Beyond Selfishness: Modeling Cooperative AI in Social Dilemmas

Khanh Do*
Grinnell College
Grinnell, Iowa, USA
dokhanh@grinnell.edu

Joyce Gill*
Grinnell College
Grinnell, Iowa, USA
gilljoyc@grinnell.edu

Nicole Moreno González*
Grinnell College
Grinnell, Iowa, USA
morenoni@grinnell.edu

Abstract

Classical Q-learning performs poorly in many multi-agent social dilemmas because each agent treats the other agents as part of a stationary environment, even though all agents are simultaneously learning and changing behavior. This mismatch often leads self-interested learners to converge to low-payoff Nash equilibrium rather than Pareto-efficient cooperative outcomes. In this paper, we reproduce the core findings of Learning to Cooperate in Multi-Agent Social Dilemmas [4] and evaluate three modifications to standard Q-Learning: Change or Learn Fast (CoLF), which adapts the learning rate in response to payoff variability; Change and Keep (CK), which delays updates after an action changes so that other agents have time to react; and CK-CoLF, a hybrid of the two. Our reproduction recovers the main trend of the original work. Q-Learning exhibits a strong tradeoff between convergence speed and final payoff, CK substantially improves cooperative stability, CoLF improves convergence behavior but does not reliably reach Pareto-efficient outcomes, and the hybrid CK-CoLF offers the best overall balance between speed and cooperative performance. We conclude by detailing an extension that maps the abstract social dilemma into a Webots traffic-coordination setting [3].

1 Introduction

Multi-agent reinforcement learning (MARL) is difficult not only because the state-action space grows quickly with the number of agents, but also because each agent's learning process changes the environment experienced by the others. In fully self-interested repeated interactions, this non-stationarity often pushes independent learners toward myopic or unstable behavior. This problem is especially important in social dilemmas, situations where each agent has an incentive to choose the action that benefits itself in the short term, even though repeated self-interested choices lead to worse outcomes for the group as a whole [1]. As a result, agents that could all benefit from cooperation may instead settle for individually rational but collectively inferior outcomes.

This paper studies that problem through a reproduction of the paper Learning to Cooperate in Social Dilemmas [4], which introduced two simple but influential design principles for cooperative learning in repeated multi-agent social dilemmas: CoLF and CK. Also, the hybrid CK-CoLF, which combines both ideas.

Our project had two goals. First, we aimed to reproduce the paper's central results in the Multi-Agent Social Dilemma (MASD) benchmark using a clean, object-oriented Python implementation. Second, we aimed to extend the abstract matrix-game setting into a more embodied scenario using Webots, where agents correspond to autonomous vehicles navigating a shared traffic dilemma. Thus, our central research question is:

> How can the CoLF and CK principles be integrated into standard Q-Learning to help self-interested agents in social dilemmas move toward Pareto-efficient outcomes instead of converging to sub-optimal Nash equilibria?

Our paper has three main contributions. First, it provides a transparent reimplementation of the MASD environment and the four principles studied in the original paper. Second, it shows that the main trends reported in the 2006 work are reproducible, while also revealing that stochastic control matters for fair comparisons. Third, it presents a concrete extension path from abstract cooperative games to an interpretable robotics-style setting in which the same algorithmic principles can be tested under richer dynamics.

2 Related Work

The most immediate prior work is Muñoz de Cote *et al.* [4], who introduced CoLF and CK as lightweight principles for improving cooperation among independent Q-learners in repeated social dilemmas. Their contribution is notable because it addresses cooperation without requiring explicit communication, centralized control, or a complex model of the other agents. Instead, the paper focuses on modifying the update dynamics of standard Q-learning so that agents respond more appropriately to the non-stationarity created by simultaneous learning.

From a modern perspective, these methods can be viewed as early, interpretable precursors to later MARL approaches. One influential example is Learning with Opponent-Learning Awareness (LOLA), which models how one agent's update will shape the future learning of another [6]. Unlike CK and CoLF, which modify the timing or aggressiveness of Q-learning updates, LOLA directly incorporates anticipated opponent learning into the optimization process. This makes it a stronger but also more complex approach than the minimalist principles we studied.

Another closely related line of work is approximate Markov Tit-for-Tat (amTFT), which approaches cooperation through reciprocity rather than update control [7]. amTFT is designed to preserve cooperation in sequential social dilemmas by cooperating with cooperative agents and retaliating against defectors. Compared with CK and CoLF, amTFT is less about making independent learning more stable and more about enforcing cooperative behavior through contingent responses. This makes it conceptually different, but still highly relevant, because it addresses the same underlying question of how self-interested agents can avoid converging to socially inefficient outcomes.

These later methods highlight how the field has evolved from simple tabular mechanisms for coping with non-stationarity to richer frameworks based on opponent modeling, reciprocity, and strategic foresight. At the same time, they also make clear why

*All authors contributed equally to this research.

CK and CoLF remain useful. Their main value is not that they outperform all modern methods, but that they isolate two simple and interpretable principles for improving cooperation under simultaneous adaptation. As a result, they continue to serve as valuable baselines for understanding the basic learning dynamics that underlie cooperation in multi-agent systems.

Because this project is primarily a reproduction study, we do not position our work as a direct comparison against state-of-the-art cooperative MARL methods. Instead, we use the original 2006 paper as the main comparison target and treat broader comparisons to methods such as LOLA and amTFT as important future work.

3 Methods

Our implementation and experimental framework are publicly available on GitHub [5].

3.1 Environment

We implemented the MASD environment described in the original paper. Our setup uses three agents ($N = 3$) and three resource units per agent ($M = 3$), yielding four discrete actions corresponding to contributions $\{0, 1, 2, 3\}$. Choosing $N = 3$ creates a medium-sized social dilemma that is not as simple as a 2-player game and can have the necessary non-staionary noise that we want to test the algorithms on. $M = 3$ also moves the problem from a binary yield-or-push into a more complex one where the agents can choose levels of cooperation.

The reward function follows the formulation defined in the source paper:

$$P_i(\mathbf{a}) = \frac{\left[\frac{1}{N}\sum_{j=1}^{N} a_j\right] - k a_i}{M(1-k)},$$

where $\mathbf{a} = [a_1, \ldots, a_N]$ is the joint action, a_i is the amount contributed by agent i towards a group goal G, and k is the selfishness factor that determines the "cost" of contributing resources. We use $k = \frac{2}{3}$, matching the main experiment discussed in the source paper. When $k = \frac{2}{3}$, a unit of resource is twice as costly to the individual as the average benefit returned to the group. This specific choice of k ensures that the agents face a true social dilemma where the selfish Nash Equilibrium (contributing 0) is collectively worse than the cooperative state (everyone contributing 3), which results in a maximum payoff of 1.0 for the entire group.

This environment in which agents can observe the other agents' previous actions but do not communicate directly allows agents to treat the other agents as part of a dynamic environment and attempt to learn a best-response to their current policies.

3.2 Principles

Our codebase implements four principles:

3.2.1 Q-Learning. Q-Learning is a model-free reinforcement algorithm that enables an agent to learn optimal actions through trial-and-error. It uses a Q-table to store expected rewards for state-action pairs, and updates them iteratively using the Bellman equation.

3.2.2 CoLF. CoLF modifies Q-Learning by using a variable learning rate. The key idea is to adjust learning rates—slow down when other agents are unpredictable, and speed up when stable. To do this, the agent tracks an estimate of expected payoffs and reward variability for each state-action pair. If the current reward deviates more than expected, the environment is treated as non-stationary and the slower learning rate is used. Otherwise, the faster learning rate is applied.

3.2.3 CK. CK introduces a delayed-update mechanism. When an agent changes its action, it does not immediately update its Q-table. Instead, it repeats the new action update the Q-value only after observing the next reward. The motivation is that the first reward after an action change may be misleading because the rest of the system has not yet had time to respond. By delaying the update, CK seeks to make the feedback more informative.

3.2.4 CK-CoLF. The hybrid CK-CoLF combines the two principles. It uses CK's deferred-update mechanism to determine when an update should occur and CoLF's variable learning-rate logic to determine how quickly the update should be applied. Conceptually, the hybrid aims to use CoLF to speed up convergence rate and CK to help achieve average payoff of 1.0.

3.3 Experimental Setup

The reproduction targets the learning-rate experiment corresponding to Figure 4 in the original paper. We use the following settings:

- 100 independent trials per condition
- 200,000 episodes per trial
- discount factor $\gamma = 0.95$
- epsilon-greedy exploration with linear decay from 0.2 to 0: $\epsilon = max(0.2 - 0.00006t, 0)$
- optimistic Q-table initialization with $V_{max} = \frac{1}{(1-\gamma)} = 20$
- learning-rate sweeps for standard Q-Learning and CK over $\alpha \in \{0.1, 0.2, 0.4, 0.8\}$
- learning-rate decays as following: $\alpha^t = \frac{a_i}{1+0.0001 \cdot n(\mathbf{a}^{t-1}, \mathrm{a}_i^t)}$, where $n(\mathbf{a}^{t-1}, \mathrm{a}_i^t)$ is the number of times that action a_i has been taken after after the joint action $\mathbf{a}^{t-1}$
- CoLF and CK-CoLF settings with $\alpha_{NS} = 0.1$, $\alpha_S = 0.4$, and weight factor $\lambda = 0.1$

3.4 Software Architecture

Our implementation follows a modular architecture designed to support reproducibility, extensibility, and fair comparison between learning algorithms. The system is organized into three primary components: Learning Algorithms, MASD Environment, and Experiment & Data Visualization Pipeline

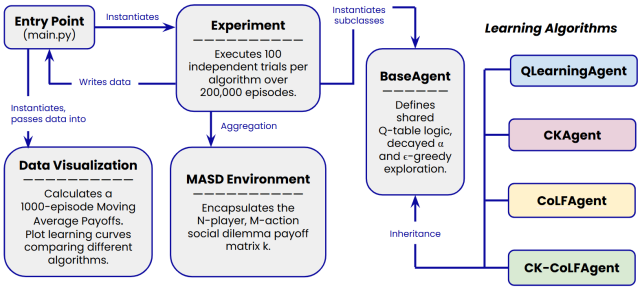

Figure 1: File Architecture Map. `main.py` acts as the entry point, coordinating the flow between setting up the MASD Environment, running the simulations via the Experiment class, and visualizing the data via the Data Visualization class.

3.4.1 Learning Algorithms. Each reinforcement learning strategy is implemented as a subclass of a shared `BaseAgent` interface. This abstraction defines the core methods required for interaction with the environment, including action selection, reward processing, and Q-value updates. Specific algorithms such as Q-Learning, CK, CoLF, and CK-CoLF override these methods to implement their respective learning rules while preserving a consistent interface.

3.4.2 MASD Environment. The environment module implements the Multi-Agent Social Dilemma (MASD) game dynamics. It manages the global state, computes rewards from joint actions, and provides observations to agents. This separation allows the same learning algorithms to be evaluated across different environments without modification.

3.4.3 Experiment & Data Visualization Pipeline. The experiment controller orchestrates repeated trials, initializes agents, sets random seeds, and records performance metrics. It ensures that each algorithm is evaluated under identical conditions, enabling controlled comparisons. The pipeline also aggregates results across trials and generates summary statistics for visualization.

3.5 Reproducibility Design

One of the most important lessons from our reproduction was methodological rather than algorithmic. In earlier runs, one subplot–our CoLF comparison–behaved inconsistently with the argument of the source paper. To investigate this, we strengthened the experiment by explicitly seeding each independent trial with a deterministic offset from a base seed. We fully acknowledge that a seed should not determine if an algorithm should win or lose. Instead, this ensured that each trial used its own random generator while preserving fairness and repeatability across algorithms. This change did not alter the implementation of CK, CoLF, or CK-CoLF themselvces. Instead, it improved the reliability of comparisons by reducing ambiguity about whether observed differences were due to learning dynamics or uncontrolled randomness.

3.6 Webots Extension

A natural next step is to move from theory to environments where cooperation has a more physical interpretation [2, 8]. Therefore, we began an extension in Webots, an open-source robot simulator, to represent the same underlying dilemma through autonomous vehicles in a traffic scenario.

In the original MASD setting, agents decide how much private resource to contribute to a shared good. In traffic coordination, autonomous vehicles face an analogous problem where selfish behavior can create globally inferior outcomes: blocking an intersection, refusing to yield, or acting too aggressively may improve short-term progress for one vehicle, while worsening the outcome or safety for all.

Our extension architecture preserves the learning-agent abstraction from the reproduction. Webots instantiates the traffic scenario, while the learning logic remains separated across controller files. The car controller connects each vehicle to its assigned learning algorithm and sends action/state information from the car to the rest of the system. The supervisor controller manages the shared environment, receives the agents' joint actions, computes the episode outcome, and reports the reward tracker. This makes it possible to compare Q-Learning, CK, and CK-CoLF under the same coordination logic while changing only the environment and reward semantics. Modeling these agents through a traffic dilemma helps translate the cooperative mechanisms studied in an abstract repeated game into a setting where the consequences of selfishness and cooperation are easier to visualize and interpret.

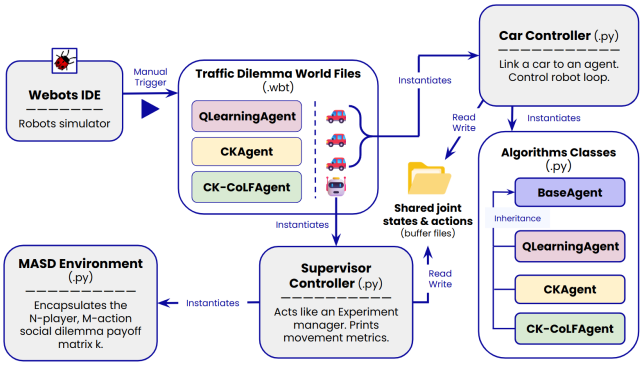

Figure 2: Webots Extension Architecture Map. The Webots interface triggers the world file, which sets up the uncontrolled 4-way intersection with three car nodes and a supervisor node. Each car is connected to its learning algorithm through the car controller, while the supervisor controller manages the shared Multi-Agent Social Dilemma (MASD) environment, reward tracker, and communication through buffer files.

4 Results

4.1 Standard Q-Learning

The Q-Learning baseline reproduces the central tradeoff highlighted in the original paper.

4.2 Change-and-Keep (CK)

CK provides a clear improvement over standard Q-Learning. Across learning rates, CK converges to substantially higher payoffs and does so more robustly than the baseline. At lower learning rates

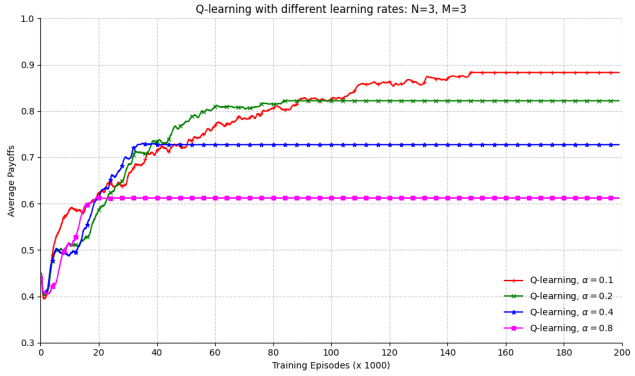

Figure 3: Q-Learning with variable learning rates: N=3, M=3. The graph is comparing Q-Learning average payoffs across different fixed learning rates and showing that lower learning rates yield better converged payoffs, but they converge much more slowly. Higher learning rates converge more quickly, but they stabilize at worse outcomes.

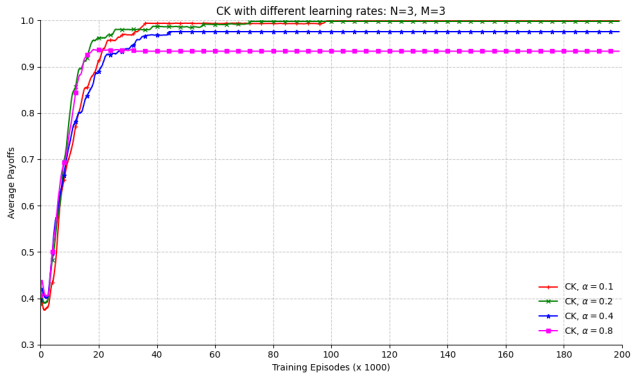

Figure 4: Change-and-Keep (CK) with variable learning rates: N=3, M=3. The graph plots the average payoffs of CK across different fixed learning rates and shows that at lower learning rates ($\alpha = 0.1$), it approaches the Pareto-efficient solution (Average Payoffs = 1.0).

and even at higher learning rates, it remains far more stable than standard Q-Learning.

These results support the intuition behind CK. By delaying updates after an action change, the algorithm allows the rest of the system to react before the new action is evaluated. This makes the reward signal more informative and reduces the chance that an agent updates from misleading transitional feedback.

4.3 Change-or-Learn Fast (CoLF)

CoLF, relative to fixed-rate Q-Learning, benefits from being able to slow down when rewards are unexpectedly volatile and speed up when they become more stable. This allows it to achieve a better balance between convergence speed and payoff quality than a single constant learning rate.

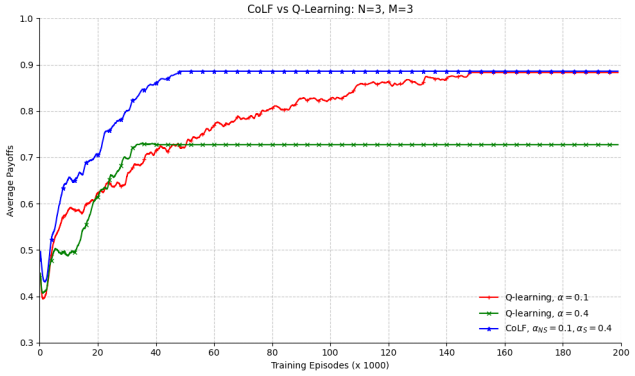

Figure 5: Change-or-Learn Fast (CoLF) vs Q-Learning: N=3, M=3. A graph of CoLF showing that it converges faster than Q-Learning but still does not by itself reliably reach the Pareto-efficient solution.

4.4 Change-or-Keep and Change-or-Learn-Fast (CK-CoLF)

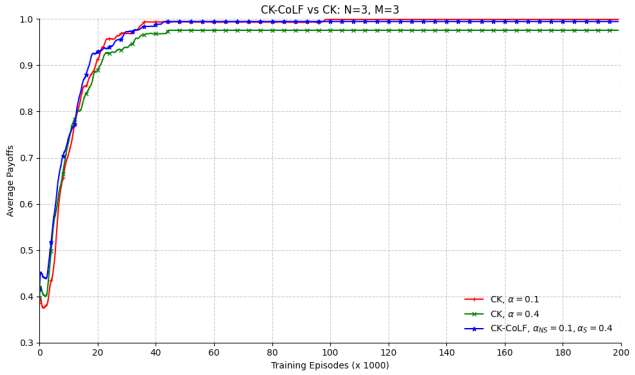

Figure 6: Hybrid model of Change & Keep and Change-or-Learn-Fast (CK-CoLF) vs Change-and-Keep (CK): N=3, M=3. A graph of CK-CoLF showing that it reaches high-payoff cooperative behavior faster than the strongest standalone CK baseline while remaining more stable than traditional Q-Learning.

CK-CoLF achieves the strongest overall balance among the methods we studied. It combines CK's ability to approach Pareto-efficient performance with CoLF's capacity to improve learning speed.

4.5 Webots

Our final product is a partially working Webots simulation environment for self-driving in a Traffic Social Dilemma: Uncontrolled 4-Way Intersection, implemented across three different worlds, one for each algorithm. The simulation runs smoothly for up to 60 episodes. The figures below shows the final composition of the CK-CoLF world, including the object tree and three progressive state images from early episodes of an initial simulation run. The partial functionality will be discussed later in the limitations section.

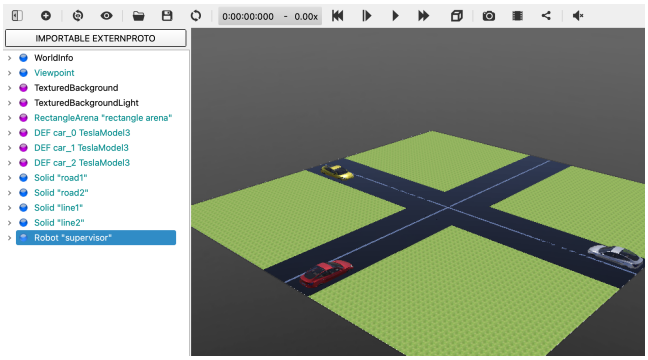

Figure 7: Webots Object Tree. The cars in this scene, which are all Change & Keep and Change-or-Learn-Fast agents, are placed at an uncontrolled 4-way intersection. The objects shown in the object tree are set up in Webots-specific World Files.

Figure 8: Three Hybrid Change & Keep and Change-or-Learn-Fast agents enacting behavior that leads to a 'Crash' outcome. In early episodes, car agents are still exploring and will often fail to appropriately reduce speed, which is acting 'selfish', leading to crashing into each other and a low collective payoff.

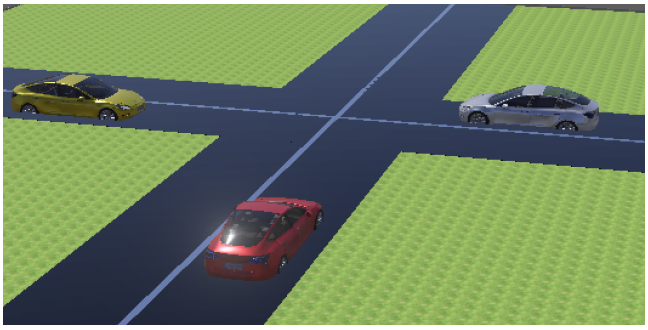

Figure 9: Three Hybrid Change & Keep and Change-or-Learn-Fast agents are learning to regulate their speed. The red and silver car agents are slower compared to the yellow car agent.

Figure 10: Three Hybrid Change & Keep and Change-or-Learn-Fast agents chose an efficient speed adjustment, which led to a 'Successful/ No crash' outcome. All car agents pass through the intersection and collectively reach an efficient outcome.

5 Discussion

This reproduction helps clarify the different roles that CK and CoLF play in cooperative learning. Following the original paper, CoLF is primarily designed to address non-stationarity caused by the other agents: it learns slowly when payoffs are unexpectedly changing and more quickly when the environment appears stable. CK, on the other hand, targets the non-stationarity created by the learner's own action changes. In the original work, Muñoz de Cote et al. argue that these principles are complementary because one manages uncertainty about other agents, while the other manages uncertainty induced by the agent itself.

Our findings broadly support that interpretation, but they also suggest an important qualification. In our reproduction, CK emerges as the more robust of the two mechanisms. It consistently improves cooperative performance, approaches Pareto-efficient outcomes, and remains stable across learning rates. CoLF, on the other hand, improves learning dynamics without reproducing the full strength of the original paper's claim. In our results, CoLF is better understood as a stabilizing mechanism that smooths learning and improves the speed-performance tradeoff rather than as a standalone route to strong cooperation.

This difference is especially important because our reproduced CoLF result does not fully match the original paper. In our experiments, CoLF does not reliably reach the Pareto-efficient solution. Instead, it improves convergence behavior without by itself guaranteeing full cooperative convergence. This suggests that CoLF may be more sensitive than CK to implementation details, stochastic control, hardware differences, or other experimental assumptions. Put differently, our reproduction still supports the intuition behind CoLF, but it supports that intuition more modestly than the original paper does.

More broadly, this project highlights the importance of methodological care in reproduction work. Multi-agent systems are highly sensitive to exploration noise and interaction effects, so even small differences in randomness handling or implementation details can alter the qualitative interpretation of an algorithm.

Finally, these results motivate the Webots traffic extension in a meaningful way. The traffic dilemma offers a richer setting in which

the distinction between external and self-induced non-stationarity may matter even more. In autonomous driving, an agent must respond both to unpredictable behavior from other vehicles and to the consequences of its own sudden action changes. For that reason, traffic coordination provides a natural next testbed for examining whether CK, CoLF, and CK-CoLF remain effective when cooperation is embedded in a more physical and socially interpretable environment.

6 Limitations and Future Work

Our study had several limitations. First, we focused primarily on reproducing the learning-rate experiment rather than the full sweep of discount-factor and selfishness-factor studies from the original paper. Second, the study does not include a formal statistical analysis of differences between methods. Although the qualitative trends are clear, stronger quantitative evidence would come from reporting confidence intervals, variance measures, or significance tests. Third, the Webots simulator is not fully functional since the Supervisor exits around episode 60 pausing the simulation for reasons that require deeper exploration of the Webots interface. Fourth, we reproduced only the algorithms studied in the original paper and did not compare them against stronger contemporary MARL baselines. As a result, our conclusions are strongest as a reproduction of the original work rather than as a statement about current state-of-the-art cooperative learning.

There are several promising directions for future research. First, this reproduction should be expanded to include the additional experiments in the original paper, especially the discount-factor and k-sensitivity analyses. This would provide a more comprehensive view of when CK, CoLF, and CK-CoLF remain effective. Second, the Webots extension should be completed with a fully specified traffic dilemma, clear reward design, and shared metrics such as delay, throughput, and collision avoidance. Additionally, the simulation must be troubleshooted by building more familiarity with the software to allow a complete run of the controllers. Third, the methods studied here should be compared against stronger contemporary cooperative MARL baselines. Such comparisons would clarify whether CK and CK-CoLF remain competitive in richer environments. or whether their primary value today lies in interpretability and robustness.

7 Conclusion

We reproduced the four learning principles described in the paper Learning to Cooperate in Multi-Agent Social Dilemmas [4]: standard Q-Learning, CoLF, CK, and CK-CoLF. Our results confirm that standard Q-Learning struggles in multi-agent social dilemmas, CK strongly improves cooperative convergence, CoLF improves the speed-stability tradeoff without reliably reaching Pareto efficiency on its own, and CK-CoLF provides the strongest overall combination of speed and cooperative performance in our experiments.

More broadly, this project reinforces a simple but important lesson: in multi-agent social dilemmas, better cooperation does not necessarily require more complex agents; it requires better rules for deciding when a reward is trustworthy and how quickly learning should respond to it. The traffic-dilemma extension in Webots provides a practical visual of this work. By applying theoretical multi-agent social dilemmas to a traffic dilemma, it offers a promising path for testing whether the same principles that improve cooperation in abstract social dilemmas can also help agents coordinate in real-world environments.

Acknowledgments

This work was completed as part of the CSC 261: Artificial Intelligence course under Professor Fernanda Eliott. Thank you to Professor Eliott for your guidance, support, and supervision of the project.

Thank you to our classmates for providing us with helpful feedback on our project progress presentations and this paper.

References

[1] American Psychological Association. 2018. Social dilemma. *APA Dictionary of Psychology*. https://dictionary.apa.org/social-dilemma

[2] A. Bodenschatz. 2025. Navigating the social dilemma of autonomous systems: normative and applied arguments. *Ethics and Information Technology* 27, 64 (2025). doi:10.1007/s10676-025-09857-y

[3] Cyberbotics Ltd. 2025. Webots: Open Source Robot Simulator. https://cyberbotics.com/

[4] Enrique Munoz de Cote, Alessandro Lazaric, and Marcello Restelli. 2006. Learning to cooperate in multi-agent social dilemmas. In *Proceedings of the Fifth International Joint Conference on Autonomous Agents and Multiagent Systems* (Hakodate, Japan) *(AAMAS '06)*. Association for Computing Machinery, New York, NY, USA, 783–785. doi:10.1145/1160633.1160770

[5] Khanh Do, Joyce Gill, and Nicole Moreno González. 2026. MAS Cooperation: Modeling Cooperative AI in Social Dilemmas. https://github.com/khanhdo05/mas-cooperation.

[6] Jakob N. Foerster, Richard Y. Chen, Maruan Al-Shedivat, Shimon Whiteson, Pieter Abbeel, and Igor Mordatch. 2018. Learning with Opponent-Learning Awareness. arXiv:1709.04326 [cs.AI] https://arxiv.org/abs/1709.04326

[7] Adam Lerer and Alexander Peysakhovich. 2018. Maintaining cooperation in complex social dilemmas using deep reinforcement learning. arXiv:1707.01068 [cs.AI] https://arxiv.org/abs/1707.01068

[8] E. Mumba, A. Sulaiman Gezawa, and C. Liu. 2025. Enhanced autonomous driving within Webots simulation for student experiments. *Intelligent Service Robotics* 18 (2025), 875–895. doi:10.1007/s11370-025-00608-y