algorithms

MDPI

Article

A Multi-Agent Advisory Board Reinforcement Learning Framework for Adaptive Cooperative Control

Onur Osman [1], Tolga Kudret Karaca [2,*], Bahar Yalcin Kavus [3], Gokalp Tulum [1] and Sajjad Nematzadeh [4]

[1] Department of Electric Electronics Engineering, İstanbul Topkapi University, 34087 Istanbul, Türkiye; onurosman@topkapi.edu.tr (O.O.); gokalptulum@topkapi.edu.tr (G.T.)
[2] Department of Industrial Engineering, İstanbul Topkapi University, 34087 Istanbul, Türkiye
[3] Quality Coordination Office, İzmir Katip Çelebi University, 35620 Izmir, Türkiye; bahar.yalcin.kavus@ikc.edu.tr
[4] Department of Computer Engineering, İstanbul Topkapi University, 34087 Istanbul, Türkiye; sajjadnematzadeh@topkapi.edu.tr
* Correspondence: tolgakudretkaraca@topkapi.edu.tr

Abstract

This study proposes Advisory Board Reinforcement Learning (AdvB-RL), a cooperative reinforcement-learning framework that integrates multiple advisory neural networks to guide policy optimization. Unlike conventional single-agent architectures, AdvB-RL maintains a set of independently trained advisory networks that contribute to action selection through a dynamic aggregation mechanism. This design preserves diverse experiential knowledge while improving learning stability and the exploration–exploitation balance. The framework is evaluated on three benchmark control tasks, namely LunarLander-v2, CartPole-v1, and MountainCar-v0, using advisory board sizes of 1, 5, and 10 members against a Double Deep Q-Network (DDQN) baseline. The best-performing configuration, 10 AdvB, achieved 270.02 $\pm$ 24.74 on LunarLander-v2 versus 227.92 $\pm$ 86.02 for DDQN, 497.79 $\pm$ 5.18 on CartPole-v1 versus 304.37 $\pm$ 144.04, and -103.16 ± 15.46 on MountainCar-v0 versus -130.71 ± 31.64, indicating higher returns together with markedly lower variability. Across the three environments, these results show that increasing the number of advisory members improves both reward consistency and overall robustness, with the 10-member setting providing the strongest performance. Within the tested configurations, the advisory board mechanism remains computationally feasible, while preliminary experiments beyond 10 advisors show diminishing returns relative to added complexity. Overall, AdvB-RL provides a robust and modular alternative to single-policy reinforcement learning for adaptive cooperative control.

Keywords: control environments; double deep Q network; advisory board reinforcement learning; lunar lander; cart pole; mountain car

Check for updates

Academic Editor: Predrag S. Stanimirović

Received: 17 February 2026
Revised: 16 March 2026
Accepted: 17 March 2026
Published: 18 March 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

1. Introduction

The challenge of achieving a successful lunar landing involves guiding a spacecraft to the lunar surface with precision and safety [1,2]. This complex task has been tackled using various methods, including reinforcement learning (RL), deep learning, and traditional control techniques [3–5]. The Lunar Lander problem is a significant testbed for these methods due to its complex environment characterized by continuous state, action spaces, and unknown system dynamics [2]. Reinforcement learning methods are widely used for solving the Lunar Lander problem because they can learn control policies directly through

interaction with the environment [2,3]. These algorithms can learn from interactions with the environment without requiring prior knowledge of the problem, enabling them to adapt to different states and improve decision-making autonomously. RL algorithms can handle varying initial conditions and task objectives, which makes them suitable for the diverse challenges of lunar landing scenarios [6]. In addition, LunarLander-v2 is a demanding benchmark because successful control depends on the coordinated management of descent velocity, orientation, lateral drift, and touchdown timing, while poor actions may lead to crash-sensitive terminal penalties. These properties make the environment suitable not only for measuring final reward, but also for assessing policy stability and robustness during learning.

AdvB-RL is conceptually distinct from several closely related reinforcement learning paradigms. Unlike conventional policy ensemble methods, which typically combine multiple policy or value models through simple averaging or voting, AdvB-RL relies on a single online learner that remains the only component actively interacting with the environment and updating its behavior through experience. The advisory members do not function as active agents in the multi-agent RL sense, since they do not simultaneously explore, cooperate, or compete within the environment; instead, they serve purely as guiding networks. The framework also differs from Bootstrapped DQN, where multiple heads are mainly designed to improve uncertainty estimation and exploration within a shared architecture. In contrast, the advisory networks in AdvB-RL are pre-trained independently and later assembled into an advisory board, rather than being jointly trained as parallel heads of the same learner. Moreover, their influence is not determined by equal-weight aggregation or majority voting, as in many committee-based systems, but by a performance-based weighted guidance mechanism. Therefore, AdvB-RL should be regarded as a guidance-driven single-agent learning framework rather than a standard ensemble, committee, or multi-agent reinforcement learning architecture.

The historical context of lunar landings, such as the Apollo missions, highlights the engineering challenges and technological advancements achieved over the decades. The success of the Apollo Lunar Lander Module, known as the Eagle, set a benchmark for subsequent landing technologies, including those used by SpaceX for their Falcon and Starship rockets [4]. Future missions to Mars will require even more advanced guidance and control capabilities to achieve the necessary landing accuracy for scientific exploration [7]. From a reinforcement learning perspective, this historical and technological context also motivates the study of control strategies that remain reliable under uncertainty and sensitive landing conditions.

Recent developments in machine learning, particularly deep learning, have opened new avenues for developing sophisticated control mechanisms for lunar landings. For instance, the LunarLander-v2 environment developed by OpenAI in the Gym 0.26.2 toolkit provides a simulated setting to test and refine RL algorithms [1,8]. Previous approaches have employed techniques such as Q-learning, Deep Q-learning (DQL), Double Deep Q-learning (DDQL), and model-based methods to address the uncertainties inherent in the environment [9]. However, these methods often fail to simultaneously handle multiple uncertainties and challenges, necessitating more robust solutions because of the Lunar Lander problem's challenging environment [10]. In particular, single-policy value-based methods may exhibit unstable performance when exploration noise, delayed rewards, and uncertain action consequences interact during training. Under such conditions, preserving diverse successful learning experiences through a cooperative advisory mechanism may provide a more robust basis for action selection. The Lunar Lander problem provides a useful benchmark for advancing reinforcement learning and deep learning techniques in challenging sequential control settings. The continued development of such methodologies

may contribute to more robust decision-making strategies for challenging landing and control tasks. Furthermore, the practical applications of these technologies in various fields highlight their versatility for autonomous decision-making and control systems [6,11,12].

By applying AdvB-RL to benchmark environments such as Lunar Lander, CartPole, and Mountain Car, we evaluate its performance against conventional and recent methods such as DDQN and MuZero. The novelty of the proposed approach lies in its cooperative policy optimization mechanism, which enables continuous refinement of decision strategies in complex RL environments. More specifically, AdvB-RL differs from conventional ensemble-style reinforcement learning approaches because it does not rely on a fixed static aggregation of pre-defined models; instead, advisory members are evaluated continuously, but updated at the board-membership level rather than via step-wise gradient optimization. It also differs from conventional multi-agent RL formulations, where multiple agents independently interact with the environment as task-solving entities, because the advisory networks in AdvB-RL cooperatively support a shared decision process rather than acting as separate autonomous agents. This distinction is important for the present study, as the proposed framework is intended to improve learning stability and reduce performance variance while preserving diverse learning trajectories.

The rest of the paper is organized as follows: Section 2 describes related works. Then, we introduce the experiment environments in Section 3. Section 4 presents our proposed novel methodology. Section 5 describes experiments and results. Discussion is presented in Section 6. Finally, Section 7 concludes the research and results.

2. Related Works

RL applications to the Lunar Lander problem have driven significant research, particularly in adapting Q-Learning to continuous state environments. One critical challenge is the overestimation of action values, which introduces a bias in the Q-learner's approximation of maximum expected values [13]. Neural networks and other estimators have been pivotal in estimating value functions without discretizing continuous state spaces. Hado van Hasselt's DDQN addresses this by using two separate estimators to decouple expected value from action value maximization, leading to faster convergence compared to DQN [14,15].

In RL, the interaction between the agent and the environment is fundamental. The environment plays a crucial role by providing state values to the agent, which selects actions based on this information. The agent updates its knowledge based on received rewards, continually evaluating action effectiveness until a terminal state is reached. Balancing exploration (random action selection) and exploitation (action selection based on value function) critically influences learning success [4]. Various RL methods have optimized landing motion in the Lunar Lander problem. Agents continuously update their environment knowledge to perform complex tasks efficiently [16–18].

Several studies have explored approaches to Lunar Lander complexities. Mnih et al. [19,20] demonstrated DQL's applicability in complex environments, while Yu's work with DQL and experience replay highlighted the benefits of a broad, shallow neural network [21]. Lu et al. employed model-based approaches, bypassing system dynamics [8], and Gou and Liu's hybrid model showed improved Mountain Car performance scenarios [9].

Comparative studies between DQN and Policy Gradient methods generally favor DQN due to environmental determinism. Nugroho compared DQL with heuristic Q-Learning, finding heuristic methods advantageous in controlled settings but limited by uncertainties [4]. Gadgil et al. introduced uncertainties into the Lunar Lander, enhancing Sarsa's performance with state discretization and supervised learning [22].

Recent RL advancements include Rusdyputra and Chowanda's data-efficient model integrating episodic memory and data augmentation, showing 50% speed improvement over DQN [23], and Kubo et al.'s Equilibrium Propagation (EP) for training the Actor network and backpropagation (BP) model, demonstrating stable high-reward behavior [24]. Hafiz and Bhat introduced a multi-agent RL system, improving efficiency and convergence [25]. Xu et al.'s Value Difference-Based Deep Sarsa and Q Networks (VBDSQN) dynamically balance exploration–exploitation, outperforming traditional methods [26], while Ma et al. introduce Freshness Prioritized Experience Replay (FPER), which prioritizes fresher experiences, enhancing DRL efficiency [27]. Zheng et al.'s Stochastic Integrated Actor-Critic (SIAC) reduces sample complexity by 50–93% across various tasks, highlighting improved efficiency and performance [28]. Kang et al.'s traditional deep deterministic policy gradient (DDPG) algorithm, based on the DNPER-DDPG, accelerates convergence with prioritized experience replay and double network structure [12]. Additionally, the study of Kurniawan et al. introduces Discrete-to-Deep Supervised Policy Learning (D2D-SPL) and Discrete-to-Deep Supervised Q-value Learning (D2D-SQL), aiming to combine neural network generalizability with tabular RL method efficiency. D2D-SPL demonstrates superior learning speed and performance over other methods in environments like Cartpole and Aircraft Manoeuvring. Meanwhile, D2D-SQL outperforms DDQN agents in complex tasks, particularly excelling in the Aircraft Manoeuvring problem. These methods represent significant advancements in RL by enhancing learning efficiency and effectiveness across diverse RL scenarios [29].

Recent studies have increasingly explored cooperative and multi-agent reinforcement learning strategies for control tasks in both discrete and continuous action spaces. In particular, multi-agent actor–critic approaches, including MADDPG and more recent cooperative actor–critic frameworks, have shown that policy coordination and information sharing can improve adaptation and robustness in dynamic environments [30–32]. Recent control-oriented studies have likewise highlighted the growing importance of cooperative control fusion and coordinated decision-making in practical reinforcement learning applications [33,34]. However, many of these approaches are designed around multiple task-solving agents or policy-level coordination mechanisms, whereas the present study focuses on a cooperative advisory structure that supports a shared decision process through dynamically evaluated advisory members.

To better position the proposed framework within the broader literature, Table 1 summarizes representative cooperative, ensemble, and multi-model reinforcement learning approaches and highlights their conceptual differences from AdvB-RL.

Table 1. Summary of related works.

Approach Family	Core Mechanism	Difference from AdvB-RL
Policy ensemble methods	Multiple policy or value models are combined to produce a final action recommendation, typically through averaging or fixed aggregation rules	AdvB-RL does not perform equal-weight policy aggregation; it uses a single online learner supported by performance-based weighted advisory guidance
Bootstrapped/multi-head value-based methods	Multiple heads are maintained within a shared architecture, mainly to improve uncertainty estimation and exploration	AdvB-RL does not rely on jointly trained heads within a single network; its advisory members are independently pre-trained networks
Committee-based systems	Multiple models provide recommendations, and the final decision is generally obtained through voting, averaging, or other static combination rules	AdvB-RL replaces static voting or averaging with a dynamic performance-based weighting mechanism

Table 1. *Cont.*

Approach Family	Core Mechanism	Difference from AdvB-RL
Multi-agent reinforcement learning [25,30–32]	Multiple active agents interact with the environment, often through cooperation, coordination, or competition	AdvB-RL is not a multi-agent framework, since only one learner interacts online with the environment, while advisory members act only as guiding networks
Cooperative control fusion/coordinated decision frameworks [33,34]	Multiple controllers, policies, or decision modules are fused to improve robustness and adaptation	These approaches typically coordinate multiple task-solving units, whereas AdvB-RL uses pre-trained advisory members to guide a single learner
AdvB-RL (proposed)	A single online learner receives guidance from a board of pre-trained advisory networks through performance-based weighting	Combines single-agent online learning with advisory support, without an equal-weight ensemble or multi-agent interaction

A further limitation in prior benchmark studies is that the weaknesses of baseline methods are often discussed only in general terms, such as overestimation bias or insufficient exploration, without relating them to benchmark-specific failure modes. In environments such as LunarLander-v2, CartPole-v1, and MountainCar-v0, these limitations may appear as unstable value updates, delayed convergence, reward variance across runs, sensitivity to sparse or delayed rewards, and reduced robustness under exploration noise. This benchmark-level perspective is important because it shows that prior studies often discuss baseline weaknesses in general terms, while the practical manifestations of these weaknesses may differ substantially across benchmark environments.

In summary, the recent reinforcement learning literature has achieved meaningful progress in improving data efficiency, convergence, and exploration–exploitation balance in benchmark control tasks. Nevertheless, limitations related to instability, variance across runs, and insufficient robustness under benchmark-specific dynamics remain visible in both single-policy and hybrid learning formulations. These limitations motivate the proposed AdvB-RL framework, which is positioned as a cooperative advisory alternative aimed at improving both stability and reward consistency across benchmark environments.

Contribution: Positioned against recent ensemble, hybrid, and multi-agent reinforcement learning studies, AdvB-RL introduces a dynamic advisory alternative for benchmark control tasks. This study introduces AdvB-RL, a novel RL framework that enhances decision-making by integrating a dynamic multi-advisory board mechanism. Unlike traditional reinforcement learning methods that rely on a single agent or static policies, AdvB-RL employs multiple advisory boards consisting of independent neural networks trained on diverse experiences. These advisory boards collaboratively influence the agent's real-time decision-making, optimizing the balance between exploration and exploitation while continuously refining policies based on cooperative feedback.

The primary theoretical contribution of AdvB-RL lies in its ability to adapt decision-making structures dynamically. Unlike ensemble RL methods, where model aggregation remains static, AdvB-RL continuously evaluates and updates advisory board members based on their effectiveness, ensuring robustness and adaptability in learning. Additionally, while multi-agent RL approaches incorporate multiple agents, they typically operate independently, whereas AdvB-RL leverages a cooperative advisory structure that refines shared policies dynamically. This unique mechanism enhances stability, reduces trial variance, and accelerates convergence in complex environments.

Benchmark evaluations on LunarLander-v2, CartPole-v1, and MountainCar-v0 confirm that AdvB-RL consistently outperforms methods such as DDQN, MuZero, and VBD-SQN by delivering superior reward maximization with lower variability. The framework's adaptability makes it suitable for reinforcement learning tasks with high-dimensional decision spaces. Compared to existing approaches, AdvB-RL addresses the limitations of

single-agent reinforcement learning systems and establishes a scalable and efficient solution for complex decision-making problems, setting a new benchmark in reinforcement learning. These contributions position AdvB-RL as a foundational advancement in AI-driven decision-making and control systems.

3. Experiment Environments

In addition to the primary 'LunarLander-v2' environment, two supplementary environments, CartPole-v1 and MountainCar-v0, were incorporated to ensure the generalizability of the proposed methodology. The experiments were conducted within the virtual environments provided by OpenAI Gym [2,21–23], a widely adopted suite offering diverse reinforcement learning environments. OpenAI Gym simplifies benchmarking for reinforcement learning algorithms, eliminating the need for custom environment creation.

Three discrete-action environments were specifically selected to showcase the model: CartPole-v1, LunarLander-v2, and MountainCar-v0. These benchmark tasks were chosen because they are widely used in discrete-control reinforcement-learning studies and, at the same time, represent different dynamics and reward structures. CartPole-v1 is a comparatively simpler dense-reward balance-control task; LunarLander-v2 is a crash-sensitive control task requiring coordinated stabilization and landing; and MountainCar-v0 is a momentum-driven task with sparse or delayed reward characteristics. This combination, therefore, provides a compact but diverse benchmark set for evaluating whether the proposed framework remains effective across different types of control difficulty. CartPole-v1 simulates a pole connected to a cart that moves along a track, where forces +1 for the right and -1 for the left are applied to the cart. Points are accrued as long as the pole remains upright, with episodes terminating if the pole tilts beyond 15 degrees from vertical or the cart moves more than 2.4 units from the center. LunarLander-v2 replicates a rocket landing scenario, where successful landings with minimal speed or energy reward between 100 and 140 points. Failure to land on the designated pad yields no landing reward, and episodes conclude upon a crash or successful landing, resulting in -100 or +100 points, respectively. Points are also awarded for each leg's contact with the ground, while the primary engine operation incurs a small penalty per frame. MountainCar-v0 uses momentum to navigate a car from a low point between two mountains toward a higher peak. Due to insufficient engine power for direct ascent, momentum plays a crucial role in overcoming the mountainous terrain.

4. Methodology

In reinforcement learning, the balance between exploration and exploitation is critical for effective learning and optimization. Traditional approaches often suffer from over-exploration or under-exploitation, leading to suboptimal policies. To address these issues, we propose a novel AdvB-RL framework, which leverages multiple agents and an advisory board of Artificial Neural Networks (ANNs) to optimize the policy through cooperative decision-making. This methodology section outlines the structure and function of this framework.

The AdvB-RL framework is designed to be flexible and adaptable to various reinforcement learning policies, including DDQN, DQN, and SARSA. The core idea is to train an agent using multiple instances of DDQN while optimizing the policy through interactions with an advisory board. The advisory board consists of copies of the trained network that have performed well in previous episodes. The framework of the novel methodology is presented in Figure 1 as follows (see Algorithm 1);

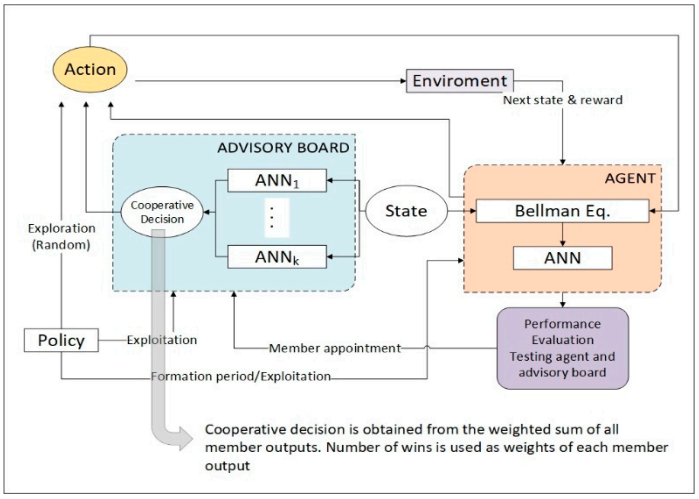

Figure 1. The framework of the novel methodology.

Exploration in RL is crucial for discovering new knowledge but can lead to inefficient learning if not managed properly. Conversely, exploitation focuses on leveraging existing knowledge to maximize rewards but risks being stuck in local minima. The AdvB-RL framework addresses these challenges using a cooperative decision-making process guided by the advisory board. During the exploration phase, the agent may take random actions to explore the environment. In the exploitation phase, the agent relies more heavily on the policy formed by the advisory board's cooperative decision. The policy is optimized based on the weighted outputs of the advisory board members, where the weights are determined by each member's performance in previous tests, which are applied after each episode. Each advisory board member i estimates a Q-value for a given state-action pair:

$$Q_i(s_i, a). \tag{1}$$

Each advisory member $i \in \{1, \ldots, K\}$ maintains its own Q-network and outputs $Q_i(s_t, a)$, representing the estimated action value of action a at state s_t. The collective decision is derived as a weighted sum of all advisory board members:

$$Q(s_t, a) = \sum_{i=1}^{K} w_i Q_i(s_t, a). \tag{2}$$

The final decision value $Q(s_t, a)$ is obtained by aggregating the action-value predictions of all K advisory members through a weighted sum, where w_i denotes the contribution weight of member i. The weights w_i are assigned based on past performance:

$$w_i = \frac{\exp(R_i)}{\sum_{j=1}^{K} \exp(R_j)}. \tag{3}$$

Here, R_i denotes the performance score of advisory members i, computed from its recent reward history. This formulation was adopted to normalize the member contributions and to assign larger weights to better-performing members while keeping the aggregation stable.

The advisory board consists of two types of members: elites and competitive members. The advisory board is central to the decision-making process in the AdvB-RL framework. It comprises multiple ANNs, each trained over different episodes. The cooperative decision is derived from the weighted sum of the outputs of these ANNs, with the number of wins or successes in previous trials determining the weight of each member's output. This

approach ensures that the decision-making process benefits from diverse perspectives while being grounded in the proven effectiveness of past strategies. The performance of both the agent and the advisory board members is periodically evaluated through a structured testing process:

Test 1: In this phase, the agent's performance is assessed by comparing its total reward with that of the elite members of the advisory board. If the agent surpasses an elite member, it replaces the weakest elite member on the advisory board.

Test 2: Evaluation of Advisory Board Members: All advisory board members are tested and graded. Next, the win value of the member who receives the highest total reward is increased by 1 point. Finally, in every discard period, the worst competitive member of the advisory board is replaced with the agent.

Test 3: Assessment of Cooperative Decision: This test monitors the advisory board's cooperative decision-making effectiveness and evaluates the board's collective output to ensure that it leads to optimal actions for the agent.

The elite member mechanism ensures that if the agent's performance is higher than that of any existing elite member, its network is included in the elite members via Test 1. On the other hand, any competitive member is eliminated if its performance is worse than that of its counterparts utilized by Test 2.

The performance score of each advisory board member is calculated as

$$R_i = \sum_{t=1}^{T} r_t^i. \tag{4}$$

The weakest elite member is identified according to its performance score as

$$i_e^* = \underset{j \in E}{\operatorname{argmin}} R_j, \tag{5}$$

where E denotes the set of elite members. If $R_{agent} > R_{i_e^*}$ the weakest elite member i_e^* is replaced by the agent model.

These dynamic replacement mechanisms ensure that the advisory board comprises the most effective networks. The evaluation process of the proposed method is presented in Figure 2. During online training, only the learner network is updated via the Bellman rule at each interaction step. The advisory board members are used as fixed reference networks while they remain on the board and contribute to action selection through weighted aggregation. However, the advisory board is dynamic at the membership level: after periodic evaluation, the current learner may replace underperforming elite or competitive members according to the appointment/replacement rules. Therefore, the framework combines a fixed-member inference stage with a dynamic board refresh mechanism.

The agent's decision-making process is grounded in the Bellman equation, which calculates the expected return of taking an action in a particular state and following the optimal policy afterward. The agent updates its policy using the modified Bellman equation:

$$Q_i(s_t,\ a_t) \leftarrow Q_i(s_t,\ a_t) + \alpha \left[r_t + \gamma \max_{a'} Q_i(s_{t+1}, a\prime) - Q_i(s_t, a_t) \right], \tag{6}$$

where α is the learning rate, γ is the discount factor for future rewards, and $Q_i(s_{t+1}, a')$ is the action-value estimate of advisory member i for the next state-action pair.

The advisory board's recommendations are fed into the Bellman Equation to determine the best possible action at each step. This integration ensures that the agent's policy is continuously optimized based on immediate and future rewards. One of the key strengths of the AdvB-RL framework is its flexibility. While this study primarily utilizes DDQN,

the framework can easily be adapted to other RL methods, such as DQN and SARSA. Furthermore, off-policy and on-policy gradient methods can be incorporated into this framework, making it versatile for various RL applications.

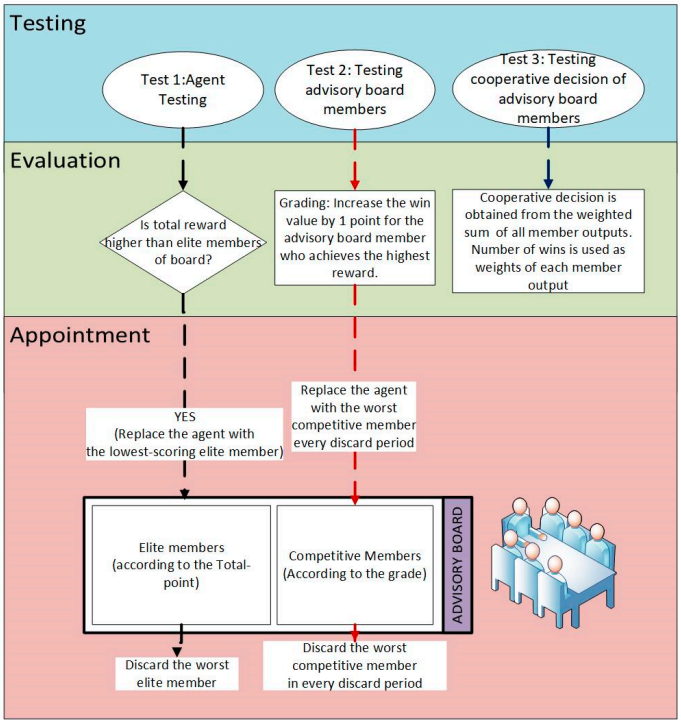

Figure 2. The performance evaluation of the advisory board.

Algorithm 1 AdvB-RL

Initialize the agent's neural network (ANN_{agent})
Initialize advisory board with k ANN_s (ANN_1, ANN_2, . . ., ANN_k)
Initialize hyperparameters
Initialize win[i] = 0 for i = 1, 2, . . ., k
FOR each episode in total episodes:
// Phase 1: Training and Decision-Making
Initialize environment, variables, and reset state
FOR each step in episode:
IF in formation period:
IF exploration phase (random actions):
action ← random action
ELSE:
Evaluate action output from ANN_{agent} given current state:
y_{agent} ← ANN_{agent}(state)
action ← argmax(y_{agent})
ENDIF
ELSE:
IF exploration phase (random actions):
action ← random action
ELSE:

Algorithm 1 AdvB-RL

FOR each i in advisory board ANN_s:
Evaluate action output from ANN_i given current state:
$y_{ANN_i} \leftarrow ANN_i(state)$
ENDFOR
Weighted sum of outputs from advisory board members:
$y_{coop} \leftarrow \sum_{i=1}^{k} win_i y_{ANN_i}$
action $\leftarrow$ argmax(y_{coop})
ENDIF
ENDIF
next state, reward $\leftarrow$ step(action, state) // Execute action in environment
Update ANN_{agent} using the Bellman update rule
ENDFOR
// Phase 2: Testing and Evaluation
// Test 1: Agent Testing
Initialize environment, variables, and reset state
$agent_{reward} \leftarrow 0$
FOR each step in episode:
$y_{agent} \leftarrow ANN_{agent}(state)$
action $\leftarrow$ argmax(y_{agent})
next state, reward $\leftarrow$ step(action, state) // Execute action in environment
$agent_{reward} \leftarrow agent_{reward}$ + reward
ENDFOR
// Test 2: Testing Advisory Board Members
// Test board members separately and grading
FOR each member in advisory board:
Initialize environment, variables, and reset state
$member_{total_reward}$[member] $\leftarrow 0$
FOR each step in episode:
$y_{ANN_{member}} \leftarrow ANN_{member}(state)$
action $\leftarrow$ argmax($y_{ANN_{member}}$)
next state, reward $\leftarrow$ step(action, state) // Execute action in environment
$member_{total_reward}$[member] $\leftarrow$ $member_{total_reward}$[member] + reward
ENDFOR
ENDFOR
best member index $\leftarrow$ argmax($member_{total_reward}$)
win[best member index] $\leftarrow$ win[best member index] + 1
// Test 3: Assessment of Cooperative Decision of Advisory Board
// Calculate cooperative decision from advisory board (weighted sum of outputs based on performance)
Initialize environment and reset state
total_reward $\leftarrow 0$
FOR each step in episode:
FOR each i in advisory board ANN_s:
$y_{ANN_{member}} \leftarrow ANN_i(state)$
ENDFOR

Algorithm 1 AdvB-RL

$y_{coop} \leftarrow \sum_{i=1}^{k} win_i y_{ANN_i}$
action $\leftarrow$ argmax(y_{coop})
next state, reward $\leftarrow$ step(action, state) // Execute action in environment
total_reward $\leftarrow$ total_reward + reward
monitor(total_reward)
ENDFOR
// Phase 3: Appointment and Member Management of Advisory Board
// If agent outperforms one of the elite members in advisory board,
// replace the lowest-scoring elite member with the agent
IF $\text{agent}_{\text{reward}}$ > min($\text{elite member}_{\text{total_reward}}$):
worst_elite_index $\leftarrow$ argmin($\text{elite member}_{\text{total_reward}}$)
$\text{ANN}_{\text{worst_elite_index}} \leftarrow \text{ANN}_{\text{agent}}$
ENDIF
// If discard period has elapsed, the worst competitive member is replaced by the current agent
IF discard period elapsed:
worst_competitive_index $\leftarrow$ argmin(win) in competitive members
$\text{ANN}_{\text{worst_competitive_index}} \leftarrow \text{ANN}_{\text{agent}}$
ENDIF
ENDFOR

5. Experiments and Results

In this study, we propose a novel framework, AdvB-RL, which we applied to the primary test environment, LunarLander-v2, as well as to classical control environments, including CartPole-v1 and MountainCar-v0, to assess both its effectiveness and its generalization capability. While utilizing DDQN as the primary algorithm, the framework is adaptable to various reinforcement learning policies. To ensure a thorough understanding of our method's behavior, we varied the number of advisory board members (1, 5, and 10) and analyzed the framework's performance. This effectively introduced a new hyper-parameter evaluated across multiple runs, with results presented in tabular format. For comparison, the standard DDQN algorithm is also tested under identical conditions. Various strategies are compared to determine the most effective advisory board strategy and to evaluate the performance difference between our framework and classic DDQN. The models were implemented using Python 3.8.18 on a system with an Intel(R) Core (TM) i7-10700F CPU @ 2.90 GHz, 32 GB of RAM, and an NVIDIA GeForce RTX 3060TI GPU with 12 GB of VRAM (Video Random Access Memory).

The code to reproduce our results is located at https://github.com/onur-osman/AdvB-RL_V1. accessed on 10 March 2026.

This section details the experiments conducted and presents the results of our analyses, focusing on the performance of different advisory board configurations and their comparison with standard DDQN approaches.

Table 2 summarizes the hyperparameters used across all strategies in our experiments, including the tuned values of the main training and network settings. These hyperparameters were determined through an extensive tuning process and then kept fixed across strategy comparisons to ensure a fair and consistent evaluation. At the same time, a limited number of benchmark-specific settings, including network architecture, formation period, and maximum episode budget, were selected according to the characteristics of each environment. Accordingly, the experiments were conducted for 1000 episodes in

LunarLander-v2 and CartPole-v1 and 1500 episodes in MountainCar-v0, with five runs per setting. These episode budgets were selected after considering related benchmark studies together with the tuning process [2,19], with the aim of providing sufficient learning time while preserving computational feasibility and comparability across experiments. A dedicated sensitivity analysis with respect to alternative hyperparameter settings and explicit noise-variant benchmark environments was outside the scope of the present study and is identified as an important direction for future work.

Table 2. Hyperparameters and values of the extensive tuning process.

Hyperparameter	Value
Maximum Episodes	1000–1500
Discount Factor (Gamma)	0.995
Learning Rate (Alpha)	1.00×10^{-3}
Discard Period	20
ANN structures	#state, 64, 64, #action (Lunar Lander) #state, 32, 32, #action (CartPole, MountainCar)
Activation Functions	ReLU, Linear
Memory Size	100,000
Mini-batch Size	64
Soft Update Parameter of Target Network	1.00×10^{-3}
Initial Epsilon	1
Minimum Epsilon	0.01
Epsilon Decay	0.995
Formation Period	[1000, 0, 1400]
Number of Advisory Board Members	[1, 5, 10]

5.1. LunarLander-v2

Table 3 compares strategies applied to the Lunar Lander problem, focusing on the final 100 episodes. This analysis specifically evaluates the performance of different configurations of AdvB in comparison to the method. The evaluation is based on the test results obtained from the last 100 episodes of each experimental run, offering a detailed assessment of the effectiveness of incorporating multiple advisory boards in enhancing the baseline method's performance.

Table 3. LunarLander-v2 performance comparison of different strategies for the last 100 test episodes.

	DDQN		**1 AdvB**	
Trial	Test Statistics	Solved (%)	Test Statistics	Solved (%)
1	203.09 ± 107.13	81	126.98 ± 173.7	62
2	221.18 ± 101.81	90	201.25 ± 100.17	66
3	233.21 ± 95.31	88	257.07 ± 58.08	94
4	258.69 ± 39.03	98	105.82 ± 200.55	50
5	223.41 ± 86.81	87	236.25 ± 93.57	87
Overall	227.92 ± 86.02	89	185.48 ± 125.22	72
	5 AdvB		**10 AdvB**	
Trial	Test Statistics	Solved (%)	Test Statistics	Solved (%)
1	268.16 ± 36.33	92	270.65 ± 20.68	98
2	232.29 ± 62.59	84	269.54 ± 26.13	97
3	266.45 ± 40.77	94	271.75 ± 23.19	99
4	268.37 ± 30.80	96	272.91 ± 20.45	99
5	252.05 ± 53.61	92	265.25 ± 33.25	95
Overall	257.46 ± 44.82	92	270.02 ± 24.74	98

The results presented in Table 3 demonstrate that increasing the number of AdvB consistently leads to improved performance, as reflected in both the overall mean scores and the percentage of problems solved. Specifically, the "5 AdvB" strategy achieves an overall mean score of 257.46 ± 44.82 and a 92% problem-solving rate, which is notably superior to the "1 AdvB" strategy's overall performance, which recorded a mean score of 185.48 ± 125.22 and a 72% problem-solving rate. The "10 AdvB" strategy exhibits the highest overall effectiveness, with an overall mean score of 270.02 ± 24.74 and a 98% problem-solving rate, indicating its robustness across trials. In comparison, the DDQN baseline, while achieving a respectable overall mean score of 227.92 ± 86.02 and an 89% problem-solving rate, falls short of the performance metrics attained by the advisory board strategies. This analysis highlights the significant advantages of employing multiple advisory boards in enhancing both the consistency and success rates of the LunarLander-v2 problem.

The findings suggest that incorporating multiple advisory boards within the RL framework significantly enhances the strategy's consistency and effectiveness, with the "10 AdvB" approach being the most promising among those tested. These trends are illustrated in Figures 3 and 4. These figures visually compare the performance of different strategies applied to the LunarLander-v2 problem, specifically focusing on average reward over episodes and the moving average with standard deviation of rewards over iterations.

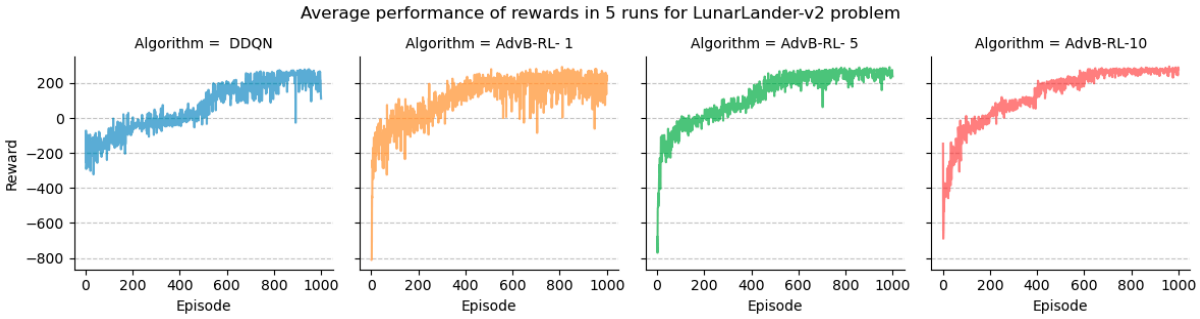

Figure 3. Comparison of Average Reward for LunarLander-v2.

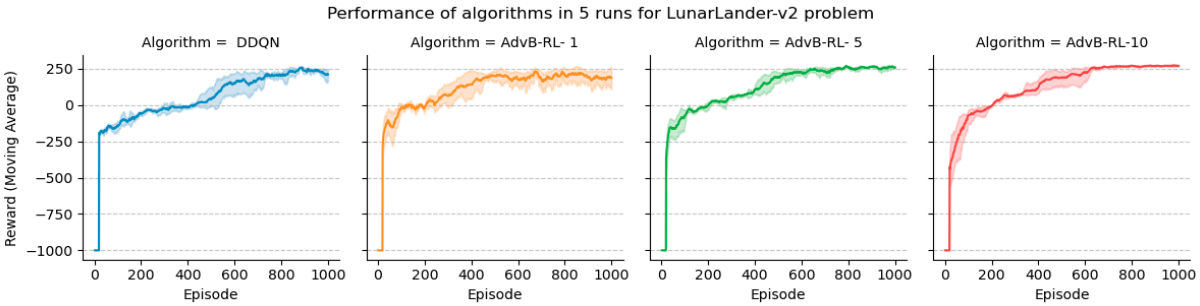

Figure 4. Moving average (rolling 20) and standard deviation for LunarLander-v2.

As seen in Figure 3, all strategies exhibit an upward trend in average reward as the number of episodes increases, indicating improved performance over time. However, the configurations with multiple advisory boards, particularly 5 AdvB and 10 AdvB, show a marked improvement in convergence speed and reward consistency. The 10 AdvB configuration achieves the highest average reward early in training, consistently maintaining superior performance throughout the episodes. This strategy not only reaches the highest overall reward but also displays the least variability, demonstrating the effectiveness of the cooperative decision-making mechanism in stabilizing learning. The 5 AdvB configuration also performs well, achieving high average rewards with slightly more variability than the 10 AdvB. The 1 AdvB strategy, while showing improvement over DDQN, exhibits slower convergence and greater fluctuations, suggesting less stable learning. In addition,

the convergence speed can be quantitatively observed in Supplementary Table S1, which shows that the moving average reward (rolling window = 20) reaches 90% of the optimal reward (180 out of 200) at approximately 650 episodes for DDQN, 450 episodes for 1 AdvB, 450 episodes for 5 AdvB, and 400 episodes for 10 AdvB.

The baseline DDQN strategy, while gradually improving, consistently underperforms relative to the advisory board-based approaches, achieving lower average rewards and displaying considerable variability throughout the episodes.

Figure 4 illustrates the moving average of rewards over 20-episode intervals across iterations, accompanied by the corresponding standard deviation for each strategy. The moving average curves, which smooth out the fluctuations in reward values, further confirm the trends observed in Figure 3. The AdvB 10 strategy maintains a consistently higher reward than the other strategies, with a relatively lower standard deviation, indicating more stable performance. The AdvB 5 strategy also shows strong performance with slightly higher variability than AdvB 10. The DDQN strategy, while following the general trend of improvement, exhibits greater variability and lower average rewards compared to the advisory board strategies. The shaded regions around the curves represent the standard deviation, providing insight into each strategy's performance variability. Overall, the figures demonstrate that strategies employing multiple advisory boards, particularly AdvB 10, are more effective and consistent in maximizing returns and minimizing variability in the LunarLander-v2 environment. Table 4 shows significant differences in performance across strategies.

Table 4. ANOVA table for LunarLander v2.

	SS	df	MS	F	*p*-Value
Between Groups	416,244.1	2	208,122	146.1142	6.56×10^{-45}
Within Groups	423,040.8	297	1424.38		
Total	839,284.8	299			

The between-group variability (SS = 416,244.1) is much higher than the within-group variability (SS = 423,040.8), reflected in a high F-statistic (146.11) and a very low *p*-value (6.56×10^{-45}). This indicates that the differences between strategies are statistically significant, with the variability between groups not due to random chance but to the strategies' effectiveness. The group mean is analyzed to determine which group makes this difference.

Figure 5 presents a boxplot analysis comparing the reward distributions of different strategies applied in the LunarLander-v2 task. The mean and the median rewards indicate that strategies with 5 and 10 advisory boards achieve superior performance for each run, reflected in higher median values than the 1 AdvB and DDQN strategies. Additionally, the interquartile range (IQR) for the 10 AdvB strategy is narrower, suggesting more consistent performance with less reward variability. As a result, 10 AdvB gives the best result for the LunarLander-v2.

Because the boxplot in Figure 5 is constructed from the last 100 test episodes of each run, the observed outliers should not be primarily attributed to intentional exploration during evaluation. Rather, they most likely correspond to occasional atypical trajectories or transient unstable policy behavior when the learned policy encounters less favorable state sequences. Their greater frequency in DDQN and 1 AdvB is consistent with the wider reward distributions and higher variability observed in the corresponding moving-average analyses, whereas the narrower interquartile range and lower variance of 10 AdvB indicate that such episodes occur less frequently.

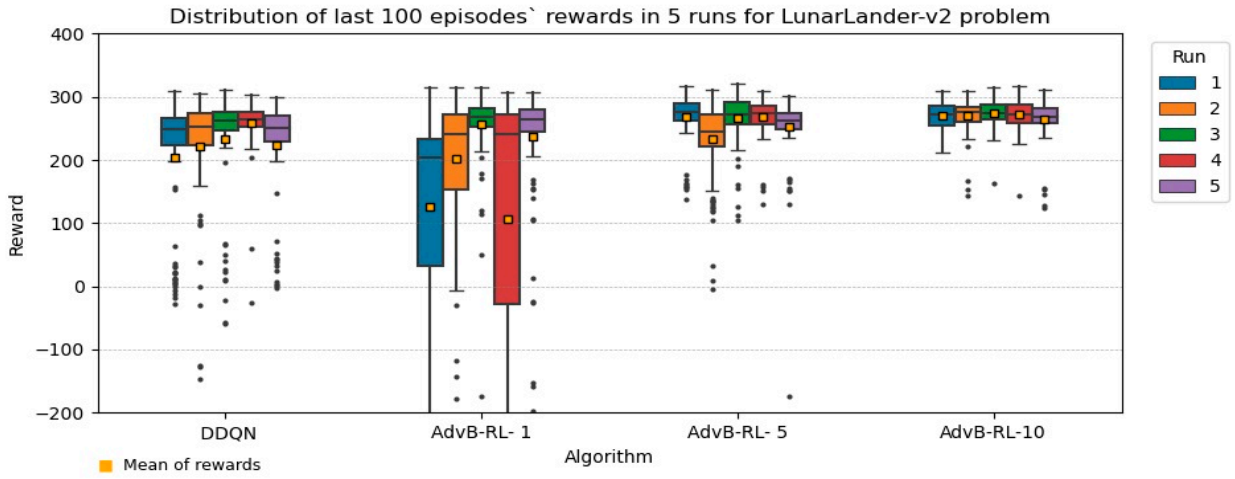

Figure 5. Boxplot Comparison of Rewards for LunarLander-v2 Strategies.

The plots reveal that all strategies show an improvement in average reward as the number of episodes increases, indicating learning and adaptation over time. However, the strategies with 5 and 10 advisory boards consistently achieve higher average rewards compared to the 1 AdvB and DDQN strategies. Notably, the 10 AdvB strategy demonstrates the highest and most consistent performance, with a smoother upward trajectory and less variability than the others. In contrast, while improving, the DDQN and 1 AdvB strategies exhibit more fluctuations and lower overall returns, suggesting less stable performance. This comparative analysis underscores the enhanced efficacy of strategies that employ multiple advisory boards, particularly the 10 AdvB approach, in achieving superior and more stable performance in the LunarLander-v2 environment. Combined with the ANOVA and box plot analysis, it is evident that the AdvB strategies, particularly AdvB 5 and AdvB 10, offer superior performance with more stability than DDQN.

A balance guides the selection of advisory board sizes between computational efficiency and the potential for improved decision-making. A single advisory board member (1 AdvB) serves as the baseline, effectively functioning as a single-agent RL system with limited policy diversity. A moderate number of 5 members allows sufficient policy diversity while maintaining manageable computational overhead. The 10-member configuration represents a larger-scale advisory system, aiming to further enhance robustness while still being computationally feasible. Preliminary experiments with advisory board sizes larger than 10 showed diminishing returns, suggesting that increasing the number of advisors beyond this point does not significantly improve performance. It is evident that a trade-off between computational efficiency and the potential for improved decision-making exists when increasing the number of advisory board members up to 10. For clarity and conciseness in the presentation, only the results for 1, 5, and 10 AdvB members are reported.

5.2. CartPole-v1

Table 5 compares strategies applied to the CartPole-v1 problem, focusing on the final 100 episodes. The evaluation is based on the test results obtained from the last 100 episodes of each experimental run, offering a detailed assessment of the effectiveness of incorporating multiple advisory boards in enhancing the baseline method's performance. Table 5 presents a performance comparison of various strategies applied to the CartPole-v1 environment across five trials, including the use of a baseline DDQN, 1, 5, and 10 AdvB models. The results demonstrate that increasing the number of advisory boards leads to improved performance and consistency. The 1 AdvB strategy yielded a moderate average score of 347.39 $\pm$ 73.41, with some variability across trials, while the 5 AdvB strategy showed significant improvement with an average score of 446.86 $\pm$ 89.45, though still

exhibiting variability. The 10 AdvB strategy achieved the highest and most consistent results, with an average score of 497.79 ± 5.18 and multiple instances where the agent reached the maximum possible score of 500.00, indicating stable and reliable performance. In contrast, the DDQN baseline method performed the worst, with an average score of 304.37 ± 144.04, and displayed the highest variability, suggesting inconsistent performance. These results highlight that increasing the number of advisory board members that are advisory significantly enhances the agent's performance and stability, with the 10 AdvB strategy consistently outperforming the lower AdvB configurations and the DDQN baseline.

Figure 6 illustrates the comparison of average rewards across episodes for each strategy. The 10 AdvB strategy demonstrates the best overall performance, achieving steady improvement after 150 episodes and stabilizing close to the maximum possible return of 500 by around 350 episodes. This strategy maintained high stability with minimal variability, as evidenced by the narrow standard deviation bands, making it the most reliable in sustaining peak performance. The 5 AdvB strategy follows closely, showing similar improvement but slightly more variability, stabilizing around 400–500 returns after 400 episodes. In contrast, the 1 AdvB strategy lags behind, taking longer to stabilize and displaying greater fluctuation, although it eventually converges to higher returns than DDQN. The DDQN baseline strategy exhibited the slowest improvement, with a more erratic learning curve and significantly lower returns than the AdvB configurations. While it gradually improves, the performance is inconsistent and fails to reach the same level of reward or stability as the AdvB strategies. Consistent with this visual trend, Supplementary Table S1 shows that the moving average reward (rolling window = 20) reaches 90% of the optimal reward (450 out of 500) at approximately 530 episodes for 10 AdvB and 680 episodes for 5 AdvB, whereas DDQN and 1 AdvB do not reach this threshold within the training horizon.

Table 5. Cartpole-V1 Performance Comparison of Different Strategies for The Last 100 Test Episodes.

Trial	DDQN	1 AdvB
1	449.18 ± 129.80	284.53 ± 65.09
2	209.83 ± 87.90	343.52 ± 93.45
3	366.51 ± 155.33	239.55 ± 53.51
4	281.86 ± 184.60	468.69 ± 53.79
5	214.49 ± 162.52	400.68 ± 101.23
Overall	304.37 ± 144.04	347.39 ± 73.41
Trial	5 AdvB	10 AdvB
1	383.37 ± 123.92	500.00 ± 0.00
2	476.98 ± 58.07	490.17 ± 19.36
3	435.00 ± 105.77	500.00 ± 0.00
4	496.43 ± 9.81	500.00 ± 0.00
5	442.55 ± 70.06	498.8 ± 6.55
Overall	446.86 ± 89.45	497.79 ± 5.18

Figure 7 presents the moving average, 20 episodes of rewards across iterations for different strategies. The lines represent the mean rewards, while the shaded areas indicate the standard deviation, showing variability in performance. AdvB 10 demonstrates the highest and most consistent reward improvement, with a relatively narrow shaded region, indicating stable performance. AdvB 5 follows closely, achieving slightly lower rewards with more variability. AdvB 1 shows slower improvement and greater variability, as indicated by a wider shaded area. Meanwhile, DDQN, although showing improvement, lags behind the advisory-based strategies and exhibits substantial variability. Overall, the

results suggest that increasing the number of advisory board members (as in AdvB 10) leads to better and more stable performance, while DDQN underperforms in comparison.

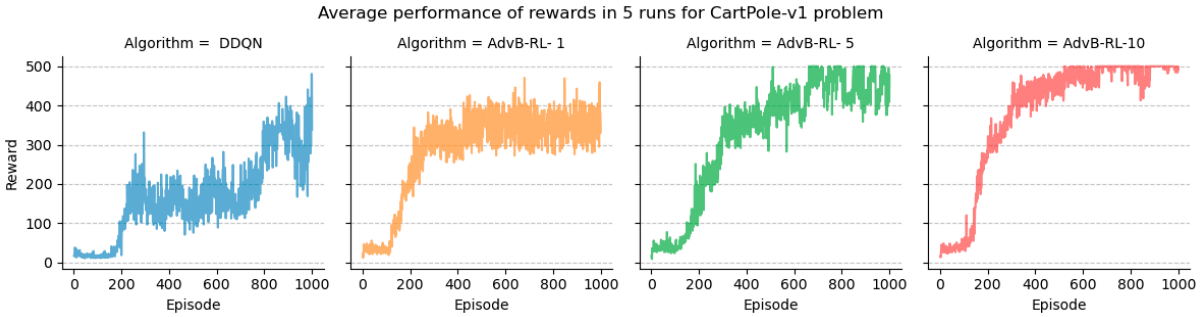

Figure 6. Comparison of Average Reward for CartPole-v1.

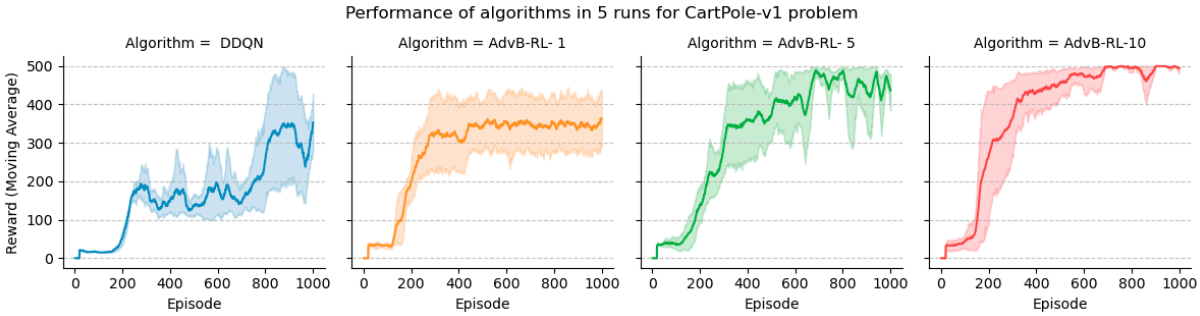

Figure 7. Moving average (rolling 20) and standard deviation for CartPole-v1.

Table 6 compares the performance between different strategies. The results show a significant difference, with the "Between Groups" sum of squares (SS) at 1,170,283 and a mean square (MS) of 585,141.7. The F-value of 657.137 indicates a substantial variation between the strategies, and the extremely small p-value (8.7×10^{-11}) confirms that these differences are statistically significant. This suggests that the performance differences between the strategies DDQN and the various AdvB configurations are meaningful and not due to random variation. The group mean is analyzed to determine which group makes this difference.

Table 6. ANOVA table for Cartpole-v1.

	SS	df	MS	F	p-Value
Between Groups	1,170,283	2	585,141.7	657.137	8.7×10^{-11}
Within Groups	264,461	297	890.4409		
Total	1,434,744	299			

Figure 8 visually represents that in the CartPole-v1 environment, the performance of the AdvB-RL framework with varying numbers of advisory board members (AdvB) demonstrates a clear trend of improvement as the number of AdvB members increases. The baseline DDQN shows a relatively wide range of rewards with a median performance between 100 and 200, indicating higher outcome variability. The AdvB-RL framework with 1 AdvB member achieves slightly more consistent performance but does not significantly improve reward. However, with 5 AdvB members, there is a notable increase in median reward and consistency, with a higher median and narrower interquartile range, suggesting more stable and improved performance. Finally, with 10 AdvB members, the framework achieves the highest median reward and consistent results. These results suggest that increasing the number of AdvB members enhances reward and stability.

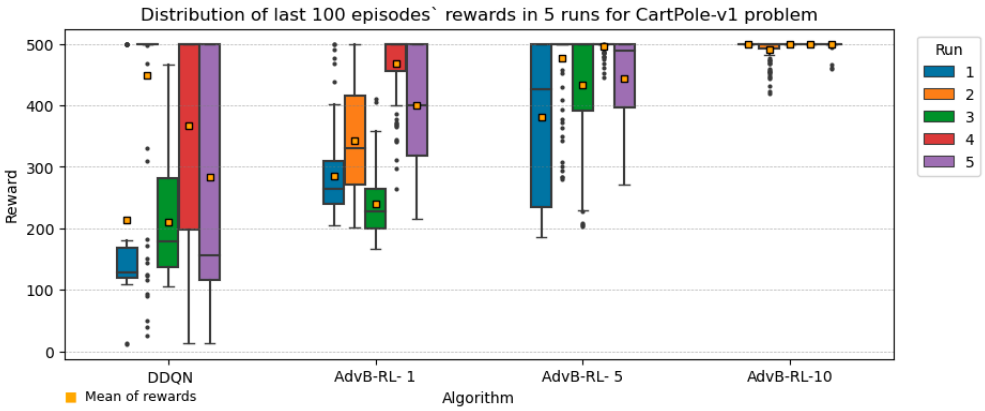

Figure 8. Boxplot Comparison of Rewards for CartPole-v1 Strategies.

5.3. MountainCar-v0

Table 7 compares strategies applied to the MountainCar-v0 problem, focusing on the outcomes from the final 100 episodes. The analysis highlights the performance of various AdvB configurations related to the baseline method. This evaluation draws on test results from the last 100 episodes of each experiment, providing a comprehensive assessment of how multiple advisory boards can improve the effectiveness of the baseline approach.

Table 7. MountainCar-v0 Performance Comparison of Different Strategies for The Last 100 Test Episodes.

Trial	DDQN	1 AdvB
1	-124.07 ± 31.95	-130.71 ± 34.76
2	-134.07 ± 31.06	-131.48 ± 28.84
3	-141.34 ± 31.60	-131.17 ± 22.98
4	-134.28 ± 32.13	-134.43 ± 45.42
5	-119.81 ± 31.47	-128.70 ± 30.70
Overall	$\mathbf{-130.71 \pm 31.64}$	$\mathbf{-131.39 \pm 32.54}$
Trial	5 AdvB	10 AdvB
1	-113.71 ± 24.46	-100.61 ± 8.75
2	-122.11 ± 26.59	-102.84 ± 18.25
3	-115.10 ± 23.59	-102.51 ± 18.75
4	-132.69 ± 29.56	-104.87 ± 12.03
5	-115.57 ± 24.32	-105.01 ± 19.57
Overall	$\mathbf{-119.83 \pm 25.70}$	$\mathbf{-103.16 \pm 15.46}$

Table 7 presents the performance evaluated across five trials, each reporting the mean reward and standard deviation. Notably, performance improvement is marked as the number of advisory boards increases from 1 to 10. For instance, the 1 AdvB configuration shows an average reward of -131.39 ± 32.54, while the 10 AdvB configuration achieves a significantly better result of -103.16 ± 15.46. The 5 AdvB configuration also shows substantial improvement, with an average reward of -119.83 ± 25.70. In contrast, the DDQN method performs similarly to the 1 AdvB configuration, with an average reward of -130.71 ± 31.64. These results indicate that incorporating multiple advisory boards enhances the baseline method's performance, with the 10 AdvB configuration yielding the best overall results across all trials.

In Figure 9, the graph compares the average reward over time for the configurations of DDQN, 1 AdvB, 5 AdvB, and 10 AdvB across 1500 episodes in the MountainCar-v0 environment. The 10 AdvB configuration consistently shows superior performance, with

the highest returns and faster convergence compared to other strategies. As the episodes progress, 10 AdvB configurations outperform the others, achieving better returns earlier. The 5 AdvB configuration follows a similar but slightly less effective trajectory, while the 1 AdvB configuration and DDQN show slower improvements and lower returns overall. The variability in performance, as indicated by fluctuations, is also more pronounced in the 1 AdvB and DDQN methods, particularly in the earlier episodes. This convergence pattern is also reflected in Supplementary Table S1, where the moving average reward (rolling window = 20) reaches the convergence threshold of −120 at approximately 1150 episodes for 10 AdvB, 1250 episodes for 5 AdvB, and 1350 episodes for 1 AdvB, while DDQN does not reach this threshold within the training horizon.

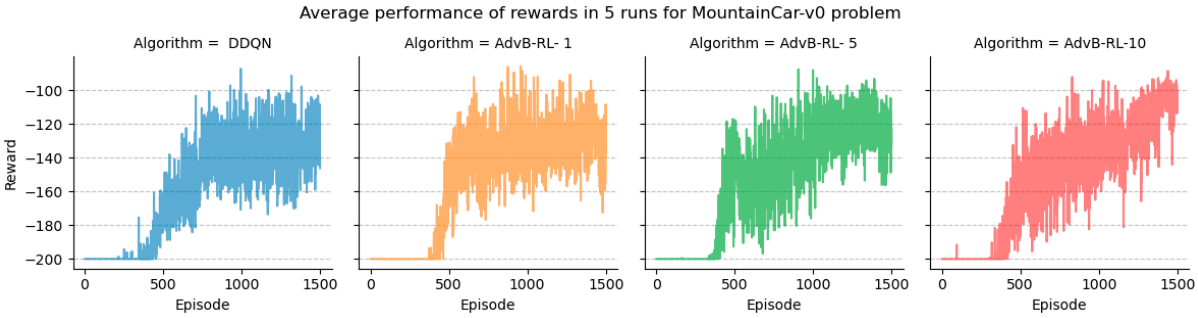

Figure 9. Comparison of Average Reward for MountainCar-v0.

Figure 10 presents the moving average and standard deviation for the same strategies, calculated over 20 episodes. This figure emphasizes the trends seen in the first graph, providing more detailed insights into the stability and consistency of each strategy. The 10 AdvB configuration demonstrates the best-moving average with lower variance over time, indicating a more stable and reliable performance. The 5 AdvB configuration also performs well but with slightly higher variance, while the 1 AdvB and DDQN methods exhibit more variability, as reflected in their wider standard deviation bands. These wider bands suggest that these strategies are less stable and take longer to converge to optimal returns.

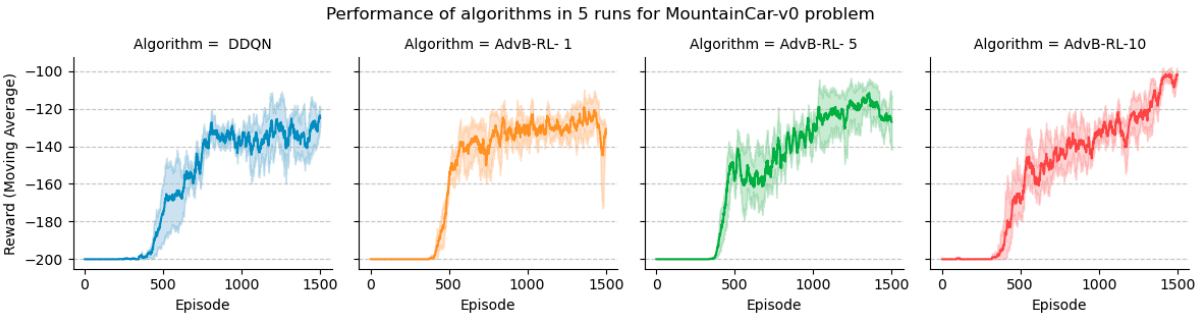

Figure 10. Moving average (rolling 20) and standard deviation for MountainCar-v0.

These two figures underscore the benefits of incorporating multiple advisory boards, with the 10 AdvB configuration consistently delivering superior performance in average reward and stability. The moving averages and standard deviations further reinforce the idea that increasing the number of advisory boards reduces variability and enhances overall performance in the MountainCar-v0 environment. The findings indicate that expanding the advisory board members, as seen with AdvB 10, results in stronger and more consistent outcomes. In contrast, DDQN lags, showing weaker and less stable performance.

The rolling window of 20 episodes was chosen as a moderate smoothing level to reduce short-term stochastic fluctuations without obscuring the overall learning dynamics.

Importantly, the same performance ordering is already visible in the corresponding unsmoothed average-reward curve (Figure 3/Figure 6/Figure 9), indicating that the observed trend is not an artifact of the smoothing choice.

Table 8 results and Figure 11 together provide a comprehensive analysis of the performance of different strategies for the MountainCar-v0 problem. The ANOVA table reveals significant differences between the strategies, with an F-value of 155.46 and an extremely small p-value (6.37×10^{-47}), indicating that the group variations are statistically significant. This means that the differences in performance across the methods DDQN, AdvB 10, AdvB 5, and AdvB 1 are not random and have a meaningful impact.

Table 8. ANOVA table for MountainCar-v0.

	SS	df	MS	F	p-Value
Between Groups	41,549.35	2	20,774.68	155.4566	6.37×10^{-47}
Within Groups	39,690.05	297	133.6365		
Total	81,239.41	299			

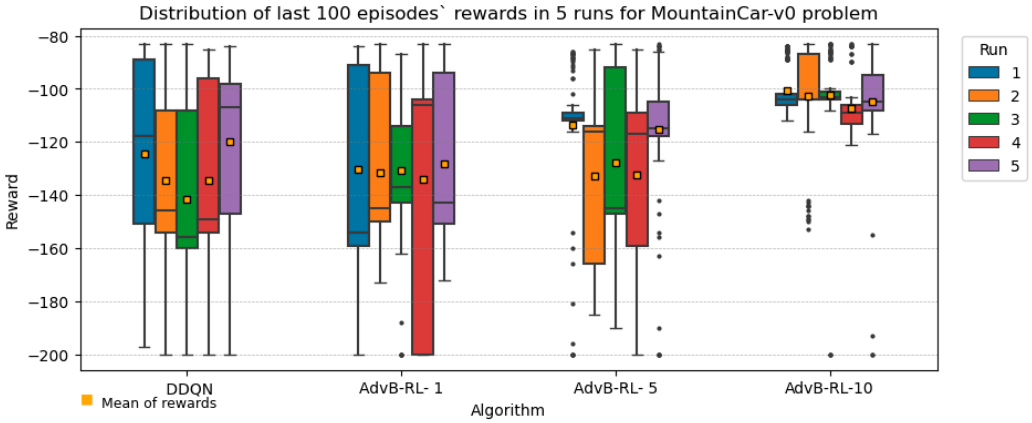

Figure 11. Boxplot Comparison of Rewards for MountainCar-v0 Strategies.

Figure 11 further illustrates this by showing that the AdvB 10 configuration has the highest median reward, with a tight interquartile range (IQR), suggesting more consistent performance. In contrast, DDQN and AdvB 1 display lower medians and wider distributions. Together, these results demonstrate that the advisory board strategies, particularly AdvB 10, outperform DDQN regarding higher rewards and more stable outcomes.

To broaden the empirical evaluation, additional value-based baselines for discrete-action control, namely C51, Dueling DDQN, Prioritized DDQN, and Rainbow DDQN, were also considered. Across the three benchmark environments, these additional comparisons further supported the robustness of the proposed method. In CartPole-v1 (Supplementary Figure S1), the additional baselines exhibited unstable learning dynamics and high variability, with average rewards remaining at approximately 300 for C51, 280 for Dueling DDQN, 270 for Prioritized DDQN, and 330 for Rainbow DDQN. In MountainCar-v0 (Supplementary Figure S2), the average rewards of the additional baselines were observed at approximately -115 for C51 and around -112 for Dueling DDQN, Prioritized DDQN, and Rainbow DDQN, indicating that the proposed method achieved clearly superior performance in this environment. In LunarLander-v2 (Supplementary Figure S3), the average rewards were observed to be approximately 250 for C51, 265 for Dueling DDQN, around 150 for Prioritized DDQN, and 235 for Rainbow DDQN, showing that the proposed method remained highly competitive against the strongest additional baseline while clearly outperforming the weaker variants. These findings indicate that AdvB-10 maintains strong and

consistent performance not only against DDQN, VBDSQN, and MuZero, but also against additional widely used DQN-family baselines.

The primary objective of this study was to apply and explore the Lunar Lander problem as the foundational use case. However, to further validate the robustness and generalization capability of the proposed method, we also incorporated two well-known reinforcement learning benchmarks: Mountain Car and Cart Pole. These problems are chosen because they present unique dynamics and challenges, allowing us to thoroughly test the adaptability and performance of our approach across different environments. Despite these additional experiments, the core focus of our research remains centered on the Lunar Lander problem, which serves as the main context for evaluating the proposed methodology. By extending the application to Mountain Car and Cart Pole, a more comprehensive assessment is aimed to be provided regarding how well the method generalizes to diverse problems in reinforcement learning.

The experimental results demonstrate the clear advantages of AdvB-RL over conventional reinforcement learning methods. The most notable strength of AdvB-RL is its stability, as evidenced by the significantly lower variance in reward distributions across all tested environments. For instance, in the LunarLander-v2 task, AdvB-RL (10 AdvB) maintained a score, whereas MuZero exhibited a much higher variance, highlighting the robustness of the proposed method. Additionally, the improved balance between exploration and exploitation is evident in the MountainCar-v0 results, where AdvB-RL outperformed baseline DDQN approaches (-103.16 ± 15.46 vs. -130.71 ± 31.64). These results collectively indicate that AdvB-RL not only enhances reward maximization but also ensures stable decision-making in reinforcement learning environments.

6. Discussion

This section summarizes the comparative analysis of AdvB-RL with 10 advisory board members (AdvB10-RL) with the methods presented in recent studies [23,24,26,27,29,35,36], using CartPole, Lunar Lander, and Mountain Car as benchmark environments presented in Table 9. Key metrics, such as average reward, variance, and convergence speed, highlight the robustness and adaptability of the AdvB10-RL in diverse tasks.

Table 9. Comparative analysis table for the methods of reinforcement learning.

Articles	Technique	Lunar Lander	Cart Pole	Mountain Car
Proposed Method	10 AdvB	270.02 ± 24.74	497.79 ± 5.18	-103.16 ± 15.46
[26]	VBDSQN	248.2 ± 21.0	NA	-114.3 ± 9.9
	DSN (Deep Sarsa)	188.9 ± 25.7	NA	-149.1 ± 18.7
[35]	MuZerolg + lc (Pretraining)	335.72 ± 162.49	104.99 ± 116.67	NA
	MuZerolg + lc	100.46 ± 123.82	410.59 ± 130.71	NA
[24]	Actor Critic bp ep	147.8 ± 10.7	193 ± 5.3	NA
[36]	DDPP	250 ± 21	NA	NA
[27]	FPER	NA	$400\pm$ (std not mentioned)	NA
[29]	D2D-SPL A3C	250 ± 30 190 ± 90	NA NA	NA NA
[23]	DQN-Augment	200 ± 215	410 ± 495	NA

NA: Not Available

In the CartPole environment, the AdvB10-RL achieved a higher average reward of 497.79 ± 5.18, compared to DQN-Augment's 410 ± 495, which is proposed in the study [23], demonstrating much lower variance. Despite DQN-Augment requiring slightly fewer episodes to solve the environment (184 eps.), its higher variance suggests instability in

performance. Similarly, in the Lunar Lander environment, AdvB10-RL outperformed DQN-Augment with an average reward of 270.02 ± 24.7 against 200 ± 215, maintaining more stable results. In the challenging Mountain Car environment, AdvB10-RL again surpassed DQN-Augment with -103.16 ± 15.46 compared to -120 ± -109, proving better control over exploration and exploitation.

In the Lunar Lander environment, while MuZero with latent game representation and latent consistency, (MuZerolg + lc) (Pretraining) proposed by [35] achieved a higher reward of 335.72 ± 162.49, its increased variance indicated less stability compared to AdvB10-RL's 270.02 ± 24.7. In CartPole, AdvB10-RL provided more consistent results, with 497.79 ± 5.18 compared to MuZerolg + lc's 410.59 ± 130.71. This highlights AdvB10-RL's superior stability, even though MuZero's pre-training phase allowed faster convergence. When compared to the actor-critic backpropagation with Equilibrium Propagation (BP-EP) method proposed in [24], for CartPole, AdvB10-RL delivered a significantly higher average reward of 497.79 ± 5.18 versus 193 ± 5.3 after 1000 episodes, demonstrating faster convergence and lower variance. Similarly, in Lunar Lander, AdvB10-RL exhibited better performance with 270.02 ± 24.7, compared to BP-EP's 147.8 ± 10.7, proving to be more efficient in learning speed and reward stability.

In the CartPole environment, Value Difference-Based Deep Sarsa and Q Networks (VBDSQN) proposed by [26] outperformed AdvB10-RL with an average reward of 992.3 ± 21.2. This result was obtained without truncating episodes at a maximum reward of 500, unlike our AdvB10-RL method, which adhered to the standard truncation setup and achieved a consistent reward of 497.79 ± 5.18. The inflated reward in VBDSQN does not necessarily reflect superior performance, as it continued accumulating rewards beyond the threshold where AdvB10-RL would have terminated. Additionally, the higher variance observed in VBDSQN suggests less stability compared to the more consistent performance of AdvB10-RL. Similarly, in Lunar Lander, AdvB10-RL showed superior performance, achieving 270.02 ± 24.7, while VBDSQN achieved 248.2 ± 21.0, by considering both stability (robustness) in the whole training process tests and success performance (98%). AdvB10-RL once again outperformed both VBDSQN and Deep Sarsa Networks (DSN) in the Mountain Car environment with -103.16 ± 15.46, underscoring its reliability in complex and unstable environments.

In Lunar Lander, the AdvB10-RL delivered slightly better performance than the Descending Dynamic Policy Programming (DDPP) proposed in [36], achieving 270.02 ± 24.7 compared to DDPP's 250 ± 21. Although DDPP demonstrated faster convergence (600 episodes), AdvB10-RL maintained higher rewards and more consistent performance over extended trials, proving more effective in reward maximization over time. In CartPole, AdvB10-RL significantly outperformed the Fresher Experience Plays a More Important Role in Prioritized Experience Replay (FPER) method proposed by [27], with an average reward of 497.79 ± 5.18 compared to FPER's 400 (standard deviation is not mentioned).

In the Lunar Lander environment, AdvB10-RL outperformed both the Discrete-to-Deep Supervised Policy Learning (D2D-SPL) and Asynchronous Advantage Actor-Critic (A3C) methods proposed in [29], achieving a reward of 270.02 ± 24.74, compared to D2D-SPL's 250 ± 30 and A3C's 190 ± 60. Moreover, AdvB10-RL demonstrated much faster convergence and more consistent results, reinforcing its superiority in complex environments requiring stability and reward maximization. Across all comparisons, the AdvB10-RL consistently exhibited superior performance, particularly regarding stability and reward maximization. Its ability to maintain lower variance while achieving higher average rewards across various environments, such as CartPole, Lunar Lander, and Mountain Car, underscores its robustness and adaptability. While larger advisory boards improved reward stability and overall performance, they also introduced additional computational

overhead. Because each advisory member requires an extra forward evaluation during action selection, the inference cost grows approximately linearly with board size. Thus, the benefits of larger advisory boards should be interpreted together with their increased runtime and memory requirements. Although the tested configurations remained feasible in our experimental setting, future work should examine this trade-off more systematically through detailed runtime and scalability analysis.

Increasing the number of advisory board members introduces additional forward evaluations during decision-making, leading to an approximately linear increase in computation time, which can be expressed as $O(M \cdot C)$, where M is the number of advisory members and C is the cost of a single advisory-member evaluation. Therefore, the observed performance improvements should be interpreted together with the corresponding runtime overhead. This reflects a trade-off between computational cost and improved decision quality through advisory diversity, especially since the marginal gains may taper as the number of members increases.

While other methods like VBDSQN and MuZero achieved higher peak rewards in some cases, their increased variability and instability make AdvB10-RL the more reliable and effective solution for reinforcement learning tasks requiring long-term consistency and performance. The hyperparameters and structural settings of AdvB-RL were determined through preliminary experiments conducted during the model-development stage. These experiments included testing different elite/competitive member compositions, advisory board update cycles, and cooperative action-selection policies. Several elite-member counts within the 10-member board (e.g., 3, 5, and 7 elite members), update/discard cycles up to 50 episodes, and alternative decision mechanisms such as best-member selection, majority voting, and weighted output aggregation were evaluated. The final configuration used in the main experiments was selected according to overall stability, convergence behavior, and reward performance, while representative examples of these sensitivity analyses are provided in the Supplementary Material (Figures S4 and S5). The empirical comparison was broadened by including additional value-based baselines for discrete-action control, as reported in Supplementary Figures S1–S3. Accordingly, the stability and robustness claims of the present study should be interpreted within the scope of the benchmark environments and training horizons evaluated here. Longer-horizon evaluations and real-world transfer analyses were not included in the present study. This study proposes using a weighting method for the collective decision-making of Advisory Board members. However, developing alternative methods tailored to different problem types is expected to be essential in future work.

7. Conclusions

This study introduced the AdvB-RL framework as a reinforcement learning approach that incorporates advisory board-based cooperative decision-making to improve learning stability and performance. Across the benchmark environments considered in this study, AdvB-RL demonstrated strong results in terms of average reward, convergence behaviour, and stability, outperforming or remaining competitive with several comparison methods, including DDQN, MuZero, and VBDSQN.

The results suggest that the use of multiple advisory members can improve the robustness of decision-making, particularly in environments where stable learning is critical. At the same time, the present findings should be interpreted within the scope of the benchmark environments and training horizons evaluated in this work. In particular, increasing the number of advisory board members introduces additional computational overhead, creating a trade-off between improved decision quality and increased runtime. Moreover, the current study did not include dedicated experiments on longer training

horizons, real-world transfer scenarios, or a comprehensive component-wise ablation of the weighting formulation.

Future work should therefore investigate the scalability of the advisory board mechanism in greater detail, including runtime and memory analyses for different board sizes, broader long-horizon evaluations, and transfer to more realistic or domain-specific reinforcement learning tasks. In addition, further work on alternative cooperative decision mechanisms and more extensive ablation studies may help clarify which components of the advisory board structure contribute most strongly to performance and stability.

Supplementary Materials: The supplementary material for this article can be found online at the GitHub link below or https://www.mdpi.com/article/10.3390/a19030230/s1. With the help of the codes in GitHub, the classification results can be obtained by applying the dataset to the desired method. https://github.com/onur-osman/AdvB-RL_V1/blob/main/algorithms-4187044-supplementary.pdf, accessed on 10 March 2026.

Author Contributions: All authors contributed to the study conception and design. All authors performed material preparation and analysis. All authors wrote the first draft of the manuscript. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The datasets generated during and/or analyzed during the current study are available from the corresponding author on reasonable request.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Furfaro, R.; Bloise, I.; Orlandelli, M.; Di Lizia, P.; Topputo, F.; Linares, R. Deep Learning for Autonomous Lunar Landing. *Adv. Astronaut. Sci.* **2018**, *167*, 3285–3306.
2. Shen, D. Comparison of Three Deep Reinforcement Learning Algorithms for Solving the Lunar Lander Problem. In *Proceedings of the 2023 International Conference on Data Science, Advanced Algorithm and Intelligent Computing (DAI 2023)*; Ahmad, B.H., Ed.; Advances in Intelligent Systems Research; Atlantis Press International BV: Dordrecht, The Netherlands, 2024; Volume 180, pp. 187–199.
3. Moerland, T.M.; Broekens, J.; Plaat, A.; Jonker, C.M. Model-Based Reinforcement Learning: A Survey. *FNT Mach. Learn.* **2023**, *16*, 1–118. [CrossRef]
4. Nugroho, L.; Zani, N.R.; Qomariyah, N.; Akmeliawati, R.; Andiarti, R.; Wijaya, S.K. Powered Landing Guidance Algorithms Using Reinforcement Learning Methods for Lunar Lander Case. *Indones. J. Aerosp.* **2021**, *19*, 43–56.
5. Shah, S.; Yao, N. Deep Reinforcement Learning for Unpredictability-Induced Rewards to Handle Spacecraft Landing. In *Proceedings of the 2023 13th International Conference on Information Science and Technology (ICIST)*; IEEE: Cairo, Egypt, 2023; pp. 73–79.
6. Zhang, N.; Gupta, A.; Chen, Z.; Ong, Y.-S. Multitask Neuroevolution for Reinforcement Learning with Long and Short Episodes. *IEEE Trans. Cogn. Dev. Syst.* **2023**, *15*, 1474–1486. [CrossRef]
7. Latyshev, K.; Garzaniti, N.; Crawley, E.; Golkar, A. Lunar Human Landing System Architecture Tradespace Modeling. *Acta Astronaut.* **2021**, *181*, 352–361. [CrossRef]
8. Lu, Y.; Squillante, M.S.; Wu, C.W. A Control-Model-Based Approach for Reinforcement Learning. *arXiv* **2019**, arXiv:2206.03567.
9. Gou, S.Z.; Liu, Y. DQN with Model-Based Exploration: Efficient Learning on Environments with Sparse Rewards. *arXiv* **2019**, arXiv:1903.09295. [CrossRef]
10. Guttulsrud, H.; Sandnes, M.; Shrestha, R. Solving the Lunar Lander Problem with Multiple Uncertainties Using a Deep Q-Learning Based Short-Term Memory Agent. In *Proceedings of the 2023 12th International Conference on Computing and Pattern Recognition*; ACM: Qingdao, China, 2023; pp. 27–33.
11. Meng, W.; Zheng, Q.; Shi, Y.; Pan, G. An Off-Policy Trust Region Policy Optimization Method with Monotonic Improvement Guarantee for Deep Reinforcement Learning. *IEEE Trans. Neural Netw. Learn. Syst.* **2022**, *33*, 2223–2235. [CrossRef]
12. Kang, C.; Rong, C.; Ren, W.; Huo, F.; Liu, P. Deep Deterministic Policy Gradient Based on Double Network Prioritized Experience Replay. *IEEE Access* **2021**, *9*, 60296–60308. [CrossRef]

13. Dollen, D.V. Investigating Reinforcement Learning Agents For. *arXiv* **2017**, arXiv:1708.02378.
14. Van Hasselt, H. Double Q-Learning. In Proceedings of the 24th International Conference on Neural Information Processing Systems, Vancouver, BC, Canada, 1 January 2010; pp. 2613–2621.
15. Van Hasselt, H.; Guez, A.; Silver, D. Deep Reinforcement Learning with Double Q-Learning. In Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, Phoenix, AZ, USA, 12–17 February 2016; Volume 30. [CrossRef]
16. Brockman, G.; Cheung, V.; Pettersson, L.; Schneider, J.; Schulman, J.; Tang, J.; Zaremba, W. OpenAI Gym. *arXiv* **2016**, arXiv:1606.01540. [CrossRef]
17. Cini, A.; D'Eramo, C.; Peters, J.; Alippi, C. Deep Reinforcement Learning with Weighted Q-Learning. *arXiv* **2022**, arXiv:2003.09280. [CrossRef]
18. Ermacora, G.; Rosa, S.; Toma, A. Fly4SmartCity: A Cloud Robotics Service for Smart City Applications. *J. Assoc. Inf. Syst.* **2016**, *8*, 347–358. [CrossRef]
19. Mnih, V.; Kavukcuoglu, K.; Silver, D.; Graves, A.; Antonoglou, I.; Wierstra, D.; Riedmiller, M. Playing Atari with Deep Reinforcement Learning. *arXiv* **2013**, arXiv:1312.5602. [CrossRef]
20. Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A.A.; Veness, J.; Bellemare, M.G.; Graves, A.; Riedmiller, M.; Fidjeland, A.K.; Ostrovski, G.; et al. Human-Level Control through Deep Reinforcement Learning. *Nature* **2015**, *518*, 529–533. [CrossRef] [PubMed]
21. Yu, X. Deep Q-Learning on Lunar Lander Game. 2019. Available online: https://www.researchgate.net/publication/333145451_Deep_Q-Learning_on_Lunar_Lander_Game (accessed on 10 March 2026).
22. Gadgil, S.; Xin, Y.; Xu, C. Solving The Lunar Lander Problem under Uncertainty Using Reinforcement Learning. In *Proceedings of the 2020 SoutheastCon*; IEEE: Raleigh, NC, USA, 2020; pp. 1–8.
23. Rusdyputra, W.; Chowanda, A. Data-Efficient Reinforcement Learning with Data Augmented Episodic Memory. *Procedia Comput. Sci.* **2023**, *227*, 1120–1126. [CrossRef]
24. Kubo, Y.; Chalmers, E.; Luczak, A. Combining Backpropagation with Equilibrium Propagation to Improve an Actor-Critic Reinforcement Learning Framework. *Front. Comput. Neurosci.* **2022**, *16*, 980613. [CrossRef]
25. Hafiz, A.M.; Bhat, G.M. Deep Q-Network Based Multi-Agent Reinforcement Learning with Binary Action Agents. *arXiv* **2020**, arXiv:2008.04109.
26. Xu, Z.; Cao, L.; Chen, X. Deep Reinforcement Learning with Adaptive Update Target Combination. *Comput. J.* **2020**, *63*, 995–1003. [CrossRef]
27. Ma, J.; Ning, D.; Zhang, C.; Liu, S. Fresher Experience Plays a More Important Role in Prioritized Experience Replay. *Appl. Sci.* **2022**, *12*, 12489. [CrossRef]
28. Zheng, J.; Kurt, M.N.; Wang, X. Stochastic Integrated Actor–Critic for Deep Reinforcement Learning. *IEEE Trans. Neural Netw. Learn. Syst.* **2024**, *35*, 6654–6666. [CrossRef] [PubMed]
29. Kurniawan, B.; Vamplew, P.; Papasimeon, M.; Dazeley, R.; Foale, C. Discrete-to-Deep Reinforcement Learning Methods. *Neural Comput. Appl.* **2022**, *34*, 1713–1733. [CrossRef]
30. Lowe, R.; Wu, Y.; Tamar, A.; Harb, J.; Abbeel, P.; Mordatch, I. Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments. In *Advances in Neural Information Processing Systems 30*; Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R., Eds.; Curran Associates, Inc.: Red Hook, NY, USA, 2017; pp. 6379–6390.
31. Zhang, Y.; Zhou, Y. Cooperative Multi-Agent Actor-Critic Approach Using Adaptive Value Decomposition and Parallel Training for Traffic Network Flow Control. *Neurocomputing* **2025**, *623*, 129384. [CrossRef]
32. Song, X.B.; Zhou, B. Cooperative Traffic Signal Control Through a Counterfactual Multi-Agent Actor–Critic with Scheduler. *Transp. Res. Part C Emerg. Technol.* **2024**, *160*, 104528. [CrossRef]
33. Gao, X.; Hu, X.; Liu, J.; Li, T. Event-Driven Prescribed Optimal Disturbance Rejection for Dynamic Positioning of Ships via Reinforcement Learning. *IEEE Trans. Autom. Sci. Eng.* **2026**, *23*, 4615–4626. [CrossRef]
34. Cao, L.; Qin, Y.; Pan, Y.; Liang, H. Prescribed Performance-Based Optimal Formation Control for USVs with Position Constraints and Yaw Angle Time-Varying Partial Constraints. *IEEE Trans. Intell. Transp. Syst.* **2024**, *26*, 4109–4121. [CrossRef]
35. Scholz, J.; Weber, C.; Hafez, M.B.; Wermter, S. Improving Model-Based Reinforcement Learning with Internal State Representations through Self-Supervision. In *Proceedings of the 2021 International Joint Conference on Neural Networks (IJCNN)*; IEEE: Shenzhen, China, 2021; pp. 1–8.
36. Li, R.; Shang, Z.; Zheng, C.; Li, H.; Liang, Q.; Cui, Y. Dynamic Policy Programming with Descending Regularization for Efficient Reinforcement Learning Control. In *Proceedings of the 2022 5th International Conference on Pattern Recognition and Artificial Intelligence (PRAI)*; IEEE: Chengdu, China, 2022; pp. 6–11.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.