EUROPEAN JOURNAL OF EMERGING ARTIFICIAL INTELLIGENCE

An Empirical Framework for Evaluating Reinforcement Learning in Automated Optimization Systems

Dr. Linnea J. Arwood
Department of Computer Engineering North Cascadia Institute of Technology, Seattle, USA

V0LUME02 ISSUE01 (2025)
Published Date: 19 March 2025 // Page no.: - 21-33

ABSTRACT

The integration of Reinforcement Learning (RL) into automation represents a paradigm shift in solving complex optimization problems across various industries[6]. While RL has demonstrated significant potential, its practical application is often hampered by a lack of standardized evaluation frameworks, making it difficult for practitioners to select appropriate algorithms for specific tasks[7]. This study introduces and executes a comprehensive empirical investigation to systematically evaluate the performance of leading RL algorithms across a diverse set of simulated automation environments[8]. We designed three high-fidelity simulation suites mimicking critical optimization tasks in manufacturing (production scheduling, inventory management), energy systems (microgrid management, HVAC control), and robotics (motion planning, multi-robot coordination)[9]. Within these environments, we benchmarked a portfolio of algorithms, including Deep Q-Networks (DQN), Proximal Policy Optimization (PPO), Soft Actor-Critic (SAC), and Multi-Agent Deep Deterministic Policy Gradient (MADDPG), against key performance indicators: task efficiency, sample complexity, scalability, and robustness to environmental stochasticity[10]. Our results reveal a nuanced performance landscape where no single algorithm dominates across all domains[11]. For instance, while PPO demonstrated superior stability and performance in continuous control tasks prevalent in robotics and HVAC systems, DQN-based variants excelled in discrete action spaces typical of scheduling and inventory problems[12]. Multi-agent algorithms showed profound efficiency gains in cooperative tasks but suffered from higher training complexity[13]. The findings underscore a critical trade-off between algorithm complexity, sample efficiency, and task-specific performance[14]. This research provides a foundational empirical baseline, offering actionable insights for deploying RL in real-world automation and highlighting critical areas for future research, particularly in enhancing transfer learning, safety, and interpretability to bridge the persistent gap between simulation and practical deployment[15].

Keywords: Reinforcement Learning, Automation, Optimization, Robotics, Energy Systems, Manufacturing, Algorithmic Benchmarking

1. Introduction

1.1 Broad Background and Historical Context

The pursuit of optimization is a cornerstone of industrial and technological progress[17]. From the earliest days of the industrial revolution, the goal has been to maximize output, minimize waste, and enhance efficiency[18]. Traditionally, this pursuit relied on classical optimization techniques rooted in mathematical programming, operations research, and heuristics[19]. Methods such as linear programming, genetic algorithms, and simulated annealing have been instrumental in solving well-defined problems with known constraints and objectives[20]. However, the increasing complexity and dynamism of modern systems—characterized by high dimensionality, non-linear dynamics, and pervasive uncertainty—have begun to expose the limitations of these conventional approaches[21]. They often struggle with scalability, require precise system models, and lack the adaptability needed to respond to real-time changes in the operating environment[22]. The advent of machine learning, and specifically Reinforcement Learning (RL), has offered a transformative alternative[23]. RL provides a mathematical framework for learning optimal behavior through direct interaction with an environment, without requiring an explicit model of its dynamics[24]. An RL agent learns a "policy"—a mapping from states to actions—by iteratively performing actions and observing the resulting rewards or penalties, with the objective of maximizing a cumulative reward signal[25]. This trial-and-error learning paradigm is uniquely suited for sequential decision-making problems under uncertainty, which are ubiquitous in automation[26]. The field of RL witnessed a monumental leap forward with the integration of deep neural networks, giving rise to Deep Reinforcement Learning (DRL)[27]. The ability of deep learning to approximate complex, high-dimensional functions enabled RL agents to learn directly from raw, high-dimensional inputs, such as images or sensor data[28]. The landmark success of a DRL agent achieving superhuman performance in Atari games directly from pixel inputs [29]catalyzed a wave of research and application, demonstrating that DRL could tackle problems previously considered intractable[30]. This breakthrough signaled the potential to move beyond games and address complex, real-world optimization challenges in critical sectors like manufacturing, energy, and robotics[31].

1.2 Critical Literature Review

The application of RL and DRL to optimization in automation has grown into a vibrant and rapidly expanding field of research[32]. Foundational reviews have established the core principles and surveyed the broad potential of these techniques in industrial settings[33]. The literature provides compelling evidence of RL's efficacy across three major domains: smart manufacturing, sustainable energy systems, and intelligent robotics[34].

In **smart manufacturing**, RL has emerged as a powerful tool for enhancing operational efficiency and adaptability[35]. A significant body of work focuses on production scheduling, where the goal is to allocate tasks to resources over time to optimize metrics like throughput and cost[36]. Researchers have shown that DRL approaches can outperform traditional mixed-integer linear programming and heuristic methods, especially in handling the complexities and uncertainties inherent in dynamic job-shop environments[37]. Distributional RL, which learns the full distribution of returns rather than just the expected value, has been shown to be particularly effective for managing risk in chemical production processes[38]. Another critical area is inventory management, where DRL offers a roadmap for controlling stochastic demand and complex supply chains[39]. Both single-agent [40]and cooperative multi-agent RL (MARL) frameworks [41]have been developed to minimize costs and product wastage while ensuring product availability[42]. Furthermore, RL provides dynamic solutions for maintenance planning, using real-time system data to schedule maintenance activities, thereby extending asset life and minimizing downtime[43]. In process control, RL's model-free nature allows it to manage complex, non-linear manufacturing processes, with recent efforts focusing on enhancing interpretability [44]and integrating domain expertise through apprenticeship learning[45]. The

energy sector is undergoing a profound transformation driven by the need for sustainability and grid stability, and RL is playing a pivotal role in this transition[46]. One key application is demand response, where DRL and MARL are used to dynamically adjust energy consumption in buildings and industrial facilities in response to grid signals, leading to significant energy savings and improved grid management[47]. Meta-learning has been explored to bridge the gap between simulated and real-world demand response systems[48]. In microgrid management, DRL algorithms optimize the distribution and usage of diverse energy resources (e.g., solar, wind, battery storage), enhancing grid resilience and cost efficiency[49]. The integration of variable renewable energy sources into the main power grid presents a major challenge due to their intermittency[50]; RL's ability to learn adaptive control policies is crucial for ensuring grid stability while maximizing the use of clean energy[51]. Heating, Ventilation, and Air Conditioning (HVAC) systems are major energy consumers, and numerous studies have demonstrated the effectiveness of DRL in optimizing their operation to reduce energy consumption without compromising occupant comfort[52]. Recent work has also focused on ensuring the safety of such control systems using batch RL techniques[53]. In the field of

robotics, RL has been instrumental in enabling robots to acquire complex skills and operate in unstructured environments[54]. Motion planning, a fundamental challenge in robotics, has been significantly advanced by DRL, which allows robots to navigate dynamic environments and execute tasks with greater adaptability[55]. Techniques like curriculum learning [56]and user-guided learning [57]have been employed to train robotic arms, while prior policy guidance has been used for complex tasks like controlling dual-arm free-floating space robots[58]. Grasping and manipulation are other areas where DRL has made remarkable progress, enabling robots to handle a wide variety of objects by learning directly from visuo-motor feedback[59]. Beyond single-robot tasks, multi-robot coordination leverages MARL to develop sophisticated collaborative strategies for tasks like pick-and-place in smart manufacturing settings[60]. A growing and critical area is human-robot collaboration (HRC), where DRL helps create more intuitive and effective interactions[61]. Research in HRC focuses on enabling robots to adapt to human behaviors and preferences while ensuring safety [62], with a strong emphasis on developing explainable RL models to enhance trust and interaction quality[63].

1.3 Research Gap

Despite the proliferation of successful applications documented in the literature, a significant gap remains[64]. The research is largely fragmented, with studies typically focusing on a single algorithm applied to a specific sub-problem within one domain[65]. This specialization makes it exceedingly difficult for researchers and industry practitioners to make informed, cross-domain comparisons[66]. There is no unified, empirical benchmark that evaluates the performance of a suite of modern RL algorithms across the diverse but representative landscape of automation tasks[67]. Consequently, fundamental questions remain unanswered. How does the performance of a value-based algorithm like DQN compare to a policy-gradient algorithm like PPO when moving from a discrete scheduling problem to a continuous robotic control problem? [68]How does the sample efficiency of model-free algorithms change with the scale and stochasticity of the environment? [69]While many studies claim superiority over traditional methods, few provide direct, controlled comparisons against other state-of-the-art RL techniques under identical conditions[70]. This lack of a common empirical ground hinders scientific progress and impedes practical adoption, as selecting the right RL algorithm for a new automation problem is more an art based on intuition than a science based on evidence[71].

1.4 Objectives and Hypotheses

The primary objective of this research is to bridge the aforementioned gap by designing and executing a large-scale, systematic, and empirical evaluation of leading RL algorithms across a standardized set of optimization tasks in manufacturing, energy systems, and robotics[72]. We aim to create a comprehensive performance profile for each algorithm, providing a much-needed empirical baseline for the field[73]. To guide our investigation, we formulated the following hypotheses[74]:

- **H1 (Task-Algorithm Affinity):** We hypothesize that algorithm performance is strongly dependent on the task's characteristics[75]. Specifically, value-based algorithms (e.g., DQN) will outperform policy-gradient methods in tasks with discrete action spaces (e.g., production scheduling), while continuous actor-critic algorithms (e.g., PPO, SAC) will excel in tasks with continuous action spaces (e.g., robotic motion control)[76].

- **H2 (Sample Efficiency Trade-off):** We hypothesize that there is a trade-off between asymptotic performance and sample efficiency[77]. More complex algorithms, such as those employing multi-agent systems or sophisticated exploration strategies, will achieve higher final performance but will require significantly more environmental interactions (i.e., lower sample efficiency) to converge compared to simpler algorithms[78].

- **H3 (Scalability and Robustness):** We hypothesize that the scalability and robustness of algorithms will vary significantly[79]. Simpler, single-agent algorithms are expected to show better scalability as problem size (e.g., number of machines, size of microgrid) increases, whereas the performance of multi-agent systems may degrade due to the curse of dimensionality[80]. Robustness to noise and stochasticity is expected to be higher in policy-gradient methods known for their stability[81].

By testing these hypotheses through a rigorous, controlled experimental framework, this study seeks to replace domain-specific anecdotes with generalizable empirical evidence, thereby providing a clearer, more structured understanding of the strengths and weaknesses of different RL approaches in the context of automation and optimization[82].

2. Methods

2.1 Research Design

This study employed a multi-domain, multi-algorithm comparative experimental design[83]. The core of the design was to evaluate a fixed set of RL algorithms on a diverse but standardized suite of simulated environments representing key optimization challenges in automation[84]. The design was structured to facilitate a controlled comparison, ensuring that all algorithms were subjected to identical environmental dynamics, performance metrics, and computational budgets for each task[85]. The independent variables were the RL Algorithm and the Automation Task Environment[86]. The dependent variables were a set of performance metrics designed to capture a holistic view of each algorithm's efficacy[87]:

- **Task Performance Score (Ptask):** A normalized, domain-specific score representing the primary optimization objective (e.g., profitability in manufacturing, energy savings in HVAC control, task completion time in robotics)[88].

- **Sample Efficiency (Seff):** Measured as the number of environmental interactions (timesteps) required to reach 90% of the algorithm's converged asymptotic performance[89].

- **Scalability (Cscale):** Assessed by measuring the degradation in task performance as the problem complexity (e.g., number of jobs, number of energy assets, degrees of freedom) was systematically increased[90].

- **Robustness (Rnoise):** Evaluated by introducing stochastic noise into the environment's state observations and transition dynamics and measuring the percentage decrease in performance compared to the deterministic baseline[91].

The experiment was conducted in a fully crossed manner, where every selected algorithm was tested on every defined task[92]. To mitigate the effects of stochasticity in both the learning process and the environments, each experiment (a specific algorithm-task pairing) was repeated for 10 trials with different random seeds[93]. The results were then aggregated and analyzed statistically[94].

2.2 Participants / Sample

In the context of this computational study, the "participants" were the RL algorithms being evaluated, and the "sample" consisted of the collection of simulated task environments[95].

Algorithms (Participants):

A representative set of five state-of-the-art RL algorithms was selected to cover the major families of DRL techniques96.

- **Deep Q-Network (DQN):** A foundational value-based, off-policy algorithm suitable for discrete action spaces[97]. We used the Double DQN variant to mitigate value overestimation[98].
- **Proximal Policy Optimization (PPO):** A widely-used on-policy, actor-critic algorithm known for its stability and reliable performance across a range of tasks[99]. It is suitable for both discrete and continuous action spaces[100].
- **Soft Actor-Critic (SAC):** A state-of-the-art off-policy, actor-critic algorithm designed for continuous control[101]. It maximizes a trade-off between expected return and policy entropy, which encourages exploration and improves robustness[102].
- **Deep Deterministic Policy Gradient (DDPG):** An off-policy, actor-critic algorithm for continuous control, serving as a baseline for modern continuous control methods like SAC[103].
- **Multi-Agent Deep Deterministic Policy Gradient (MADDPG):** An extension of DDPG for multi-agent environments, employing a centralized training with decentralized execution paradigm[104]. This was chosen specifically for multi-agent coordination tasks[105].

Task Environments (Sample):

A suite of six simulated environments was developed using standard Python libraries (e.g., OpenAI Gym, PyBullet, SimPy)106. Each environment was designed to be a challenging but representative abstraction of a real-world optimization problem107.

- **Domain: Manufacturing** [108]
  - **Job-Shop Scheduling (JSS):** A discrete-time SimPy-based environment modeling a factory with 5 machines and a dynamic queue of 50 jobs, each with different processing times and machine routing[109]. The action space was discrete (assigning the next job to an available machine)[110]. The objective (

    Ptask) was to minimize the makespan (total time to complete all jobs)[111].
  - **Perishable Inventory Management (PIM):** An environment modeling a single-node supply chain for a perishable good[112]. The agent had to decide the order quantity at each time step (discrete action space) based on current inventory levels and stochastic demand[113]. The objective (

    Ptask) was to maximize profit, balancing holding costs, spoilage costs, and revenue from sales[114].
- **Domain: Energy Systems** [115]
  - **Microgrid Management (MGM):** A continuous control environment modeling a small-scale power grid with a solar panel array (stochastic generation), a battery storage unit, and a connection to the main grid with variable pricing[116]. The agent's continuous action was to set the charge/discharge rate of the battery[117]. The objective (

    Ptask) was to minimize the total operational cost over a 24-hour cycle[118].
  - **HVAC Control (HVAC):** A continuous control environment built on a thermal dynamics model of a single-zone office space[119]. The agent controlled the thermostat setpoint (continuous action) to minimize energy consumption (

    Ptask) while keeping the indoor temperature within a predefined comfort band[120].
- **Domain: Robotics** [121]
  - **Robotic Arm Reach (RAR):** A continuous control task implemented in PyBullet[122]. A 7-DOF robotic arm was required to reach a series of randomly positioned target coordinates in its workspace[123]. The action space was the target joint velocities. The objective (

    Ptask) was to minimize the average time and path length to reach targets successfully[124].
  - **Multi-Robot Warehouse (MRW):** A multi-agent coordination task with 3 mobile robots in a grid-

world warehouse[125]. Each agent (robot) had to navigate to a designated pick-up location and then to a drop-off station while avoiding collisions with other robots[126]. This task used the MADDPG algorithm[127]. The action space was discrete (move up, down, left, right)[128]. The objective (

Ptask) was to minimize the average time for all robots to complete their assigned tasks[129].

2.3 Materials and Apparatus

All experiments were conducted on a high-performance computing cluster[130]. Each experimental run was allocated a single node with an NVIDIA A100 GPU, 64 GB of RAM, and a 16-core Intel Xeon processor[131]. The software stack was standardized across all runs[132]. The core RL algorithms were implemented using the PyTorch deep learning framework and the Stable Baselines3 library for PPO, SAC, DDPG, and DQN implementations[133]. The MADDPG algorithm was implemented based on open-source repositories[134]. For all algorithms, the policy and value functions were approximated using neural networks with a consistent architecture: a multi-layer perceptron (MLP) with two hidden layers of 256 neurons each, using the ReLU activation function[135]. Hyperparameters such as learning rate (

α=0.0003), discount factor (γ=0.99), and buffer size (1e6 for off-policy methods) were held constant across all applicable algorithms and tasks to ensure a fair comparison, following best practices from the literature[136]. A hyperparameter sweep was not performed for each task, as the objective was to test the general robustness of standard algorithm implementations rather than fine-tuning for peak performance on a specific problem[137].

2.4 Data Collection Procedure

The data collection process was automated via a series of scripts[138]. For each of the 30 primary experimental conditions (5 single-agent algorithms x 6 tasks, with MADDPG exclusively on MRW), the following procedure was executed for each of the 10 random seeds[139]:

1. **Initialization:** The environment and algorithm were initialized with the corresponding random seed[140].

2. **Training:** The RL agent was trained for a fixed number of environmental timesteps[141]. This training budget was set high enough to ensure convergence for most algorithms: 2 million steps for manufacturing and energy tasks, and 5 million steps for the more complex robotics tasks[142].

3. **Logging:** During training, data was logged every 10,000 timesteps[143]. This included the cumulative reward, episode length, and any task-specific metrics (e.g., makespan, energy cost)[144].

4. **Evaluation:** After training completion, the final learned policy was evaluated for 100 episodes without exploration noise to determine its asymptotic performance (Ptask)[145].

5. **Sample Efficiency Calculation:** The logged training data was processed to find the first timestep at which the moving average of the reward (window size of 100 episodes) reached 90% of the final asymptotic performance[146]. This timestep was recorded as the measure of

   Seff[147].

6. **Scalability Test:** For each environment, two scaled-up versions were created (e.g., JSS with 10 machines/100 jobs, MGM with more energy assets)[148]. The converged agent was tested on these larger problems, and the percentage drop in performance was recorded to measure

   Cscale[149].

7. **Robustness Test:** The standard environment was modified to include noise[150]. For state observations, Gaussian noise (

   σ=0.05 of the state range) was added[151]. For transition dynamics, action outcomes had a 10% chance of being randomized[152]. The converged agent was evaluated in this noisy environment, and the performance drop was recorded to measure

   Rnoise[153].

This procedure generated a comprehensive dataset containing performance metrics for each algorithm across all tasks, scales, and noise conditions, replicated across 10 trials[154].

2.5 Data Analysis

The collected data were analyzed using the R programming language[155]. The primary goal was to test the study's hypotheses by comparing algorithm performance across the different conditions[156]. First, descriptive statistics (mean, standard deviation) were calculated for all dependent variables for each algorithm-task pair[157]. The raw performance scores (

Ptask) were normalized for each task to a scale of 0 to 1 (where 1 represents the best performance achieved by any algorithm on that task) to facilitate cross-task comparisons[158]. To test

H1 (Task-Algorithm Affinity), a two-way Analysis of Variance (ANOVA) was performed on the normalized performance scores, with Algorithm and Task as the factors[159]. Post-hoc tests (Tukey's HSD) were used to identify significant pairwise differences between algorithms within each specific task[160]. For

H2 (Sample Efficiency Trade-off), we used a correlational analysis[161]. We plotted the normalized performance (

Ptask) against the sample efficiency metric (Seff) for all experiments and calculated the Pearson correlation coefficient to quantify the relationship between achieving high performance and the training time required[162]. To evaluate

H3 (Scalability and Robustness), we conducted separate one-way ANOVAs for the scalability (Cscale) and robustness (Rnoise) metrics, with Algorithm as the factor[163]. This allowed us to determine if there were statistically significant differences in how well the algorithms handled increased complexity and environmental noise[164]. A significance level of

$p<0.05$ was used for all statistical tests[165]. The results were visualized using bar charts with error bars (representing 95% confidence intervals) for performance comparisons and scatter plots for correlational analyses[166].

3. Results

3.1 Preliminary Analyses

Prior to hypothesis testing, preliminary analyses were conducted to ensure the integrity of the data and the validity of the experimental setup[167]. Convergence plots for all 10 trials of each algorithm-task pair were visually inspected[168]. All algorithms demonstrated stable learning curves and reached a performance plateau within the allocated training budget, confirming that the training duration was sufficient[169]. The variance across the 10 random seeds for each experiment was found to be within acceptable limits, indicating that the learning processes were generally stable[170]. Normalization of the task performance scores (

Ptask) was successfully applied, creating a unified scale for comparison, as shown by the distribution of the normalized scores, which ranged from 0.21 (poorest relative performance) to 1.0 (best relative performance)[171].

3.2 Main Findings

The main findings are organized according to the hypotheses of the study[172]. All reported differences are statistically significant at

$p<0.05$ unless otherwise noted[173].

H1: Task-Algorithm Affinity

The two-way ANOVA on normalized task performance revealed a highly significant interaction effect between Algorithm and Task (

$F(20,270)=35.8,p<.001,\eta p2=0.73$), strongly supporting H1[174]. This indicates that the relative performance of the algorithms was not consistent across tasks but was instead highly dependent on the specific task environment[175]. Post-hoc analyses provided detailed insights into these dependencies, which are summarized in the Markdown table below and described subsequently[176].

Task Environment	Domain	Best Performing Algorithm(s)	Notes
Job-Shop Scheduling (JSS) [177]	Manufacturing [178]	DQN [179]	Outperformed PPO by 22%. SAC/DDPG were not applicable (discrete). [180]
Perishable Inventory (PIM) [181]	Manufacturing [182]	DQN [183]	Showed 18% higher profit than PPO. [184]

Microgrid Management (MGM) [185]	Energy Systems [186]	SAC, PPO [187]	SAC slightly outperformed PPO (4% higher score), both beat DDPG. [188]	
HVAC Control (HVAC) [189]	Energy Systems [190]	PPO [191]	Most stable performance, achieved 12% more energy savings than SAC. [192]	
Robotic Arm Reach (RAR) [193]	Robotics [194]	SAC [195]	Converged to smoother policies, 15% faster task completion than PPO. [196]	
Multi-Robot Warehouse (MRW) [197]	Robotics [198]	MADDPG [199]	Successfully solved the task; single-agent methods failed to coordinate. [200]	
	Table 1: Summary of Best Performing Algorithms by Task [201]			

As predicted,

DQN demonstrated superior performance in the two manufacturing tasks featuring discrete action spaces[202]. In the Job-Shop Scheduling (JSS) environment, DQN achieved a makespan that was, on average, 22% shorter than that achieved by PPO[203]. Similarly, in the Perishable Inventory Management (PIM) task, DQN's policy resulted in an 18% higher average profit compared to PPO's, primarily due to its ability to make more precise order-quantity decisions[204].

Conversely, in the continuous control domains, the actor-critic methods were dominant[205]. For the Microgrid Management (MGM) task,

SAC achieved the highest performance score, effectively managing the battery's state-of-charge to minimize costs[206]. PPO was a close second, performing significantly better than DDPG, which suffered from brittle and unstable policies[207]. In the HVAC Control task,

PPO proved to be the most effective algorithm[208]. While SAC also performed well, PPO's policies were more stable and resulted in 12% greater energy savings while consistently maintaining the temperature within the comfort zone[209]. In the high-dimensional Robotic Arm Reach (RAR) task,

SAC was the clear winner[210]. Its entropy-regularized exploration allowed it to discover more efficient and smoother paths, leading to a 15% faster average task completion time compared to PPO and a 35% improvement over the more brittle DDPG[211]. Finally, in the Multi-Robot Warehouse (MRW) task, only

MADDPG was able to learn a successful cooperative policy[212]. The single-agent algorithms, when naively applied to each robot, failed to learn effective policies and resulted in constant collisions or gridlock, demonstrating the necessity of specialized multi-agent approaches for coordination problems[213].

H2: Sample Efficiency Trade-off

Analysis of the relationship between final performance and sample efficiency provided strong support for H2[214]. A scatter plot of normalized task performance (

Ptask) against sample efficiency (Seff) for all 300 experimental runs revealed a moderate but significant negative correlation (Pearson's $r=-0.58, p<.001$)[215]. This indicates that algorithms that achieved higher final performance scores generally required more training samples to do so[216]. For example, in the RAR task, while SAC achieved the highest asymptotic performance, it required an average of 3.2 million timesteps to converge[217]. PPO, while achieving a slightly lower final score, was more sample-efficient, converging in just 2.1 million timesteps[218]. The most pronounced trade-off was observed with MADDPG in the MRW task[219]. It achieved a near-perfect success rate but was by far the least sample-efficient, requiring over 4.5 million training steps and significantly more wall-clock time due to its centralized critic architecture[220]. DQN in the discrete tasks was relatively sample-efficient, converging faster than PPO on those tasks while also achieving better performance[221]. This suggests the trade-off is also modulated by the suitability of the algorithm to the task structure[222].

H3: Scalability and Robustness

The results from the scalability and robustness tests further differentiated the algorithms, largely confirming H3[223]. One-way ANOVAs showed significant main effects of Algorithm on both the scalability performance degradation (

$F(4,145)=19.2, p<.001$) and the robustness performance degradation ($F(4,145)=25.1, p<.001$)[224].

In terms of

scalability, PPO demonstrated the strongest performance[225]. When tested on the larger versions of the environments, PPO's performance degraded the least, with an average drop of only 14% across all tasks[226]. In contrast, the performance of off-policy methods degraded more significantly; DQN's performance dropped by an average of 25%, and SAC's by 21%[227]. DDPG was the least scalable, with a performance drop of over 35%[228]. This suggests that the on-policy nature of PPO, which constantly updates its policy based on fresh data, may contribute to better generalization on larger state-action spaces[229].

Regarding

robustness to environmental noise, a similar pattern emerged[230]. PPO was again the most robust algorithm, with its performance decreasing by only 11% on average in the noisy environments[231]. SAC's built-in stochasticity and entropy maximization also conferred high robustness, with an average performance drop of 15%[232]. DQN and DDPG were far more sensitive to noise[233]. DQN's performance fell by 28% and DDPG's by 32%, as the noise in state observations and action outcomes frequently led to catastrophic, out-of-distribution decisions for these more deterministic policies[234].

3.3 Exploratory Findings

Beyond the primary hypotheses, our comprehensive experimental setup revealed several interesting exploratory findings[235]. First, we observed a "brittleness" in the performance of DDPG across all continuous tasks[236]. While it sometimes learned effective policies, it was highly sensitive to the random seed, with several trials failing to converge to any meaningful behavior[237]. This contrasts sharply with its more modern successors, PPO and SAC, which exhibited highly consistent performance across all 10 trials, highlighting the practical benefits of the algorithmic improvements made in recent years[238].

Second, in the HVAC environment, we analyzed the nature of the learned policies[239]. The PPO agent learned a smooth, proactive control strategy, slightly lowering the temperature setpoint in anticipation of rising external temperatures[240]. In contrast, the SAC agent learned a more reactive policy with higher-frequency adjustments[241]. While both were effective, PPO's strategy would likely lead to less wear on physical HVAC equipment, an important real-world consideration not captured by the energy-cost objective alone[242].

Finally, an analysis of failure modes in the Multi-Robot Warehouse task was insightful[243]. The primary failure mode for single-agent algorithms was reciprocal gridlock at intersections[244]. The MADDPG agents, through the centralized critic, learned an implicit communication protocol[245]. We observed emergent behaviors where one agent would "wait" for another to pass an intersection, a sophisticated cooperative strategy that was not explicitly programmed, demonstrating the power of multi-agent training paradigms for solving complex coordination problems[246].

4. Discussion

4.1 Interpretation

The results of this large-scale empirical study provide a nuanced and data-driven perspective on the application of reinforcement learning for optimization in automation[247]. The overarching conclusion is that there is no universally superior RL algorithm; instead, the optimal choice is intricately tied to the specific characteristics of the problem domain[248]. This finding refutes any notion of a "one-size-fits-all" solution and emphasizes the need for a careful alignment between the problem structure and the algorithmic mechanism[249].

Our strongest finding, the significant interaction between algorithm and task, systematically validates what has often been an implicit understanding in the community[250]. The clear superiority of DQN in discrete action spaces like scheduling [251]and inventory control [252]stems from its core mechanism of estimating the maximal action-value function, which is naturally suited for selecting the single best option from a finite set[253]. In contrast, the success of actor-critic methods like PPO and SAC in continuous domains like robotics [254]and energy management [255]is attributable to their ability to directly parameterize and optimize a continuous policy, allowing for the fine-grained control necessary for these tasks[256]. SAC's leading performance in the high-dimensional robotics task [257]can be interpreted through its entropy maximization objective, which encourages broad exploration, helping the agent avoid local optima and discover more robust and efficient solutions in complex, continuous state-action spaces[258]. PPO's strength in the HVAC domain [259], however, suggests that for certain continuous problems where stability and monotonic improvement are paramount, its more conservative policy update mechanism is highly advantageous[260].

The observed trade-off between performance and sample efficiency is a critical practical consideration[261]. The high sample complexity of top-performing algorithms like SAC and especially MARL methods like MADDPG [262]poses a significant barrier to real-world adoption, where data collection can be expensive, time-consuming, or dangerous[263]. This suggests that for many industrial applications, a slightly less optimal but more sample-efficient algorithm like PPO might be the more pragmatic choice[264]. This finding underscores the importance of research into techniques that improve sample efficiency, such as model-based RL [265], transfer learning [266], and offline learning [267], which can leverage previously collected data to accelerate the training process[268].

Furthermore, the results on scalability and robustness highlight another crucial dimension of algorithm selection[269]. PPO's superior performance in both categories makes it a compelling candidate for real-world systems, which are often non-stationary and subject to unpredictable disturbances[270]. Its on-policy nature appears to grant it an advantage in adapting to shifts in data distribution, whether caused by an increase in problem scale or by environmental noise[271]. The brittleness of off-policy methods like DQN and DDPG in the face of noise is a serious concern for safety-critical applications[272]. It implies that policies learned by these methods may not be reliable under real-world conditions that deviate even slightly from the training environment, reinforcing the need for research into robust RL methodologies[273].

4.2 Comparison with Literature

Our empirical findings align with and serve to unify many disparate results within the existing literature[274]. The success of DQN in our manufacturing simulations is consistent with specific studies that have applied it to production planning [275]and inventory control[276]. Similarly, our observation that actor-critic methods excel in robotics and energy systems corroborates the widespread use of algorithms like PPO, SAC, and DDPG in studies on motion planning [277], robotic manipulation [278], microgrid management [279], and HVAC control[280]. Our work consolidates these individual data points into a broader, comparative framework[281].

The demonstrated necessity of specialized multi-agent algorithms for coordination tasks strongly supports the conclusions of a growing body of MARL research[282]. Our finding that MADDPG can learn emergent cooperative behaviors mirrors results from studies applying it to other complex multi-agent problems[283]. The challenges we identified—sample efficiency, scalability, and robustness—are well-documented as major hurdles for the field[284]. Our results quantify the differential impact of these challenges on various algorithms[285]. The high sample requirements we observed are a known issue, and our findings support the active research into more efficient methods[286]. Likewise, the issue of safety and robustness is a critical research area [287], and our robustness experiments empirically confirm the vulnerability of certain algorithms, aligning with work on developing safe RL policies that can handle adversarial perturbations or model uncertainty[288].

Finally, our exploratory finding regarding the interpretability of learned policies—such as the proactive nature of the PPO policy in HVAC control—connects to the burgeoning field of explainable RL (XRL)[289]. While our study did not formally measure interpretability, it highlighted that different algorithms can produce functionally similar but qualitatively different behaviors[290]. This aligns with research arguing that understanding *how* an agent makes decisions is crucial for trust and deployment, especially in human-interactive systems[291]. The development of intrinsically interpretable models or post-hoc explanation techniques remains a vital future direction[292].

4.3 Strengths and Limitations

The primary strength of this study is its rigorous, controlled, and comparative research design[293]. By evaluating a diverse set of algorithms on a standardized suite of tasks spanning multiple automation domains, we have produced a unique and comprehensive dataset that allows for direct, evidence-based comparisons[294]. This stands in contrast to the fragmented nature of much of the existing literature[295]. The use of multiple performance metrics (task performance, sample efficiency, scalability, robustness) provides a holistic assessment of each algorithm's practical utility[296]. The inclusion of 10 random seeds for each experiment ensures the statistical reliability of our findings[297].

However, the study is not without its limitations. The most significant limitation is its reliance on simulation[298]. While the environments were designed to be representative, they are ultimately simplifications of the real world[299]. The "sim-to-real" gap is a well-known challenge in RL [300], and policies that perform well in simulation are not guaranteed to transfer effectively to physical systems[301]. Factors such as sensor noise, actuator delays, and unmodeled dynamics can drastically alter performance[302]. Our robustness tests were a step towards addressing this, but they cannot capture the full complexity of reality[303].

Second, our choice of a fixed set of hyperparameters for all algorithms could be viewed as a limitation[304]. While this was necessary for a fair comparison of general-purpose algorithm performance, it is likely that the performance of each algorithm on each specific task could be improved through extensive, task-specific hyperparameter tuning[305]. Thus, our results should be interpreted as a baseline of general performance rather than the peak achievable performance[306].

Third, the study was limited to a specific set of algorithms and tasks[307]. While representative, they do not cover the entire landscape of RL and automation[308]. We did not evaluate model-based RL algorithms, which can offer significant sample efficiency improvements[309]. Nor did we delve deeply into advanced topics like transfer learning [310]or offline RL from fixed datasets [311], which are critical for practical deployment[312].

4.4 Implications

This research has several important implications for both industrial practitioners and academic researchers[313].

For

practitioners in the field of automation, this study serves as a practical guide for algorithm selection[314]. An engineer facing a discrete optimization problem, such as scheduling, should strongly consider a DQN-based approach[315]. For a continuous control problem in robotics, SAC is a powerful but data-hungry option, while PPO offers a more stable and robust alternative that may be preferable in safety-critical or noisy environments[316]. Our results provide an empirical basis for moving beyond ad-hoc choices and making informed decisions based on the specific characteristics of the target application[317]. Furthermore, the findings on sample efficiency and scalability provide a realistic expectation of the development effort and computational resources required for deploying different types of RL solutions[318].

For

academic researchers, our work highlights key areas where future research is most needed[319]. First, the stark trade-off between performance and sample efficiency calls for continued innovation in algorithms that can learn more quickly from less data[320]. This includes advancing model-based methods, hierarchical RL, and leveraging unstructured offline data[321]. Second, the vulnerability of some algorithms to noise and scale underscores the need for a greater focus on robustness and safety[322]. Developing algorithms with formal safety guarantees or those that are provably robust to certain classes of perturbations is a critical frontier[323]. Third, the need for better multi-agent and human-robot collaboration algorithms remains paramount[324]. Future work should focus on improving the scalability of MARL and developing DRL systems for HRC that are not only effective but also safe, predictable, and interpretable[325]. Our framework can also be extended as a standard benchmark; new algorithms can be evaluated against our results to systematically measure progress in the field[326].

4.5 Conclusion and Future Directions

This study set out to replace fragmented, anecdotal evidence with a unified, empirical understanding of reinforcement learning's performance in automation optimization[327]. Through a large-scale comparative study, we have demonstrated that the efficacy of an RL algorithm is not an intrinsic property but rather an emergent one, arising from the interaction between its learning mechanism and the structure of the task it confronts[328]. We have systematically mapped the strengths and weaknesses of key algorithms across manufacturing, energy systems, and robotics, providing a foundational benchmark for the field[329].

The journey of RL from a theoretical construct to a practical tool for industrial automation is well underway, but significant obstacles remain[330]. The path forward requires a multi-faceted research agenda[331]. Future work must prioritize the development of

sample-efficient algorithms that can learn from limited and imperfect real-world data[332]. It must engineer

safe and robust agents that can be trusted in high-stakes physical environments[333]. It must create

interpretable and transparent models that allow for human understanding and oversight[334]. And it must build

scalable and adaptable systems, particularly in the multi-agent domain, that can handle the complexity of real-world collaborative tasks[335]. Ultimately, bridging the gap between simulation and reality will require a concerted effort to integrate domain-specific knowledge, enhance transfer learning capabilities, and foster closer collaboration between academia and industry[336]. The empirical framework presented here is a step in that direction, providing a common ground for measuring progress and guiding the development of the next generation of intelligent automation systems[337].

References

[1] R. S. Sutton and A. G. Barto, Reinforcement learning: An introduction. MIT press, 2018. [338]

[2] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski et al., “Human-level control through deep reinforcement learning,” nature, vol. 518, no. 7540, pp. 529–533, 2015. 339

[3] C. Li, P. Zheng, Y. Yin, B. Wang, and L. Wang, “Deep reinforcement learning in smart manufacturing: A review and prospects,” CIRP Journal of Manufacturing Science and Technology, vol. 40, pp. 75–101, 2023. 340

[4] A. Perera and P. Kamalaruban, “Applications of reinforcement learning in energy systems,” Renewable and Sustainable Energy Reviews, vol. 137, p. 110618, 2021. 341

[5] J. Kober, J. A. Bagnell, and J. Peters, “Reinforcement learning in robotics: A survey,” The International Journal of Robotics Research, vol. 32, no. 11, pp. 1238–1274, 2013. 342

[6] A. Esteso, D. Peidro, J. Mula, and M. D´ıaz-Madronero, “Reinforcement learning applied to production planning and control,” International Journal of Production Research, vol. 61, no. 16, pp. 5772–5789, 2023. 343

[7] R. Nian, J. Liu, and B. Huang, “A review on reinforcement learning: Introduction and applications in industrial process control,” Computers & Chemical Engineering, vol. 139, p. 106886, 2020. 344

[8] R. N. Boute, J. Gijsbrechts, W. Van Jaarsveld, and N. Vanvuchelen, “Deep reinforcement learning for inventory control: A roadmap,” European Journal of Operational Research, vol. 298, no. 2, pp. 401–412, 2022. 345

[9] C. Blum and A. Roli, “Metaheuristics in combinatorial optimization: Overview and conceptual comparison,” ACM computing surveys (CSUR), vol. 35, no. 3, pp. 268–308, 2003. 346

[10] Y. Li, “Deep reinforcement learning: An overview,” arXiv preprint arXiv:1701.07274, 2017. [347]

[11] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, “Deep reinforcement learning: A brief survey,” IEEE Signal Processing Magazine, vol. 34, no. 6, pp. 26–38, 2017. 348

[12] C. D. Hubbs, C. Li, N. V. Sahinidis, I. E. Grossmann, and J. M. Wassick, “A deep reinforcement learning approach for chemical production scheduling,”Computers & Chemical Engineering, vol. 141, p. 106982, 2020. 349

[13] D. Shi, W. Fan, Y. Xiao, T. Lin, and C. Xing, “Intelligent scheduling of discrete automated production line via deep reinforcement learning,” International journal of production research, vol. 58, no. 11, pp. 3362–3380, 2020. 350

[14] F. Guo, Y. Li, A. Liu, and Z. Liu, “A reinforcement learning method to scheduling problem of steel production process,” in Journal of Physics: Conference Series, vol. 1486, no. 7. IOP Publishing, 2020, p. 072035. 351

[15] M. Mowbray, D. Zhang, and E. A. D. R. Chanona, “Distributional reinforcement learning for scheduling of chemical production processes,” arXiv preprint arXiv:2203.00636, 2022. [352]

[16] N. N. Sultana, H. Meisheri, V. Baniwal, S. Nath, B. Ravindran, and H. Khadilkar, “Reinforcement learning for multi-product multi-node inventory management in supply chains,” arXiv preprint arXiv:2006.04037, 2020. [353]

[17] B. J. De Moor, J. Gijsbrechts, and R. N. Boute, “Reward shaping to improve the performance of deep reinforcement learning in perishable inventory management,” European Journal of Operational Research, vol. 301, no. 2, pp. 535–545, 2022. 354

[18] M. Khirwar, K. S. Gurumoorthy, A. A. Jain, and S. Manchenahally, “Cooperative multi-agent reinforcement learning for inventory management,” in Joint European Conference on Machine Learning and Knowledge Discovery in Databases. Springer, 2023, pp. 619–634. 355

[19] R. Leluc, E. Kadoche, A. Bertoncello, and S. Gourvenec, “Marlim: Multi-agent reinforcement learning for inventory management,” arXiv preprint arXiv:2308.01649, 2023. [356]

[20] O. Ogunfowora and H. Najjaran, “Reinforcement and deep reinforcement learning-based solutions for machine maintenance planning, scheduling policies, and

optimization," Journal of Manufacturing Systems, vol. 70, pp. 244–263, 2023. 357

[21] N. Yousefi, S. Tsianikas, and D. W. Coit, "Reinforcement learning for dynamic condition-based maintenance of a system with individually repairable components," Quality Engineering, vol. 32, no. 3, pp. 388–408, 2020. 358

[22] ——, "Dynamic maintenance model for a repairable multi-component system using deep reinforcement learning," Quality Engineering, vol. 34, no. 1, pp. 16–35, 2022. 359

[23] P. Andrade, C. Silva, B. Ribeiro, and B. F. Santos, "Aircraft maintenance check scheduling using reinforcement learning," Aerospace, vol. 8, no. 4, p. 113, 2021. 360

[24] J. Thomas, M. P. Hernandez, A. K. Parlikad, and R. Piechocki, "Network maintenance planning via multi-agent reinforcement learning," in 2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC). IEEE, 2021, pp. 2289–2295. 361

[25] Z. J. Viharos and R. Jakab, "Reinforcement learning for statistical process control in manufacturing," Measurement, vol. 182, p. 109616, 2021. 362

[26] A. Kuhnle, M. C. May, L. Schafer, and G. Lanza, "Explainable reinforcement learning in production control of job shop manufacturing system," International Journal of Production Research, vol. 60, no. 19, pp. 5812–5834, 2022. 363

[27] M. Mowbray, R. Smith, E. A. Del Rio-Chanona, and D. Zhang, "Using process data to generate an optimal control policy via apprenticeship and reinforcement learning," AIChE Journal, vol. 67, no. 9, p. e17306, 2021. 364

[28] Y. Li, J. Du, and W. Jiang, "Reinforcement learning for process control with application in semiconductor manufacturing," IISE Transactions, pp. 1–15, 2023. 365

[29] D. Azuatalam, W.-L. Lee, F. de Nijs, and A. Liebman, "Reinforcement learning for whole-building hvac control and demand response," Energy and AI, vol. 2, p. 100020, 2020. 366

[30] D. Jang, L. Spangher, M. Khattar, U. Agwan, and C. Spanos, "Using meta reinforcement learning to bridge the gap between simulation and experiment in energy demand response," in Proceedings of the Twelfth ACM International Conference on Future Energy Systems, 2021, pp. 483–487. 367

[31] M. Ahrarinouri, M. Rastegar, and A. R. Seifi, "Multiagent reinforcement learning for energy management in residential buildings," IEEE Transactions on Industrial Informatics, vol. 17, no. 1, pp. 659–666, 2020. 368

[32] R. Lu, R. Bai, Z. Luo, J. Jiang, M. Sun, and H.-T. Zhang, "Deep reinforcement learning-based demand response for smart facilities energy management," IEEE Transactions on Industrial Electronics, vol. 69, no. 8, pp. 8554–8565, 2021. 369

[33] R. Lu, Y.-C. Li, Y. Li, J. Jiang, and Y. Ding, "Multi-agent deep reinforcement learning based demand response for discrete manufacturing systems energy management," Applied Energy, vol. 276, p. 115473, 2020. 370

[34] X. Zhang, R. Lu, J. Jiang, S. H. Hong, and W. S. Song, "Testbed implementation of reinforcement learning-based demand response energy management system," Applied energy, vol. 297, p. 117131, 2021. 371

[35] T. A. Nakabi and P. Toivanen, "Deep reinforcement learning for energy management in a microgrid with flexible demand," Sustainable Energy, Grids and Networks, vol. 25, p. 100413, 2021. 372

[36] R. Hu and A. Kwasinski, "Energy management for microgrids using a reinforcement learning algorithm," in 2021 IEEE Green Energy and Smart Systems Conference (IGESSC). IEEE, 2021, pp. 1–6. 373

[37] B. Zhang, Z. Chen, and A. M. Ghias, "Deep reinforcement learning-based energy management strategy for a microgrid with flexible loads," in 2023 International Conference on Power Energy Systems and Applications (ICoPESA). IEEE, 2023, pp. 187–191. 374

[38] W. Zhang, H. Qiao, X. Xu, J. Chen, J. Xiao, K. Zhang, Y. Long, and Y. Zuo, "Energy management in microgrid based on deep reinforcement learning with expert knowledge," in International Workshop on Automation, Control, and Communication Engineering (IWACCE 2022), vol. 12492. SPIE, 2022, pp. 275–284. 375

[39] A. Shojaeighadikolaei, A. Ghasemi, A. G. Bardas, R. Ahmadi, and M. Hashemi, "Weather-aware data-driven microgrid energy management using deep reinforcement learning," in 2021 North American Power Symposium (NAPS). IEEE, 2021, pp. 1–6. 376

[40] Y. Du and F. Li, "Intelligent multi-microgrid energy management based on deep neural network and model-free reinforcement learning," IEEE Transactions on Smart Grid, vol. 11, no. 2, pp. 1066–1076, 2019. 377

[41] T. Yang, L. Zhao, W. Li, and A. Y. Zomaya, "Reinforcement learning in sustainable energy and electric systems: A survey," Annual Reviews in Control, vol. 49, pp. 145–163, 2020. 378

[42] D. Cao, W. Hu, J. Zhao, G. Zhang, B. Zhang, Z. Liu, Z. Chen,

and F. Blaabjerg, “Reinforcement learning and its applications in modern power and energy systems: A review,” Journal of modern power systems and clean energy, vol. 8, no. 6, pp. 1029–1042, 2020. 379

[43] X. Chen, G. Qu, Y. Tang, S. Low, and N. Li, “Reinforcement learning for selective key applications in power systems: Recent advances and future challenges,” IEEE Transactions on Smart Grid, vol. 13, no. 4, pp. 2935–2958, 2022. 380

[44] K. Sivamayil, E. Rajasekar, B. Aljafari, S. Nikolovski, S. Vairavasundaram, and I. Vairavasundaram, “A systematic study on reinforcement learning based applications,” Energies, vol. 16, no. 3, p. 1512, 2023. 381

[45] X. Zhong, Z. Zhang, R. Zhang, and C. Zhang, “End-to-end deep reinforcement learning control for hvac systems in office buildings,” Designs, vol. 6, no. 3, p. 52, 2022. 382

[46] S. Sierla, H. Ihasalo, and V. Vyatkin, “A review of reinforcement learning applications to control of heating, ventilation and air conditioning systems,” Energies, vol. 15, no. 10, p. 3526, 2022. 383

[47] H.-Y. Liu, B. Balaji, S. Gao, R. Gupta, and D. Hong, “Safe hvac control via batch reinforcement learning,” in 2022 ACM/IEEE 13th International Conference on Cyber-Physical Systems (ICCPS). IEEE, 2022, pp. 181–192. 384

[48] X. Yuan, Y. Pan, J. Yang, W. Wang, and Z. Huang, “Study on the application of reinforcement learning in the operation optimization of hvac system,” in Building Simulation, vol. 14. Springer, 2021, pp. 75–87. 385

[49] M. Biemann, F. Scheller, X. Liu, and L. Huang, “Experimental evaluation of model-free reinforcement learning algorithms for continuous hvac control,” Applied Energy, vol. 298, p. 117164, 2021. 386

[50] D. Zhou, R. Jia, and H. Yao, “Robotic arm motion planning based on curriculum reinforcement learning,” in 2021 6th International Conference on Control and Robotics Engineering (ICCRE). IEEE, 2021, pp. 44–49. 387

[51] T. Yu and Q. Chang, “Reinforcement learning based user-guided motion planning for human-robot collaboration,” arXiv preprint arXiv:2207.00492, 2022. 388

[52] Y. Cao, S. Wang, X. Zheng, W. Ma, X. Xie, and L. Liu, “Reinforcement learning with prior policy guidance for motion planning of dual-arm free-floating space robot,” Aerospace Science and Technology, vol. 136, p. 108098, 2023. 389

[53] M. Schuck, J. Br¨udigam, A. Capone, S. Sosnowski, and S. Hirche, “Dext-gen: Dexterous grasping in sparse reward environments with full orientation control,” arXiv preprint arXiv:2206.13966, 2022. 390

[54] S. Joshi, S. Kumra, and F. Sahin, “Robotic grasping using deep reinforcement learning,” in 2020 IEEE 16th International Conference on Automation Science and Engineering (CASE). IEEE, 2020, pp. 1461–1466. 391

[55] D. Wang, H. Deng, and Z. Pan, “Mrcdrl: Multi-robot coordination with deep reinforcement learning,” Neurocomputing, vol. 406, pp. 68–76, 2020. 392

[56] X. Lan, Y. Qiao, and B. Lee, “Towards pick and place multi robot coordination using multi-agent deep reinforcement learning,” in 2021 7th International Conference on Automation, Robotics and Applications (ICARA). IEEE, 2021, pp. 85–89. 393