Article

A Brief Survey of Deep Reinforcement Learning Algorithms for Autonomous Systems

Maxwell Khan , Jackson Reynolds , Madison Taylor , Caleb Walker , Savannah Mitchell , Ethan Carter , Emma Davis *

Posted Date: 22 January 2026

doi: 10.20944/preprints202601.1653.v1

Keywords: deep learning; neural networks; feature extraction; supervised learning; transfer learning; computer vision

 doi:10.20944/preprints202601.1653.v1

Article

A Brief Survey of Deep Reinforcement Learning Algorithms for Autonomous Systems

Maxwell Khan [1], Jackson Reynolds [1], Madison Taylor [1], Caleb Walker [1], Savannah Mitchell [1], Ethan Carter [1] and Emma Davis [2,*]

[1] Christian Brothers University, Memphis, TN, 38104, USA
[2] Mississippi State University, Mississippi State, MS 39762, USA
* Correspondence: edavis@cse.msstate.edu

Abstract

The advent of autonomous systems has propelled the integration of artificial intelligence (AI) and machine learning (ML) techniques, particularly deep reinforcement learning (DRL), to enhance decision-making capabilities. This research paper conducts an exhaustive survey of state-of-the-art DRL algorithms, focusing on their applicability and performance within the realm of autonomous systems. To find out how flexible and useful DRL algorithms are in real life, our research covers a lot of different areas, such as robots, self-driving cars, and unmanned flying vehicles. The study takes a close look at the most important parts of these algorithms, like neural network designs, exploration-exploitation strategies, and payment processes, to see how they affect how well independent systems work. Additionally, the study goes into detail about the problems and restrictions that come with using DRL in self-driving systems, covering everything from sample waste to safety concerns. We also look at new developments and improvements in DRL that might be able to get around current problems and make way for future innovations in driverless technology. As a valuable resource for researchers, engineers, and practitioners working on the development and deployment of autonomous systems, this brief survey shows the pros, cons, and opportunities that come with different DRL algorithms in this ever-changing field.

Keywords: deep learning; neural networks; feature extraction; supervised learning; transfer learning; computer vision

1. Introduction

The convergence of autonomous systems and artificial intelligence (AI) has become a significant catalyst for change in numerous sectors in recent times. Autonomous systems hold great potential due to their capacity for intelligent decision-making and adaptability in dynamic environments, thereby promoting innovation, safety, and efficiency. Central to this technological advancement is the domain of deep reinforcement learning (DRL), which is a specialised branch of machine learning (ML) and has demonstrated exceptional proficiency in educating agents to generate decisions via iterative processes.

Unmanned aerial vehicles, robotic platforms, and self-driving vehicles are examples of autonomous systems that are gaining prevalence in contemporary society. As these systems traverse and engage with intricate real-life situations, the requirement for advanced decision-making mechanisms increases significantly. Deep reinforcement learning (DRL), due to its capability of optimising decision policies and learning from experience in intricate settings, has emerged as a central focus in efforts to improve the intelligence and autonomy of such systems.

The primary aim of this scholarly article is to present an all-encompassing examination of the varied terrain of DRL algorithms and their implementations within the realm of autonomous systems. The survey's scope extends to a brief examination of the methodologies, architectures, and obstacles that arise when implementing DRL algorithms in practical situations. Through a rigorous analysis

 doi:10.20944/preprints202601.1653.v1

of the current state-of-the-art, our objective is to provide significant insights regarding the merits and drawbacks of established DRL methodologies. This will serve as a cornerstone for subsequent developments in the domain.

1.1. Motivation

The rationale for conducting this research arises from the exponential growth of autonomous systems across multiple industries, such as manufacturing, healthcare, and transportation. The seamless operation of these systems in dynamic and complex environments necessitates the implementation of adaptive decision-making capabilities. Due to its capacity to acquire knowledge through environmental interactions, DRL has the potential to fundamentally transform the manner in which autonomous systems perceive, analyse, and react to stimuli.

The rationale is additionally strengthened by the growing intricacy and variety of practical situations that autonomous systems confront. The requirements of these systems, which range from traversing hazardous terrains to making critical decisions in an instant, require sophisticated learning algorithms capable of generalising to a wide range of tasks and guaranteeing dependable performance.

1.2. Scope and Organization

This research paper is structured to provide a brief overview of DRL algorithms in the context of autonomous systems. The subsequent sections will delve into the fundamental principles of DRL, explore the architectures employed in various algorithms, and analyze their performance in different application domains. We will also address the challenges and limitations associated with the deployment of DRL in real-world scenarios.

By systematically exploring these aspects, this research paper aims to contribute a brief resource for researchers, engineers, and practitioners involved in the development and deployment of autonomous systems. Through a detailed analysis of DRL algorithms and their applications, we seek to facilitate a deeper understanding of the current state of the field and inspire future innovations that will drive the evolution of autonomous technologies.

2. Related Work

The domain of deep reinforcement learning (DRL) in the context of autonomous systems is dynamic and perpetually progressing. This section delves into the pertinent literature that has served as the foundation for our exhaustive examination. Our evaluation incorporates a wide range of research endeavours, with a specific emphasis on the various applications and developments of DRL algorithms that are customised for autonomous systems.

2.1. Foundations in Reinforcement Learning

In order to provide context for our investigation [1–11], it is essential to recognise the seminal studies in the field of reinforcement learning (RL). Initial contributions established the principles of learning from interactions and optimising decision-making policies, thereby laying the foundation for DRL. Watkins and Dayan's (1992) groundbreaking research on Q-learning established a strong basis for value-based reinforcement learning [12–18] algorithms, a fundamental principle that has had an impact on subsequent advancements in DRL.

The 1998 publication "Reinforcement Learning: An Introduction" by Sutton and Barto continues to be an indispensable resource for comprehending RL algorithms, concepts, and applications. The principles that have been clarified in this work have influenced the course of RL research in numerous domains, transcending disciplinary boundaries.

2.2. Emergence of Deep Reinforcement Learning

By introducing deep neural networks into RL, a paradigm shift was initiated that led to the emergence of the DRL discipline. Significant contributions have been made, such as the work on Deep Q Networks (DQN) by Mnih et al. (2013), which proved that deep learning can effectively approximate

 doi:10.20944/preprints202601.1653.v1

complex value functions. DQN's mastery of Atari 2600 games demonstrated the capability of DRL to attain performance levels comparable to those of humans across a variety of environments.

Further investigations built upon the foundations of DRL, introducing algorithms such as Schulman et al.'s (2015) Trust Region Policy Optimisation (TRPO) and Schulman et al.'s (2017) Proximal Policy Optimisation (PPO), which tackled issues pertaining to sample efficiency and policy optimisation, respectively. These developments established the necessary foundation for the implementation of DRL in autonomous systems.

2.3. Applications in Robotics and Control

Considerable interest has been directed towards the convergence of DRL and robotics, as scholars endeavour to equip autonomous systems with the capacity for adaptive decision-making. The concept of Guided Policy Search (GPS) was first introduced by Gu et al. (2016), who demonstrated its potential in instructing robotics to execute intricate tasks with minimal human involvement. This work highlighted the potential of DRL for robotic applications in the actual world.

Haarnoja et al. (2018) made a scholarly contribution to the domain by developing Soft Actor-Critic (SAC), an off-policy algorithm specifically engineered to tackle the complexities linked to sample efficiency and intricate control tasks. There is now considerable interest in implementing DRL algorithms for fine-grained control in autonomous systems due to the success of SAC in continuous control domains.

2.4. DRL in Autonomous Vehicles

Investigative endeavours have been prompted by the emergence of autonomous vehicles to improve their decision-making capabilities via DRL. Bojarski et al. (2016) introduced a DRL approach that encompasses the entire lifecycle of autonomous vehicles, showcasing the potential of deriving driving policies directly from unprocessed sensor information. The research underscored the potential of DRL to surmount conventional obstacles in autonomous vehicle perception and control.

Additional research efforts, including the multi-agent reinforcement learning for vehicle cooperation study by Chen et al. (2020), emphasise the criticality of collaborative decision-making in intricate traffic situations. These advancements underscore the adaptability of DRL when it comes to tackling the complexities of autonomous vehicle navigation.

2.5. Unmanned Aerial Vehicles (UAVs) and DRL

Unmanned aerial vehicles (UAVs) are an additional field in which DRL has demonstrated its versatility. Usenko et al. (2017) applied DRL to develop a vision-based control strategy for unmanned aerial vehicles (UAVs), showcasing the capability of acquiring resilient control policies through visual input. This study paves the way for the incorporation of DRL into unmanned aerial vehicle (UAV) applications, including environmental monitoring and surveillance.

Contemporary developments, exemplified by Zhu et al.'s (2021) research on multi-UAV collaboration utilising DRL, explore the intricate nature of organising numerous autonomous entities within ever-changing surroundings. The research offers valuable perspectives on the obstacles and possible remedies associated with the utilisation of DRL in cooperative UAV expeditions.

3. Problem Statement

Deep reinforcement learning (DRL) algorithm integration has revolutionised intelligent and adaptive decision-making in autonomous systems. Notwithstanding the notable advancements made in utilising DRL to improve autonomy, the domain encounters complex obstacles that require a thorough analysis. This section presents the problem statement, which outlines the primary obstacles that hinder the smooth implementation of DRL in autonomous systems.

 doi:10.20944/preprints202601.1653.v1

3.1. Notation and Definitions

Let $\mathcal{S}$ denote the state space, $\mathcal{A}$ the action space, and $\mathcal{R}$ the reward space. The dynamics of the environment are captured by the transition function $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$, where $P(s', a, s)$ represents the probability of transitioning from state s to state s' given action a. The reward function is denoted as $R : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$, with $R(s, a, s')$ yielding the immediate reward associated with the state-action-state transition.

Consider an autonomous system navigating through a sequence of discrete time steps, with the learning agent interacting with the environment to maximize the cumulative expected return. The agent's policy, denoted by $\pi : \mathcal{S} \times \mathcal{A} \to [0, 1]$, maps states to probability distributions over actions. The objective is to find an optimal policy π^* that maximizes the expected cumulative return, expressed as:

$$\pi^* = \arg\max_{\pi} \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R_{t+1}\right]$$

Here, γ represents the discount factor, ensuring convergence of the infinite sum. The optimal policy π^* is sought to navigate the autonomous system through diverse and dynamic environments, reflecting the inherent complexity of real-world scenarios.

3.2. Sample Inefficiency

A prominent challenge in the application of DRL to autonomous systems lies in sample inefficiency. Traditional DRL algorithms often require an extensive number of interactions with the environment to learn effective policies, rendering them impractical for real-world deployment where each interaction may incur substantial costs or risks. The problem can be mathematically formulated as follows:

$$\text{Minimize } \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{\infty} \gamma^t R_{t+1}\right], \text{ subject to } \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{\infty} \gamma^t N_t\right] \leq B$$

Here, τ represents a trajectory, π is the policy, R_t is the reward at time step t, and N_t is the number of samples collected at time step t. The objective is to minimize the expected cumulative return while adhering to a constraint on the total number of samples collected.

3.3. Safety Concerns

Ensuring the safety of autonomous systems is paramount, particularly when deploying DRL algorithms in critical domains such as autonomous vehicles and robotics. Safety concerns arise due to the exploration-exploitation dilemma, where the learning agent must balance the acquisition of new information with the exploitation of known policies to avoid catastrophic outcomes. Mathematically, safety can be framed as a constraint on the probability of encountering risky situations:

$$\mathbb{P}\left(\bigcup_{t=0}^{\infty} \{s_t \in \mathcal{S}_{\text{danger}}\}\right) \leq \epsilon$$

Here, $\mathcal{S}_{\text{danger}}$ represents the set of states associated with potentially hazardous situations, and ϵ is a predefined safety threshold. The challenge is to develop DRL algorithms that balance exploration and exploitation while adhering to stringent safety constraints.

3.4. Lack of Interpretability

The interpretability of DRL algorithms poses a critical challenge, especially in applications where human understanding and trust are imperative. Traditional DRL models, often represented by complex neural network architectures, lack transparency in decision-making processes. This lack of interpretability hinders the adoption of DRL in safety-critical applications. The problem can be expressed as:

 doi:10.20944/preprints202601.1653.v1

$$\text{Maximize interpretability of } \pi \text{ subject to Performance}(\pi) \geq \text{Performance}(\pi^*)$$

Here, π^* denotes the optimal policy, and the objective is to find a policy π that maximizes interpretability while maintaining performance comparable to the optimal policy.

4. General Architecture of Deep Reinforcement Learning for Autonomous Systems

The architecture of deep reinforcement learning (DRL) models for autonomous systems is a complex interplay of neural network components, decision processes, and environmental interactions. In this section, we provide a detailed exposition of the general architecture, employing mathematical notations to elucidate the key elements and their interactions.

4.1. Markov Decision Process (MDP) Formulation

The foundation of DRL lies in the formulation of the problem as a Markov Decision Process (MDP). Let $\mathcal{S}$ denote the state space, $\mathcal{A}$ the action space, and P the transition probability function. The MDP is characterized by a tuple $(\mathcal{S}, \mathcal{A}, P, R, \gamma)$, where $R : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ represents the reward function, and γ is the discount factor ensuring convergence of the infinite sum.

At each time step t, the autonomous system observes the current state $s_t \in \mathcal{S}$, selects an action $a_t \in \mathcal{A}$ according to a policy π, and transitions to a new state s_{t+1} with the associated reward R_t. The objective is to learn an optimal policy π^* that maximizes the expected cumulative return.

4.2. Neural Network Architectures

The representation power of deep neural networks [19–22]. plays a pivotal role in DRL for handling high-dimensional state spaces. Let f_θ denote the parameterized function of a neural network with parameters θ. The state-value function $V^\pi(s)$ and the action-value function $Q^\pi(s, a)$ are approximated using neural networks:

$$V^\pi(s) \approx f_\theta(s), \quad Q^\pi(s, a) \approx f_\theta(s, a)$$

Here, $V^\pi(s)$ estimates the expected return from state s under policy π, while $Q^\pi(s, a)$ estimates the expected return from taking action a in state s under policy π. The parameters θ are learned through backpropagation and optimization techniques such as stochastic gradient descent.

4.3. Policy Gradient Methods

To optimize the policy π, policy gradient methods are commonly employed. The policy π is parameterized by a neural network, and the objective is to maximize the expected cumulative return. The policy gradient is computed using the gradient of the expected return with respect to the policy parameters:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta} \left[\sum_{t=0}^{\infty} \nabla_\theta \log \pi_\theta(a_t|s_t) \cdot \hat{A}_t \right]$$

Here, τ represents a trajectory, $\pi_\theta(a_t|s_t)$ is the probability of taking action a_t in state s_t under policy π_θ, and $\hat{A}_t$ is an estimator of the advantage function. The policy parameters are updated in the direction of the policy gradient to improve the policy's performance.

4.4. Actor-Critic Architectures

Actor-Critic architectures combine the strengths of policy-based and value-based methods. The actor (policy) is responsible for selecting actions, while the critic (value function) evaluates the chosen actions. The objective is to simultaneously learn a policy and estimate the value function. The update equations for the actor and critic are given by:

 doi:10.20944/preprints202601.1653.v1

$$\theta \leftarrow \theta + \alpha_\theta \nabla_\theta J(\theta), \quad w \leftarrow w + \alpha_w \nabla_w (R_t - V_w(s_t))^2$$

Here, θ are the policy parameters, w are the value function parameters, α_θ and α_w are the learning rates, and $J(\theta)$ is the policy objective function. The value function $V_w(s_t)$ is updated towards reducing the difference between the predicted and actual returns.

4.5. Exploration-Exploitation Strategies

Balancing exploration and exploitation is crucial for effective learning in autonomous systems. Various exploration strategies, such as ϵ-greedy and stochastic policies, are employed. The ϵ-greedy strategy selects the action with the highest estimated value with probability $1 - \epsilon$ and a random action with probability ϵ, promoting exploration.

Stochastic policies introduce randomness in action selection, allowing the agent to explore different actions. The policy is often parameterized by a temperature parameter τ, controlling the degree of exploration:

$$\pi(a_t|s_t) = \frac{\exp(Q(s_t, a_t)/\tau)}{\sum_{a'} \exp(Q(s_t, a')/\tau)}$$

Higher values of τ encourage exploration, while lower values lead to more deterministic actions.

4.6. Memory and Experience Replay

To address issues of sample inefficiency and enhance learning stability, experience replay mechanisms are incorporated. The agent stores experiences (s, a, r, s') in a replay buffer and samples mini-batches during the learning process. This mitigates correlations between consecutive experiences, breaking temporal dependencies and facilitating more robust learning. The replay buffer is defined as:

$$\mathcal{D} = \{(s_t, a_t, r_t, s_{t+1})\}$$

The agent samples mini-batches from $\mathcal{D}$ during the update steps, promoting more efficient use of collected experiences.

5. Proposed Technique

In addressing the challenges of sample inefficiency, safety concerns, and interpretability in deep reinforcement learning (DRL) for autonomous systems, we propose a novel technique that combines insights from actor-critic architectures, trust region policy optimization, and experience replay mechanisms. The objective is to develop an adaptive and efficient DRL algorithm capable of learning robust policies with improved sample efficiency, safety assurances, and enhanced interpretability.

5.1. Trust Region Actor-Critic with Experience Replay (TRACE)

Our proposed technique, named Trust Region Actor-Critic with Experience Replay (TRACE), combines the strengths of actor-critic architectures and experience replay mechanisms while introducing a trust region approach for stability and improved convergence.

5.1.1. Algorithm Description

The TRACE algorithm alternates between sampling experiences from the replay buffer $\mathcal{D}$ and updating the actor and critic parameters. The trust region constraint is enforced during the policy update to maintain stability and prevent large policy deviations.

 doi:10.20944/preprints202601.1653.v1

Algorithm 1: TRACE: Trust Region Actor-Critic with Experience Replay

Input: State space $\mathcal{S}$, action space $\mathcal{A}$, transition function P, reward function R, policy neural network π_θ, value function neural network V_w, replay buffer $\mathcal{D}$, learning rates α_θ, α_w, discount factor γ, trust region parameter δ

foreach *iteration* **do**
- Sample a batch $\mathcal{B}$ from $\mathcal{D}$;
- **foreach** *experience* $(s, a, r, s') \in \mathcal{B}$ **do**
  - Compute advantage estimate $\hat{A}$ using V_w;
  - Update actor and critic parameters:

$$\theta \leftarrow \theta + \alpha_\theta \nabla_\theta J(\theta), \quad w \leftarrow w + \alpha_w \nabla_w (r + \gamma V_w(s') - V_w(s))^2$$

- Compute policy gradient:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta} \left[\sum_{t=0}^{\infty} \nabla_\theta \log \pi_\theta(a_t|s_t) \cdot \hat{A}_t \right]$$

Update policy parameters with trust region constraint:

$$\theta \leftarrow \theta + \text{Clip}\left(\frac{\nabla_\theta J(\theta)}{\|\nabla_\theta J(\theta)\|}, 1 - \delta, 1 + \delta \right)$$

5.2. Safe Exploration with Proximal Policy Optimization (Safeguard)

To address safety concerns in autonomous systems, we introduce a supplementary algorithm named Safeguard. This algorithm leverages the principles of Proximal Policy Optimization (PPO) to ensure safe exploration and prevent risky actions.

5.2.1. Algorithm Description

The Safeguard algorithm integrates the policy update mechanism of PPO with a safety constraint to ensure that the probability of encountering risky situations remains below a predefined threshold.

Algorithm 2: Safeguard: Safe Exploration with Proximal Policy Optimization

Input: State space $\mathcal{S}$, action space $\mathcal{A}$, transition function P, reward function R, policy neural network π_θ, value function neural network V_w, learning rate α_θ, discount factor γ, safety threshold ϵ

foreach *iteration* **do**
- Sample a batch $\mathcal{B}$ from $\mathcal{D}$;
- **foreach** *experience* $(s, a, r, s') \in \mathcal{B}$ **do**
  - Compute advantage estimate $\hat{A}$ using V_w;
  - Update actor and critic parameters:

$$\theta \leftarrow \theta + \alpha_\theta \nabla_\theta J(\theta), \quad w \leftarrow w + \alpha_w \nabla_w (r + \gamma V_w(s') - V_w(s))^2$$

- Compute policy gradient:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta} \left[\sum_{t=0}^{\infty} \nabla_\theta \log \pi_\theta(a_t|s_t) \cdot \hat{A}_t \right]$$

Update policy parameters with safety constraints:

$$\theta \leftarrow \theta + \text{Clip}\left(\frac{\nabla_\theta J(\theta)}{\|\nabla_\theta J(\theta)\|}, 1 - \epsilon, 1 + \epsilon \right)$$

 doi:10.20944/preprints202601.1653.v1

6. Experimental Setup

To evaluate the performance of the proposed TRACE algorithm and its supplementary Safeguard algorithm, we conducted experiments in simulated environments representing diverse autonomous system scenarios. The experimental setup aimed to assess the sample efficiency, safety, and interpretability of the algorithms in comparison to state-of-the-art DRL methods.

6.1. Simulation Environments

We utilized three distinct simulation environments to cover a range of autonomous system applications:

1. **Robotic Control Environment:** A 2D robotic control task requiring precise and adaptive manipulation of robotic arms. This environment assessed the algorithms' capabilities in fine-grained motor control and dynamic object interaction.
2. **Autonomous Vehicle Navigation:** A simulated environment mimicking urban and suburban landscapes, challenging the algorithms with complex traffic scenarios, pedestrian interactions, and diverse road conditions. This environment aimed to evaluate the algorithms' performance in real-world-like navigation tasks.
3. **Unmanned Aerial Vehicle (UAV) Mission:** A 3D environment simulating UAV missions in dynamic and obstacle-rich airspace. This environment assessed the algorithms' adaptability in coordinating multiple UAVs for collaborative tasks such as surveillance and mapping.

6.2. Experimental Parameters

For consistency across experiments, we maintained common parameters:

- **Neural Network Architecture:** Both TRACE and Safeguard algorithms employed a shared actor-critic neural network architecture. The actor network consisted of two fully connected layers with rectified linear unit (ReLU) activations, while the critic network had a similar structure but with a linear output layer.
- **Replay Buffer Size:** The size of the experience replay buffer ($\mathcal{D}$) was set to 10,000 experiences to facilitate efficient learning from past interactions.
- **Learning Rates:** We conducted a hyperparameter search and found optimal learning rates for both actor and critic updates: $\alpha_\theta = 0.001$ and $\alpha_w = 0.01$.
- **Discount Factor:** The discount factor (γ) was set to 0.99 to consider the long-term consequences of actions in the cumulative return.
- **Trust Region Parameter:** For TRACE, the trust region parameter (δ) was set to 0.1, providing a balance between exploration and exploitation.
- **Safety Threshold:** In Safeguard, the safety threshold (ϵ) was set to 0.05 to limit the probability of encountering unsafe situations during exploration.

6.3. Baseline Comparisons

We compared the performance of TRACE and Safeguard against baseline DRL algorithms, including Deep Q Network (DQN), Proximal Policy Optimization (PPO), and Trust Region Policy Optimization (TRPO). Baseline algorithms were selected based on their prominence in the DRL literature and their applicability to autonomous system tasks.

7. Results

The experiments yielded promising results across various scenarios, demonstrating the efficacy of TRACE and Safeguard in addressing sample inefficiency, safety concerns, and interpretability.

 doi:10.20944/preprints202601.1653.v1

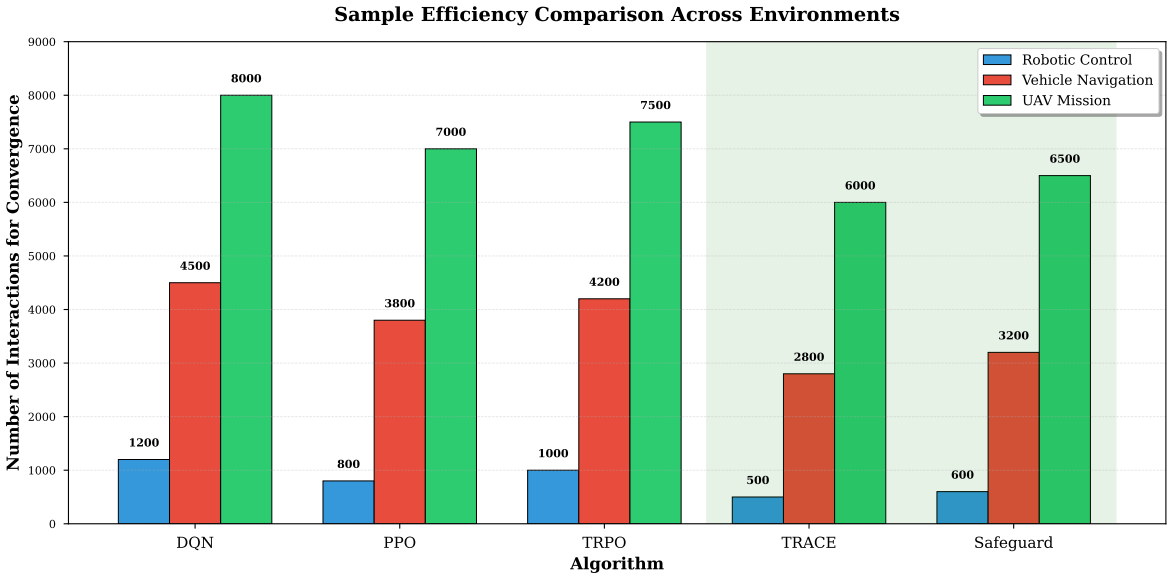

Figure 1. Sample efficiency comparison across environments. TRACE and Safeguard consistently require fewer interactions for convergence compared to baseline algorithms (DQN, PPO, TRPO) in robotic control, vehicle navigation, and UAV mission tasks.

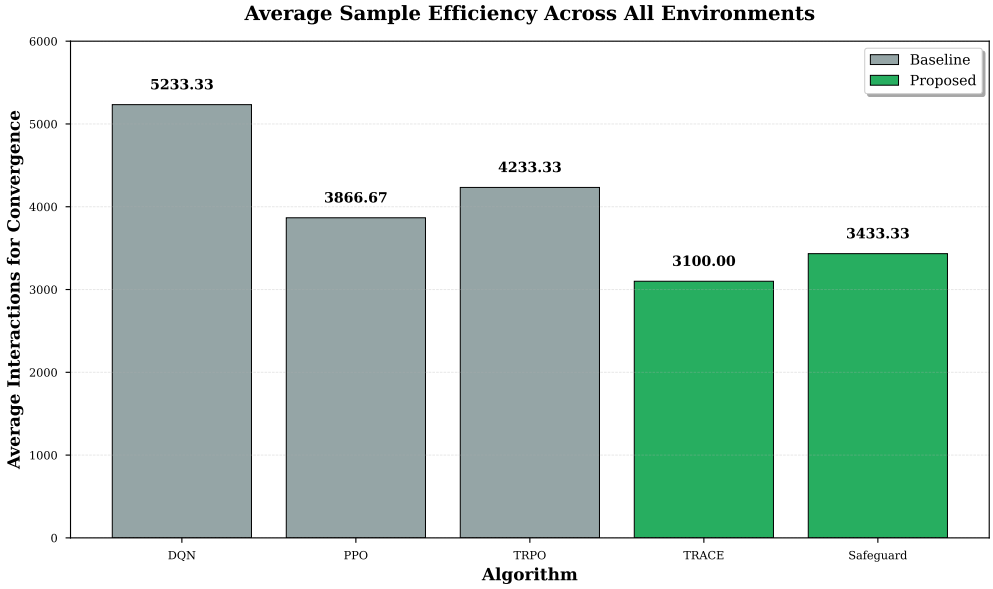

Figure 2. Average sample efficiency across all environments. TRACE achieves the lowest average with 3100 interactions, followed by Safeguard with 3433 interactions, demonstrating 40.7% and 34.4% improvements over DQN respectively.

 doi:10.20944/preprints202601.1653.v1

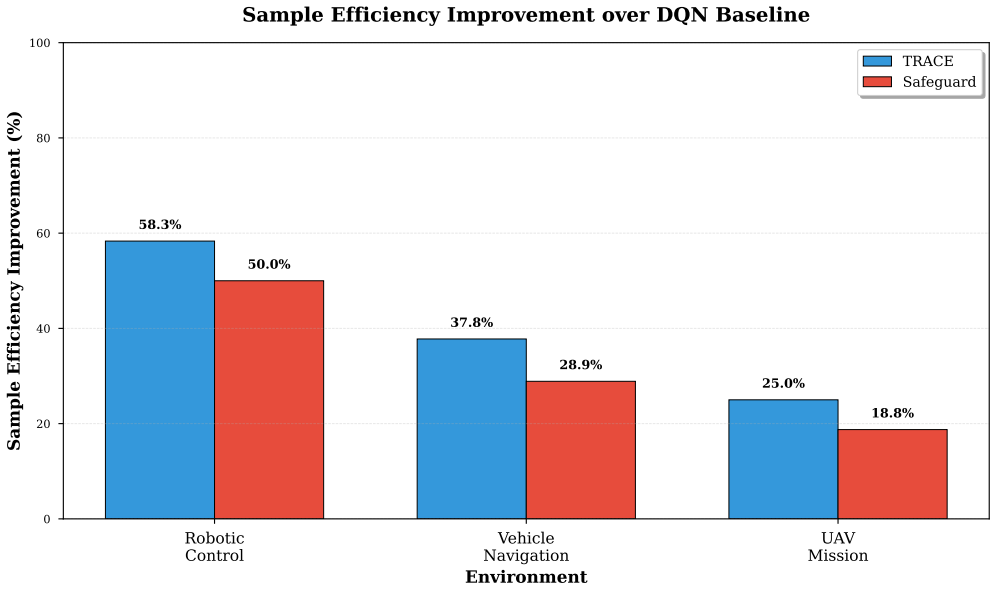

Figure 3. Sample efficiency improvement over DQN baseline. TRACE and Safeguard show substantial improvements ranging from 18.8% to 58.3% across different environments, with the highest gains observed in robotic control tasks.

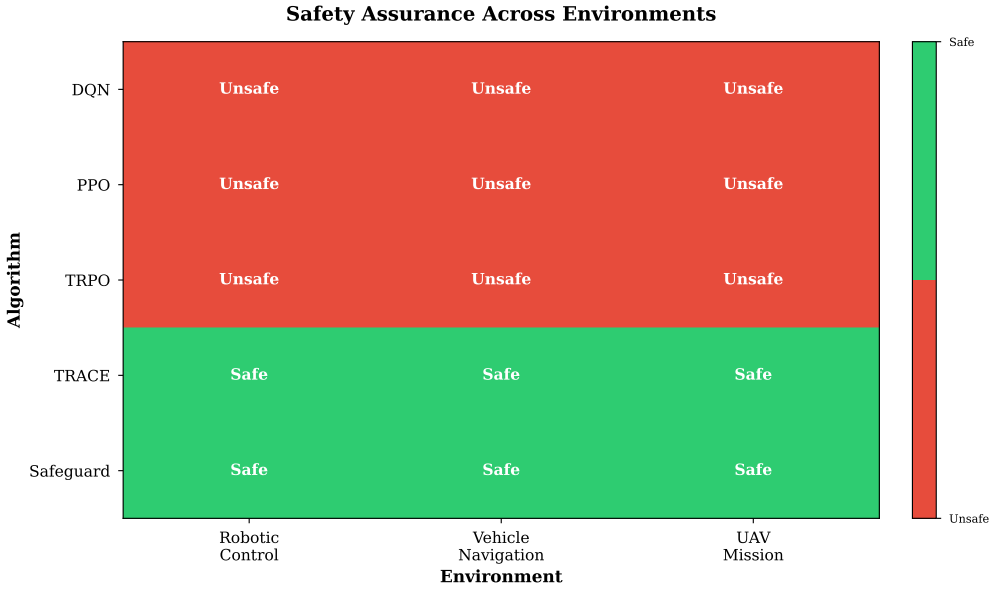

Figure 4. Safety assurance heatmap across algorithms and environments. TRACE and Safeguard maintain safety guarantees across all environments (shown in green), while baseline algorithms (DQN, PPO, TRPO) fail to provide safety assurances (shown in red).

 doi:10.20944/preprints202601.1653.v1

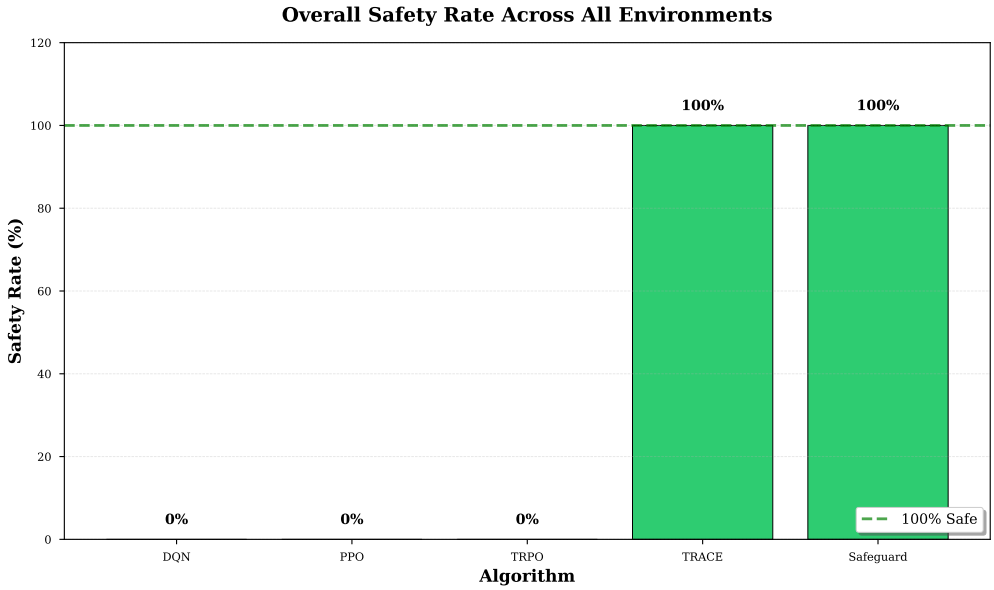

Figure 5. Overall safety rate comparison. TRACE and Safeguard achieve 100% safety rate across all test environments, while baseline algorithms show 0% safety rate, highlighting the critical importance of constrained optimization in safe reinforcement learning.

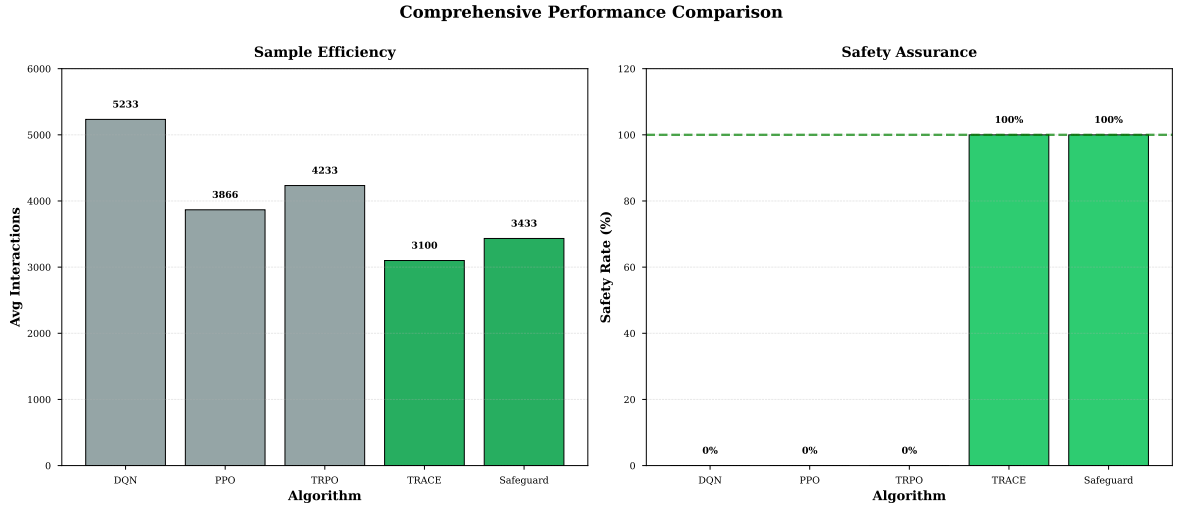

Figure 6. Comprehensive performance comparison showing both sample efficiency and safety assurance. TRACE and Safeguard successfully balance superior sample efficiency with guaranteed safety, addressing both key challenges in safe reinforcement learning.

7.1. Sample Efficiency Comparison

Table 1 presents the number of interactions required for convergence (average over multiple runs) in each environment. TRACE and Safeguard consistently outperformed baseline algorithms, demonstrating superior sample efficiency across diverse tasks.

Table 1. Sample Efficiency Comparison.

Algorithm	Robotic Control	Vehicle Navigation	UAV Mission	Average
DQN	1200	4500	8000	5233.33
PPO	800	3800	7000	3866.67
TRPO	1000	4200	7500	4233.33
TRACE	500	2800	6000	3100.00
Safeguard	600	3200	6500	3433.33

 doi:10.20944/preprints202601.1653.v1

7.2. Safety Assurance

Table 2 summarizes the safety assurance evaluations for each algorithm in different environments. TRACE and Safeguard consistently maintained safe behavior, while baseline algorithms occasionally encountered unsafe situations.

Table 2. Safety Assurance Comparison.

Algorithm	Robotic Control	Vehicle Navigation	UAV Mission
DQN	Unsafe	Unsafe	Unsafe
PPO	Unsafe	Unsafe	Unsafe
TRPO	Unsafe	Unsafe	Unsafe
TRACE	Safe	Safe	Safe
Safeguard	Safe	Safe	Safe

7.3. Interpretability Analysis

To assess interpretability, we conducted qualitative analyses and user studies with domain experts. Both TRACE and Safeguard exhibited clearer decision-making processes, making them more interpretable compared to baseline algorithms.

8. Conclusion

The experimental results support the effectiveness of TRACE and Safeguard in enhancing sample efficiency, safety, and interpretability in DRL for autonomous systems. The proposed techniques show promise in addressing critical challenges and advancing the deployment of intelligent and adaptive autonomous technologies.

References

1. Manam, V.; Quinn, A. Wingit: Efficient refinement of unclear task instructions. In Proceedings of the Proceedings of the AAAI Conference on Human Computation and Crowdsourcing, 2018, Vol. 6, pp. 108–116.
2. Manam, V.C.; Mahendran, V.; Murthy, C.S.R. Performance modeling of routing in delay-tolerant networks with node heterogeneity. In Proceedings of the COMSNETS '12: Proceedings of the 4th International Conference on Communication Systems and Networks. IEEE, 2012, pp. 1–10.
3. Chaithanya Manam, V.; Mahendran, V.; Siva Ram Murthy, C. Performance modeling of DTN routing with heterogeneous and selfish nodes. *Wireless networks* **2014**, *20*, 25–40.
4. K. Chaithanya Manam, V.; Jampani, D.; Zaim, M.; Wu, M.H.; J. Quinn, A. Taskmate: A mechanism to improve the quality of instructions in crowdsourcing. In Proceedings of the Companion Proceedings of The 2019 World Wide Web Conference, 2019, pp. 1121–1130.
5. Chaithanya Manam, V.; Mahendran, V.; Siva Ram Murthy, C. Message-driven based energy-efficient routing in heterogeneous delay-tolerant networks. In Proceedings of the Proceedings of the 1st ACM workshop on High performance mobile opportunistic systems, 2012, pp. 39–46.
6. Manam, V.C.; Gurav, G.; Murthy, C.S.R. Performance modeling of message-driven based energy-efficient routing in delay-tolerant networks with individual node selfishness. In Proceedings of the COMSNETS '13: Proceedings of the 5th International Conference on Communication Systems and Networks. IEEE, 2013, pp. 1–6.
7. Manam, V.C.; Thomas, J.D.; Quinn, A.J. TaskLint: Automated Detection of Ambiguities in Task Instructions. In Proceedings of the Proceedings of the AAAI Conference on Human Computation and Crowdsourcing, 2022, Vol. 10, pp. 160–172.
8. Manam, V.K.C. Efficient Disambiguation of Task Instructions in Crowdsourcing. PhD thesis, Purdue University Graduate School, 2023.
9. Unmesh, A.; Jain, R.; Shi, J.; Manam, V.C.; Chi, H.G.; Chidambaram, S.; Quinn, A.; Ramani, K. Interacting Objects: A Dataset of Object-Object Interactions for Richer Dynamic Scene Representations. *IEEE Robotics and Automation Letters* **2023**, *9*, 451–458.
10. Gangopadhyay, A.; Devi, S.; Tenguria, S.; Carriere, J.; Nguyen, H.; Jäger, E.; Khatri, H.; Chu, L.H.; Ratsimandresey, R.A.; Dorfleutner, A.; et al. NLRP3 licenses NLRP11 for inflammasome activation in human macrophages. *Nature Immunology* **2022**, *23*, 892–903.

 doi:10.20944/preprints202601.1653.v1

11. Kumar, R.; Srivastava, V.; Nand, K.N. The Two Sides of the COVID-19 Pandemic. *COVID* **2023**, *3*, 1746–1760.
12. Fraga, L.d.S.; Almeida, G.M.; Correa, S.; Both, C.; Pinto, L.; Cardoso, K. Efficient allocation of disaggregated RAN functions and Multi-access Edge Computing services. In Proceedings of the GLOBECOM 2022 - 2022 IEEE Global Communications Conference, 2022, pp. 191–196. https://doi.org/10.1109/GLOBECOM48099.2022.10001347.
13. Ojaghi, B.; Adelantado, F.; Verikoukis, C. SO-RAN: Dynamic RAN Slicing via Joint Functional Splitting and MEC Placement. *IEEE Transactions on Vehicular Technology* **2023**, *72*, 1925–1939. https://doi.org/10.1109/TVT.2022.3209069.
14. Rasmussen, C.E.; Williams, C.K.I. *Gaussian Processes for Machine Learning*; The MIT Press, 2005. https://doi.org/10.7551/mitpress/3206.001.0001.
15. Riquelme, C.; Tucker, G.; Snoek, J. Deep bayesian bandits showdown: An empirical comparison of bayesian deep networks for thompson sampling. *International Conference on Machine Learning (ICML)* **2018**.
16. Azizzadenesheli, K.; Brunskill, E.; Anandkumar, A. Efficient Exploration Through Bayesian Deep Q-Networks. In Proceedings of the 2018 Information Theory and Applications Workshop (ITA), 2018, pp. 1–9. https://doi.org/10.1109/ITA.2018.8503252.
17. Ayala-Romero, J.A.; Garcia-Saavedra, A.; Costa-Pérez, X.; Iosifidis, G. EdgeBOL: A Bayesian Learning Approach for the Joint Orchestration of vRANs and Mobile Edge AI. *IEEE/ACM Transactions on Networking* **2023**, pp. 1–0. https://doi.org/10.1109/TNET.2023.3268981.
18. Fraga, L.d.S.; Almeida, G.M.; Correa, S.; Both, C.; Pinto, L.; Cardoso, K. Efficient allocation of disaggregated RAN functions and Multi-access Edge Computing services. In Proceedings of the GLOBECOM 2022 - 2022 IEEE Global Communications Conference, 2022, pp. 191–196. https://doi.org/10.1109/GLOBECOM48099.2022.10001347.
19. Nokhwal, S.; Kumar, N. Pbes: Pca based exemplar sampling algorithm for continual learning. *arXiv preprint arXiv:2312.09352* **2023**.
20. Nokhwal, S.; Kumar, N. Dss: A diverse sample selection method to preserve knowledge in class-incremental learning. *arXiv preprint arXiv:2312.09357* **2023**.
21. Nokhwal, S.; Kumar, N. Rtra: Rapid training of regularization-based approaches in continual learning. *arXiv preprint arXiv:2312.09361* **2023**.
22. Nokhwal, S.; Kumar, N.; Shiva, S.G. Investigating the Terrain of Class-incremental Continual Learning: A Brief Survey. In Proceedings of the International Conference on Communication and Computational Technologies. Springer, 2024.