependencies, alternative methods such as QMIX may be more suitable (Rashid et al., 2018).

2.4 Deep Q-Network and Double DQN

Deep Q-Network (DQN) introduced function approximation via neural networks to represent the action-value function, enabling Q-learning to scale effectively to high-dimensional state spaces (Mnih et al., 2013). However, DQN suffers from an overestimation bias due to the maximization step in target value calculations (van Hasselt et al., 2015). DDQN avoids this by separating action selection from evaluation. DDQN's target update rule is given by:

$$y = r + \gamma Q(s', \arg\max_{a'} Q(s', a'; \theta); \theta^-)$$

where θ and θ^- represent parameters of the online and target networks respectively (van Hasselt et al., 2015). Reducing this overestimation bias is particularly beneficial in multi-agent reinforcement learning, as it provides more stable value estimations critical for effective cooperation.

Additionally, to take advantage of the symmetrical structure inherent to cooperative tasks within the *Overcooked* two-agent environment, the replay buffer is augmented with symmetric experiences. Specifically, each original experience tuple (s, a, r, s') is stored alongside its symmetric counterpart (s_m, a_m, r, s'_m). This effectively doubles the usable data and can significantly improve sample efficiency and promote better generalization (Lin et al., 2020; Kim et al., 2024).

2.5 Exploration Strategies

An epsilon-greedy exploration with linear decay was used. Initially high exploration ($\epsilon_{max} = 1.0$) linearly decays to a lower bound ($\epsilon_{min} = 0.1$) by 80% of training episodes. After this point, the parameters remain constant to ensure stable convergence. Formally, the exploration probability at episode i is computed as:

$$\epsilon_i = \max\left(\epsilon_{min},\ \epsilon_{max} - (\epsilon_{max} - \epsilon_{min})\frac{i}{0.8 \cdot E_{max}}\right)$$

where E_{max} is the total number of episodes used for training. During policy evaluation, agent policies are tested with a small fixed exploration probability ($\epsilon = 0.05$) which introduces slight randomness. This allows verification of learned MARL policy robustness and generalization under minor changes and perturbations, reflecting realistic operational uncertainties.

2.6 Curriculum Learning

Curriculum learning is a structured training approach in machine learning, where models or agents are systematically exposed to tasks of increasing complexity. Initially, agents are trained on simplified scenarios to acquire foundational skills and gradually progress toward tackling more intricate tasks. Using the internal weights of the network trained on simpler tasks as a starting point, rather

Component	Overcooked Problem	Description
State Space (S)	96-dimensional vector	Fully observable MDP: Player-centric features including relative positions, object states, and other agent states.
Action Space (A)	Discrete: {up, down, left, right, stay, interact}	Movement and contextual *interact* action determined by the environment tile facing the agent.
Transition Dynamics	Deterministic discrete simulation	Deterministic state transitions occur based on simultaneous agent actions within kitchen environment rules (e.g., cooking, serving, pickup).
Reward Function ($R(s, a)$)	$R = R_{\text{pot}} + R_{\text{dish}} + R_{\text{pickup}} + R_{\text{soup}}$	(+) 20 points per successful soup delivery *Reward Shaping (used in this study):* (+) 3 point for agent placing onion in pot. (+) 1 point for agent picking up dishes. (+) 5 points for agent picking up soup. (−) Inefficient actions indirectly penalized by consuming valuable time without reward.
Episode Termination	Fixed 400-timestep horizon	Episodes always run exactly 400 timesteps without early termination.
Environmental Parameters	Layouts: *Cramped Room, Coordination Ring, Counter Circuit*	Three kitchen layouts requiring varying levels of multi-agent collaboration, coordination complexity, and policy generalization.

Table 1: Overcooked Environment Components

than initializing from a random distribution allows the models to efficiently build upon previously mastered concepts, thereby accelerating the learning process (Narvekar et al., 2020).

3 Problem Definitions

The Overcooked environment is a cooperative multi-agent scenario that models the complexities of coordinated decision-making. Agents must collaboratively complete tasks that require interdependent actions under time constraints, making it an ideal testbed for exploring MARL algorithms that facilitate cooperation and efficient task allocation. Specifically, agents are tasked with preparing and delivering soups, demanding close coordination and effective strategy alignment to maximize performance.

3.1 Environment Description

In the discrete Overcooked environment, two chef agents cooperate in a simulated kitchen to prepare onion soups. Preparing each soup involves placing exactly three onions into a pot, cooking for 20 timesteps, transferring the cooked soup onto a dish, and serving it to a designated area (Carroll et al., 2019a;b). Although agents do not receive explicit penalties, inefficiencies such as dropping soups or incorrect servings implicitly penalize them by wasting valuable time.

Agents perform discrete actions, including directional movements (up, down, left, right, stay) and a contextual *interact* action. Each episode is constrained to 400 timesteps, during which the primary goal is to maximize the number of successful soup deliveries. Performance benchmarks require achieving an average of at least seven soup deliveries per episode, assessed across three progressively challenging kitchen layouts (see Figure 1). The key components of the environment, such as termination conditions and reward structures, are summarized in Table 1.

3.2 Overcoming Critical Challenges

The complexity of the Overcooked environment mainly comes from its multi-agent nature, requiring agents to coordinate actions effectively while navigating shared kitchen resources like pots, onions, and dishes. Agents must develop complementary strategies, minimize collisions, and efficiently manage resources under tight time constraints. Another complication is the delayed and sparse reward structure. Agents only receive rewards when successfully delivering completed soups. This sparsity makes it difficult to assign credit to individual actions, therefore requiring advanced credit-assignment methods and long-term cooperative strategies (Albrecht et al., 2024). Additionally, agents encounter a high-dimensional observation space represented by detailed, agent-specific, 96-

dimensional vectors, complicating the extraction of important information such as relative positions and cooking statuses.

To tackle these challenges, specialized MARL methods have been applied. CTDE frameworks facilitate coordination during training while allowing agents to operate independently during execution (Albrecht et al., 2024). VDN explicitly allocates credit to each agent's actions, effectively addressing the sparse reward problem (Sunehag et al., 2017). Furthermore, advanced reward shaping provides agents with meaningful intermediate feedback signals, helping reduce issues related to delayed and sparse rewards. Curriculum learning, training progressively from simpler to complex layouts, ensures policy generalization. Collectively, these approaches offer a comprehensive solution to the complex coordination and learning challenges inherent in the Overcooked environment, as further detailed in Section 2.4.

4 Methodology

This section outlines the methodology used for training and evaluating various MARL algorithms within the discrete action-space of the *Overcooked* environment. It provides details on the structured training procedures, hyperparameter optimization strategies, neural network architectures, and ablation studies used to assess the impact of specific algorithmic components.

4.1 Training and Evaluation Process

Training and evaluation followed a structured, reproducible pipeline, with episode counts increasing by layout complexity (1000, 2500, and 5000, respectively). The process included:

- Hyperparameter optimization on the *Cramped Room* layout (10 trials, 200 episodes each) using Optuna (Akiba et al., 2019), guided by literature-recommended values.
- Training and evaluating the baseline IQL-DDQN and the cooperative-focused VDN-DDQN model across all layouts using optimized parameters. Performance was compared directly.
- Monitoring cumulative rewards and soups delivered per episode. Evaluations occurred every 10 episodes using greedy policies with slight noise injection, each spanning 100 simulations.
- Implementing curriculum learning by initializing training for the complex *Counter Circuit* layout using pretrained weights from simpler layouts.
- Conducting ablations on *Cramped Room* to evaluate reward shaping, DDQN vs. DQN, and symmetric vs. standard replay buffers.

4.2 Hyperparameter and Neural Network Architectures

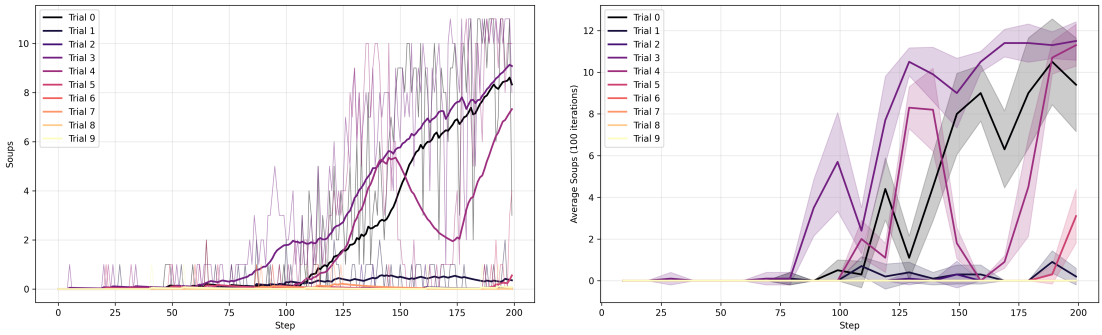
Given the sensitivity of MARL algorithms to hyperparameter selection, Bayesian optimization (using Optuna) was used to systematically identify effective parameter values. Table 2 summarizes the search ranges for both hyperparameters and neural network architectures. The optimization targeted parameters known to significantly affect agent performance, including discount factors, learning rates, replay buffer sizes, epsilon decay ratios, and batch sizes (Sunehag et al., 2017). However, hyperparameter optimization is not the primary focus of this study. It serves as an initial step to establish reasonable baseline settings for the subsequent experiments.

4.3 Ablation Considerations

To systematically analyze the contributions of various algorithmic components, three specific ablation cases were examined exclusively within the *Cramped Room* layout. First, the relative performance of the standard DQN and Double DQN architectures was evaluated to determine whether reducing Q-value overestimation bias provides meaningful benefits. Second, training with explicit reward shaping was compared against training without reward shaping to quantify potential improvements in performance. Finally, the impact of augmenting the replay buffer with symmetric experiences was assessed against using a regular shared replay buffer, highlighting any improvements in sample efficiency and generalization.

Hyperparameter	Search Range	Optimized
Learning Rate (α)	$[10^{-5}, 10^{-2}]$ (log scale)	10^{-5}
Discount Factor (γ)	[0.90, 0.99]	0.98
Soft Update Rate (τ)	$[10^{-4}, 10^{-2}]$ (log scale)	2^{-3}
Epsilon Decay Ratio	[0.5, 1.0]	0.70
Batch Size	256, 512, 768, 1024	1024
Hidden Layer 1 Size	64, 128, 192, 256	192
Hidden Layer 2 Size	64, 128, 192, 256	128

Table 2: Hyperparameter Optimization Ranges for DDQN



(a) Training

(b) Evaluation

Figure 2: Hyperparameter optimization (left) training curves with smoothed curves ($\alpha = 0.95$) (right) evaluation curves (1σ) across 100 iterations

For computational efficiency, each of these studies used evaluations consisting of 100 episodes. Although limited, this number of episodes provides enough insight into the relative impact of each algorithmic component, allowing for quick comparisons without large computational costs. Additionally, the patterns observed in these focused experiments can inform and guide algorithmic refinements applicable to more complex scenarios.

5 Experiment Results

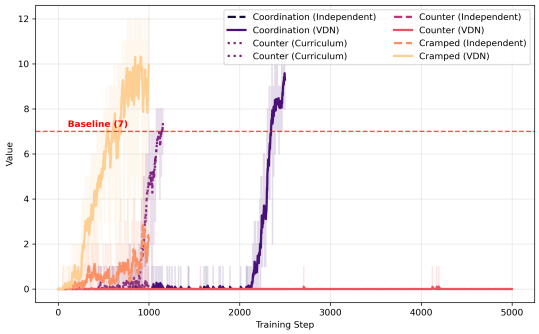
5.1 Hyperparameter Optimization

Hyperparameter optimization serves as an initial screening phase to quickly identify effective parameters that support training stability and enhance overall performance.

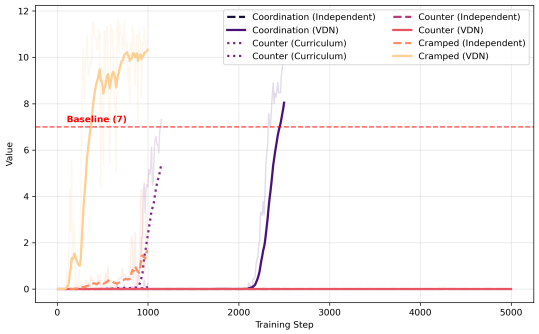
Figure 2 summarizes the optimization results from ten trials conducted in the *Cramped Room* layout. Among these, *Trial 3* demonstrated the most consistent improvements in early convergence and final performance, achieving a higher average number of soup deliveries than the other configurations. Other trials demonstrated larger variability or slower convergence, highlighting the importance of parameter selection in shaping policy learning behaviour. Table 2 shows the optimal hyperparameters found through this optimization process, focusing on key factors such as learning rate (α), discount factor (γ), soft update rate (τ), epsilon decay ratio, batch size, and hidden layer sizes. These hyperparameters are consistently used as the fix configuration for corresponding comparative evaluations and ablation studies.

5.2 Results Comparison

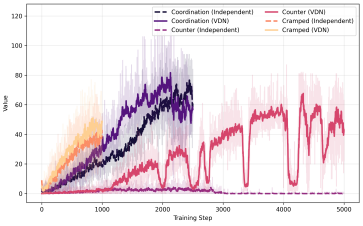
A systematic comparison between the baseline (IQL-DDQN) and VDN-DDQN approaches was performed across the three Overcooked environment layouts: *Cramped Room*, *Coordination Ring*, and *Counter Circuit*. Figure 3 presents a comprehensive view of performance metrics, including soups delivered, individual agent rewards, and detailed actions such as dish pickups and onion placements.



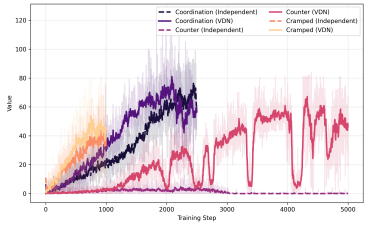
(a) Soups-made Training



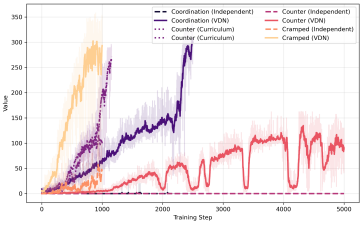
(b) Soups-made Evaluation



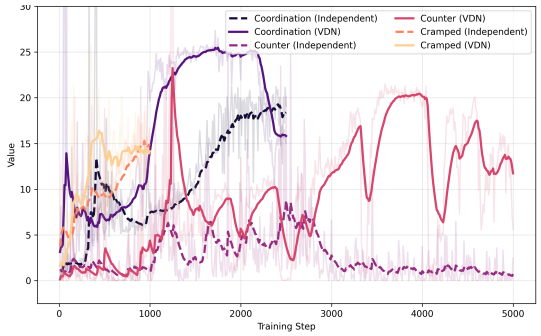
(c) Reward Agent-1 Training



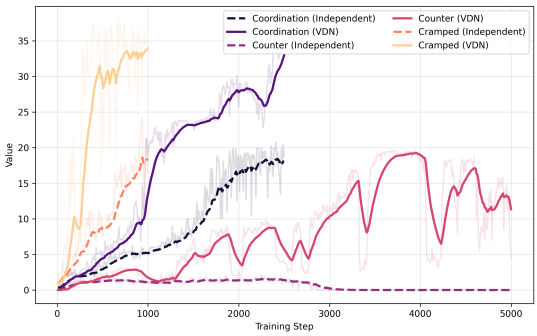
(d) Reward Agent-2 Training



(e) Reward Total Training



(f) Total Dish Pick-up



(g) Total Onion Potting

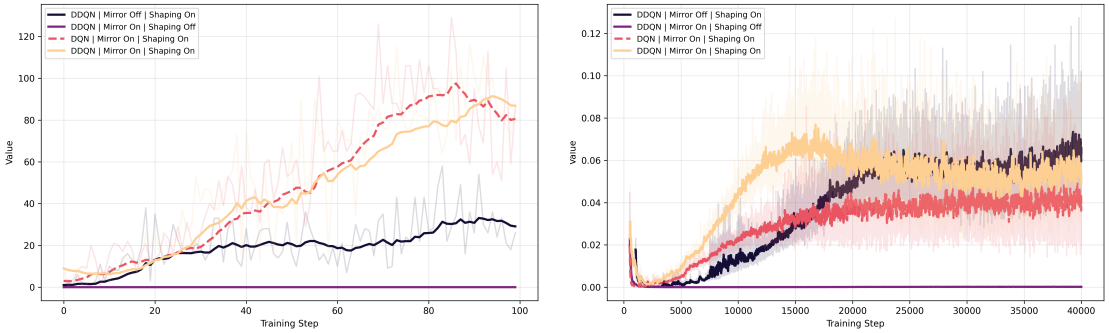
Figure 3: Comparative training and evaluation results for baseline (IQL-DDQN) and VDN-DDQN approaches across three layouts. Top row: soups delivered per episode (left: training, right: evaluation averaged every 10 episodes). Middle row: agent-specific and combined shaped rewards. Bottom row: detailed action metrics—dish pickups (left) and onion placements (right)—highlighting differences in agent behaviour and coordination across layouts.

A complete summary of the final training and evaluation results for the number of Soups-made can be found in Table 3. The VDN-DDQN approach showed a clear advantage over the baseline IQL in more straightforward layouts (*Cramped Room* and *Coordination Ring*), surpassing the benchmark of 7 soups delivered. This improvement indicates that joint-value decomposition successfully encourages cooperative strategies, allowing agents to coordinate better actions like onion placement and dish management (Sunehag et al., 2017; Rashid et al., 2018). On the other hand, in the most complex layout (*Counter Circuit*), VDN-DDQN struggled to complete the 7-soup delivery objective despite showing steady growth in cumulative rewards. This difference between increasing shaped rewards and the lack of soup deliveries suggests two possible explanations:

- The agents may still be in the early stages of effectively learning the task sequences required for successful soup completion. Given that the shaped rewards show a consistent upward trend, additional training episodes could allow agents to eventually overcome coordination challenges and complete tasks by increasing the opportunity for agents to explore.

	Cramped			Coordination			Counter		
Scenario	**Eps**	**Train**	**Test**	**Eps**	**Train**	**Test**	**Eps**	**Train**	**Test**
Independent	1000	4	3	2500	0	0	5000	0	0
VDN	1000	11	12	2500	14	14	5000	0	0
Curriculum (VDN)	x	x	x	x	x	x	1100	8	7

Table 3: Train and Test Soups-made results summary for each layout across methods



(a) Reward (b) Loss

Figure 4: Episode reward trajectories (left) and corresponding loss curves (right) for different algorithm configurations in the *Cramped Room* layout. Curves are smoothed using an exponential moving average ($\alpha = 0.90$).

- Alternatively, agents may be *reward gaming* by repeatedly exploiting intermediate rewards instead of task completion. For instance, agents repeatedly place onions or pick up dishes without progressing toward serving completed soups. This outcome indicates a potential pitfall of poor reward shaping, where intermediate rewards can dominate the sparse task-completion rewards.

Ultimately, the consistent increase in shaped rewards without actual soup completion suggests that the current reward structure may incentivize short-term gains over full-task completion (Albrecht et al., 2024). Adjusting the relative magnitude of rewards or introducing additional shaped rewards (such as spatial distance to objectives) might help guide agent behaviour to complete the whole task sequence. While VDN-DDQN clearly outperforms the baseline in simpler layouts, its reduced effectiveness in more complex scenarios shows the need for further refinements. These could include adjusted reward shaping, extended training durations, or more sophisticated learning frameworks (see Section 5.5).

Fortunately, introducing curriculum learning significantly improved agent performance in the most complex layout. Initially, training the VDN-DDQN model from scratch failed to deliver even a single soup after 5000 episodes. However, by initializing agents with policies pretrained on the simpler *Coordination Ring* layout, agents reached the benchmark of seven delivered soups within approximately 1100 episodes. This substantial reduction highlights curriculum learning's effectiveness in overcoming exploration barriers and enabling agents to transfer previously learned cooperative strategies to more challenging scenarios. While effective, the optimal curriculum path is uncertain. Therefore, to better understand which algorithmic components contributed most significantly to these performance improvements, an ablation study was conducted.

5.3 Ablation Analysis

An ablation study was performed using the *Cramped Room* layout to better understand the contributions of different algorithm components. Figure 4 presents the episode rewards and corresponding loss curves across the different configurations. Several important observations can be made from this analysis:

- Symmetric Buffer Impact: Including symmetric experiences significantly improved episode rewards. This improvement likely results from effectively doubling the experience coverage and ensuring symmetric state-action pairs are well represented. This method is particularly useful in structured environments like *Overcooked*, where symmetric situations are apparent, leading to better generalization and sample efficiency.
- Reward Shaping Influence: Reward shaping significantly impacted performance compared to setups without it. The original reward structure is highly sparse, providing rewards only on successful soup delivery, which makes training challenging. Without shaping, early episodes often provide no learning signals. This causes loss values to quickly drop toward zero and learning to stall. Reward shaping helps by offering informative intermediate feedback, keeping the loss gradient active and helping agents learn the task sequence needed to complete soups.
- DQN vs. DDQN Effectiveness: No significant performance difference appeared between DQN and DDQN. This could be because the *Cramped Room* layout is relatively simple and deterministic, which reduces the impact of DDQN's main advantage, overcoming Q-value overestimation (van Hasselt et al., 2015). In this low-complexity scenario, it is likely that the standard DQN already produces relatively stable estimates. Unfortunately, this makes the added benefits of DDQN less obvious.

Overall, these ablation results demonstrate just how important careful reward design and effective experience replay strategies are. They also suggest that certain algorithmic improvements, like DDQN, might show clearer benefits in more complex or dynamic environments. Insights like these help target specific areas for future refinement, aiming to further improve performance in multi-agent cooperative tasks.

5.4 General Challenges

This study highlighted significant progress and several persistent challenges within MARL. The baseline IQL-DDQN method notably struggled due to its inability to explicitly model agent interactions, complicating accurate credit assignment—an established challenge in cooperative MARL (Sunehag et al., 2017). While VDN-DDQN excelled in simpler layouts, its performance dropped sharply in more complex scenarios like the Counter Circuit, emphasizing difficulties in handling complexity, coordination scalability, and sparse rewards (Rashid et al., 2018). Careful reward shaping is essential, as overly aggressive shaping can lead agents to exploit intermediate rewards rather than achieving primary goals. Although symmetric replay buffers improved sample efficiency, challenges persisted due to limited exploration and diversity of experiences, suggesting a need for more robust exploration or enhanced replay strategies. Curriculum learning showed clear benefits, yet determining the optimal training sequence and duration remains challenging. Addressing these issues is vital to enhance MARL's applicability in real-world scenarios.

5.5 Future Work

Several directions could improve agent performance within complex multi-agent tasks like *Overcooked*. Exploring advanced non-linear MARL frameworks (QMIX or MAPPO), could better capture complex interactions and enhance agent coordination. Additionally, incorporating algorithmic enhancements such as prioritized experience replay, multi-step learning, and distributional reinforcement learning (RL) may help improve sample efficiency, convergence speed, and overall performance stability. Furthermore, adopting potential-based reward shaping techniques, specifically incorporating spatial information or task-related progress, might provide clearer guidance, alleviating potential issues related to sparse and delayed rewards. Finally, extending training durations would allow agents more extensive exploration opportunities, helping them overcome initial coordination barriers to fully master the required sequences of tasks. These recommendations offer pathways toward more robust, generalized, and effective MARL systems for complex collaborative environments.

6 Conclusion

Experiments confirmed the importance of well-tuned hyperparameters, collaborative value decomposition, and carefully designed reward structures within the Overcooked environment. Although baseline IQL-DDQN showed limited cooperation, VDN-DDQN provided strong performance on simpler tasks, underscoring the benefits of reward shaping and symmetric replay buffers. However, persistent difficulties in complex scenarios highlight the need for advanced MARL approaches such as curriculum learning, refined reward-shaping strategies, and improved exploration methods. These findings lay the groundwork for future enhancements that aim to achieve robust coordination and effective task completion in complex multi-agent environments.

References

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2019.

Stefano V. Albrecht, Filippos Christianos, and Lukas Schäfer. *Multi-Agent Reinforcement Learning: Foundations and Modern Approaches*. MIT Press, 2024. URL https://www.marl-book.com.

Micah Carroll, Rohin Shah, Mark K. Ho, Thomas L. Griffiths, and Anca Dragan. Overcooked-ai: A benchmark environment for fully cooperative human-ai task performance. https://github.com/HumanCompatibleAI/overcooked_ai, 2019a. Accessed: 2024-03-21.

Micah Carroll, Rohin Shah, Mark K. Ho, Thomas L. Griffiths, Sanjit A. Seshia, Pieter Abbeel, and Anca Dragan. On the utility of learning about humans for human-ai coordination, 2019b. URL https://arxiv.org/abs/1910.05789.

Hyeonah Kim, Minsu Kim, Sungsoo Ahn, and Jinkyoo Park. Symmetric replay training: Enhancing sample efficiency in deep reinforcement learning for combinatorial optimization, 2024. URL https://arxiv.org/abs/2306.01276.

Yijiong Lin, Jiancong Huang, Matthieu Zimmer, Yisheng Guan, Juan Rojas, and Paul Weng. Invariant transform experience replay: Data augmentation for deep reinforcement learning, 2020. URL https://arxiv.org/abs/1909.10707.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning, 2013. URL https://arxiv.org/abs/1312.5602.

Sanmit Narvekar, Bei Peng, Matteo Leonetti, Jivko Sinapov, Matthew E. Taylor, and Peter Stone. Curriculum learning for reinforcement learning domains: A framework and survey, 2020. URL https://arxiv.org/abs/2003.04960.

Tabish Rashid, Mikayel Samvelyan, Christian Schroeder de Witt, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning, 2018. URL https://arxiv.org/abs/1803.11485.

Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinicius Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Z. Leibo, Karl Tuyls, and Thore Graepel. Value-decomposition networks for cooperative multi-agent learning, 2017. URL https://arxiv.org/abs/1706.05296.

Hado van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning, 2015. URL https://arxiv.org/abs/1509.06461.

A Available Repository

All code and supplementary materials used in this research are available in a GitHub repository. The repository can be accessed at: https://github.gatech.edu/gt-omscs-rldm/7642RLDMSpring2025kodendaal3 where the latest commit hash is: