

Anomaly Detection in UR5 Robots Using Reinforcement Learning for Smart Manufacturing

Messaoudi Salma¹, Kenneth Duran^{2,3}, Esequiel Garcia^{2,3}, Md Shahriar Forhad^{2,3}, Ahmed Bendaouia^{2,3}, El Hassan Abdelwahed¹ and Jianzhi Li^{2,3}

¹ Cadi Ayyad University, UCA, Faculty of Sciences Semlalia, Computer Systems Engineering Laboratory (LISI), Marrakech, Morocco.

² Institute for Advanced Manufacturing (IAM), University of Texas Rio Grande Valley, USA.

³ Manufacturing and Industrial Engineering Department, University of Texas Rio Grande Valley, USA.

1. Abstract

Traditional anomaly detection methods in industrial robotics often rely on supervised learning, which struggles with dynamic environments and requires extensive labeled data. In this study, we propose a reinforcement learning (RL) framework for real-time, unsupervised anomaly detection in a UR5 robotic arm. Using high-dimensional data for different payloads and operational conditions, we train four separate Deep Q-Network (DQN) agents, each dedicated to a specific anomaly type: general, position, velocity, and current. Unlike Q-learning, which is ineffective with large state spaces, DQN leverages deep neural networks to model complex robot behavior. Our experiments show promising results, achieving up to 93.2% accuracy, with the best performance observed in position anomaly detection (F1-score of 0.89). This study highlights the scalability and adaptability of RL-based models for autonomous robot monitoring, and paves the way for future integration with temporal-aware architectures and real-time Digital Twin environments.

Keywords: Reinforcement Learning, Deep Q-Network, Q-Learning, Anomaly Detection, Autonomous Manufacturing.

2. Introduction

Artificial Intelligence (AI) has significantly transformed the field of robotics by enabling systems to learn, adapt, and make decisions dynamically in complex and changing environments. Traditional robotic control approaches, including rule-based logic and supervised learning models, often require large volumes of labeled data and predefined instructions. While effective in controlled scenarios, these methods can lack the adaptability and robustness required in real-time industrial applications, where robots are expected to operate under diverse and unpredictable conditions. Reinforcement Learning has emerged as a powerful alternative for intelligent decision-making in robotics. Unlike supervised learning, RL does not rely on labeled datasets; instead, it learns optimal behaviors through trial-and-error interactions with the environment by maximizing cumulative rewards. This characteristic

makes RL particularly suited for dynamic anomaly detection tasks, where new or unforeseen system states may arise during operation. In our previous work [1], we investigated anomaly detection in the UR5 robotic arm using deep learning and classical machine learning models. Although the approach achieved promising results, it relied on externally generated labels and offline feature engineering, limiting its scalability and adaptability to unseen conditions.

In this study, we propose a Reinforcement Learning-based approach for anomaly detection in a six-joint industrial robotic arm (UR5) using real-world data collected from the National Institute of Standards and Technology (NIST [2]). Unlike traditional methods that rely on a single model for general fault detection, we design four separate RL agents, each trained to detect anomalies in a specific Robot Performance Metric (RPM):

- **General anomaly detection (y):** Identifies any type of abnormal behavior.
- **Position anomaly detection (y_pos):** Detects deviations in the robot's spatial positioning.
- **Velocity anomaly detection (y_velocity):** Identifies irregularities in joint or end-effector speeds.
- **Current anomaly detection (y_current):** Detects unusual electrical current consumption patterns.

The primary objective of this research is to demonstrate the applicability and effectiveness of RL for real-time anomaly detection in industrial robotics. Our experimental evaluation focuses on classification performance across the defined metrics using accuracy, precision, recall, and F1-score. Ultimately, this work highlights the advantages of RL in developing intelligent, adaptive, and data-efficient monitoring systems for Digital Twin-based robot performance analysis.

3. State of the Art

Anomaly detection in autonomous robotic systems is a critical challenge in Industry 4.0, where systems must remain reliable, adaptive, and self-aware under evolving operational conditions. Traditional rule-based monitoring approaches struggle to generalize in such dynamic environments. In contrast, reinforcement learning has emerged as a powerful data-driven alternative, offering the capacity to learn optimal behaviors directly from interactions, without requiring labeled datasets or predefined rules.

While numerous studies have explored the use of RL and digital twins in robotics, their primary focus often lies in task automation, motion control, or collaborative behavior optimization. For example, [3] employed a SARSA-based agent combined with augmented reality to enhance human-robot task sharing. Similarly, [4] trained RL agents within simulated digital twin environments to improve sim-to-real transfer for robotic manipulation. Extending this concept to grasping, [5] leveraged synthetic data and 3D perception to train grasping agents entirely in simulation before deploying them on physical UR3 robots. While these works demonstrate the benefits of RL in improving autonomy and reducing training costs, they do not directly tackle fault or anomaly detection. A growing body of research, however, suggests that RL may also support intelligent perception and inspection in uncertain settings. [6], for instance, demonstrated the viability of RL for adaptive view planning in remanufacturing environments. Using a Soft Actor-Critic (SAC) agent constrained by robot kinematics, they achieved over

90% surface coverage with fewer than nine scans on average. By integrating ROS and MoveIt for execution, their system bridges the sim-to-real gap offering a promising foundation for RL-driven anomaly inspection in dynamic industrial conditions. In a different domain, [7] applied a deep RL framework to mobile robot navigation in tight, human-populated environments. By modeling wall proximity, obstacle dynamics, and human movement using a multi-layer neural network, they significantly reduced collisions and “freezing” behaviors that demonstrate how RL can enhance robust perception-based decision-making. Other hybrid approaches such as residual reinforcement learning, introduced by [8], combine hand-crafted control policies with learned residuals to improve performance in the presence of modeling errors, contact dynamics, or actuator noise. These studies collectively highlight RL’s ability to adapt under uncertainty, even though their emphasis remains on task performance rather than internal system diagnostics. Similarly, [9] integrated a deep learning-enhanced digital twin to support human safety in collaborative settings, while [10] developed a multi-agent transfer RL approach for dynamic task allocation in changing production environments showing the adaptability of RL across hierarchical industrial decision-making levels. More recently, large language models (LLMs) have been explored as generative agents to automate robotic control design. [11] demonstrated that GPT-4 can autonomously generate locomotion trajectories, solve inverse kinematics, and design RL reward functions for humanoid robots, achieving near-human performance in simulation. These results align with a broader trend to integrate generative AI into robotic pipelines, further reducing the need for manual engineering and enabling adaptive behavior in complex scenarios.

Beyond reinforcement learning, several recent works have addressed anomaly detection using supervised or unsupervised machine learning. In robotic welding, [12] combined the Local Outlier Factor (LOF) with a neural network to remove sensor noise and reconstruct missing data under varying lighting conditions. In parallel, [13] proposed an adaptive grasping system powered by CNNs and optimization algorithms, augmented with transfer learning to bridge the sim-to-real gap. Their system achieves 96% success in real grasping scenarios without manual tuning, illustrating how data-efficient learning can support robust robotic execution. In the realm of perception reliability, [14] introduced a Bayesian anomaly detection and recovery architecture for mobile robots, capable of filtering sensor noise and maintaining system stability in uncertain environments. For joint-level diagnostics, [15] proposed a wavelet-based monitoring approach using decision trees to detect non-stationary joint anomalies based on multi-sensor fusion. Similarly, [16] designed a control architecture for 4DOF robotic arms that integrates neural networks, digital twins, and model predictive control to maintain trajectory accuracy under variable conditions. Localization performance has also been enhanced via ML techniques. [17] fused visual ground-truth data with odometry using XGBoost, significantly reducing positioning error while maintaining real-time feasibility. On robotic assembly lines, [18] employed time-series classification from sensor data for anomaly detection and fault classification in closed-loop industrial settings. Meanwhile, [19] proposed an Accelerated Gradient Descent SVM, enhanced with Gaussian filtering, to detect faults in multi-joint robotic arms using only attitude sensor data achieving 98% accuracy without supervision. Broader reviews, such as that by [20], have outlined the evolution of Fault Detection and Isolation (FDI) from rule-based strategies to deep models like CNNs, RNNs, and autoencoders. They emphasized persistent challenges in data availability, interpretability, and the need for integration with edge computing for real-time deployment. Complementing these efforts, [21] surveyed the integration of AI into collaborative robotics, highlighting features like voice-based control, dynamic collision avoidance, and multimodal sensing as key enablers of human-centric and safe automation under the Industry 5.0 paradigm.

Despite the richness of these contributions, most approaches, whether based on RL or conventional ML, prioritize external performance metrics such as motion efficiency, task completion, or sensor fidelity. They often overlook the internal health monitoring of the robotic system itself. Our work addresses this gap by framing anomaly detection as a reinforcement learning problem, where a Deep Q-Network is trained to classify robot states (normal vs. anomalous) under varying operational conditions such as load, speed, and temperature. Unlike prior work, our method does not rely on labeled datasets or expert supervision but learns directly from raw state transitions in a continuous environment. We further propose a robust preprocessing pipeline with engineered features and standardized evaluation metrics (accuracy, recall, F1-score) to rigorously validate detection performance. This reframes reinforcement learning not just as a tool for action selection, but as a foundation for intelligent, adaptive fault awareness in next-generation robotic systems.

4. Methodology

4.1. Dataset Description and Preprocessing

Understanding the structure and characteristics of the dataset is a critical step before applying any machine learning or reinforcement learning technique. In this section, we present a detailed description of the data used in this study, including its origin, composition, and relevance to the problem of anomaly detection in robotic systems. We also explain the preprocessing steps applied to ensure the dataset's suitability for sequential modeling. This preparation is essential to guarantee that the models can effectively capture the temporal dependencies and complex patterns associated with the UR5 robotic arm's behavior under various operating conditions.

4.1.1. Dataset Overview

The dataset used in this study is sourced from the National Institute of Standards and Technology, which provides a comprehensive set of data that includes six distinct subsets. Each subset represents different experimental conditions involving varying payloads, robot speeds, and temperatures, which significantly influence the performance of the UR5 robotic arm. This diverse data allows for a robust exploration of the UR5's performance in real-world conditions, providing a valuable foundation for the application of RL to detect anomalies under different scenarios. Each of these six datasets follows a consistent structure and contains a total of 92 features, which are categorized into three main groups: system data, operational data, and engineered features, please refer to the **Table 1** for the detailed feature breakdown. These features capture various aspects of the UR5 robot's behavior, including the individual joints, forces acting on the robot, and temporal dynamics. The dataset is structured to ensure sequential time dependencies, making it particularly suitable for models that leverage temporal information, such as RL algorithms.

Table 1 : Categorization of the 92 features from the NIST dataset used in UR5 anomaly detection.

Feature Type	Number of Features	Description	Examples
7D System Data	60	Features related to each joint of the UR5 robot, capturing	Joint velocities, Joint currents, Joint

		velocities, currents, positions, and accelerations.	positions, Joint accelerations...
Operational Data	13	Features describing the overall robot state, including Cartesian position, rotational orientation, forces, and timestamps.	Robot_time, Cartesian coordinates (x, y, z, rx, ry, rz), Robot TCP Force
Engineered Features	19	Features derived from raw data through feature engineering, such as differences in positions, velocities, currents, and time.	Δ Position, Δ Velocity, Δ Current, Δ Robot Time

Each of the six datasets is indexed by a sequential Robot_time, which ensures that time-dependent models, can learn from the temporal progression of events. This feature allows the models to effectively capture and predict patterns in the robot’s behavior, accounting for both immediate and cumulative changes in performance.

4.1.2. Preprocessing Steps

Prior to applying reinforcement learning techniques, several preprocessing steps are performed to ensure the quality and suitability of the data for sequential modeling. The entire pipeline, from data preprocessing and feature engineering to model training and process adjustment, is illustrated in the **Figure 1** below:

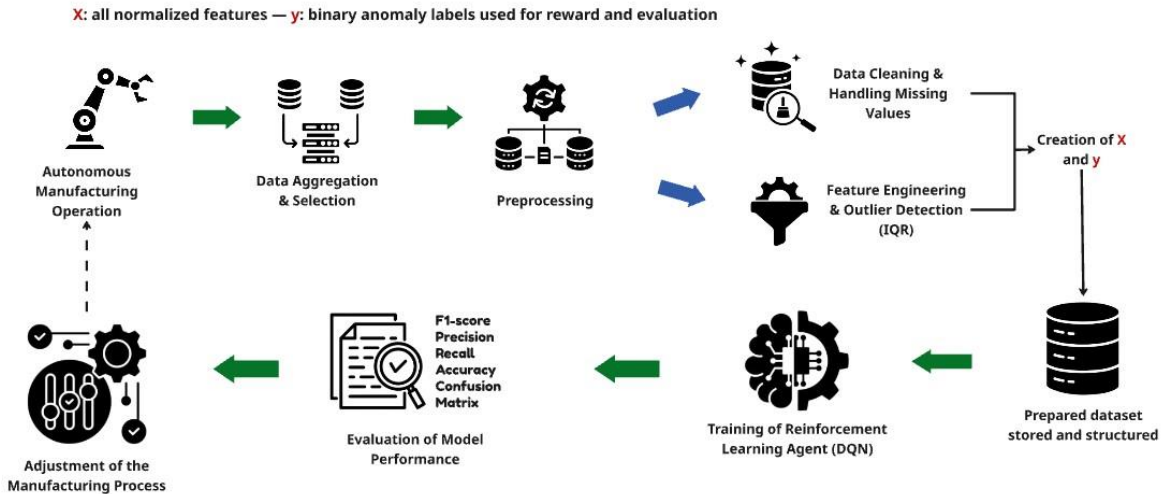


Figure 1 : Reinforcement Learning-Based Anomaly Detection Pipeline

- **Data Cleaning and Handling of Missing Values:** The dataset is thoroughly inspected for missing, null, or inconsistent entries. In cases where missing values are identified, they are imputed using the mean of the corresponding feature, preserving the statistical distribution without introducing bias.
- **Feature Standardization:** Given the heterogeneous nature of the features, ranging from forces and positions to velocities and torques, it is essential to bring them onto a comparable scale. A min-max normalization technique, specifically the

MinMaxScaler, is applied to transform all numerical features into a fixed range, typically [0, 1]. This ensures that no single feature dominates the learning process due to scale differences, while preserving the relative relationships between values.

- **Dataset Splitting:** To enable robust training and evaluation, each of the six datasets is partitioned into training (80%) and testing (20%) subsets. The splitting is performed chronologically to maintain the temporal integrity of the sequences, which is critical for time-series analysis and reinforcement learning tasks.
- **Temporal Structuring:** The sequential nature of the data is preserved throughout preprocessing. Each record is indexed by a Robot_time variable, ensuring that the temporal dependencies between observations are maintained. This structuring is vital for capturing time-dependent patterns and for training models that rely on sequence context, such as Long Short-Term Memory (LSTM) or reinforcement learning agents.

Feature engineering plays a pivotal role in enhancing the performance of machine learning and reinforcement learning models by enabling the extraction of relevant patterns from raw data. In this study, we construct 19 additional features that complement the original dataset, aiming to enrich the representation of robot behavior and improve anomaly detection capabilities. These engineered features are grouped into two main categories: difference-based features (Table 2) and time-derived features (Table 3).

Difference-Based Features: To better capture deviations between the robot's actual behavior and its expected or target behavior, we compute a set of difference-based features. These features are extracted for each of the six joints of the UR5 robot and include differences in position, velocity, and current consumption. This results in a total of $6 \times 3 = 18$ engineered features.

Time-Derived Feature: In addition to spatial and dynamic deviations, temporal irregularities are also considered. A time-derived feature is introduced to capture possible delays or anomalies in the data acquisition process. This feature measures the elapsed time between two consecutive records in the dataset.

Table 2 : Difference-based features engineered for each of the 6 joints of the UR5 robot (total of 18 features).

Feature Name	Description
Δ Position	Measures the deviation between the actual joint position and the target position.
Δ Velocity	Captures the discrepancy between the actual and target joint velocities.
Δ Current	Identifies variations between the actual and expected electrical current drawn by the joints.

Table 3 : Time-derived feature to represent temporal gaps between sequential data points.

Feature Name	Description	Computation
Δ Robot Time	Captures the time difference between two consecutive timestamps, indicating potential delays in system response.	$\text{Robot_Time}_t - \text{Robot_Time}_{t-1}$

4.2. Reinforcement Learning

Reinforcement Learning is a machine learning paradigm where an agent learns to make decisions by interacting with an environment. Instead of learning from labeled data (as in supervised learning), the agent receives rewards or penalties based on its actions, optimizing its strategy over time, the general reinforcement learning loop is illustrated in the **figure 2**.

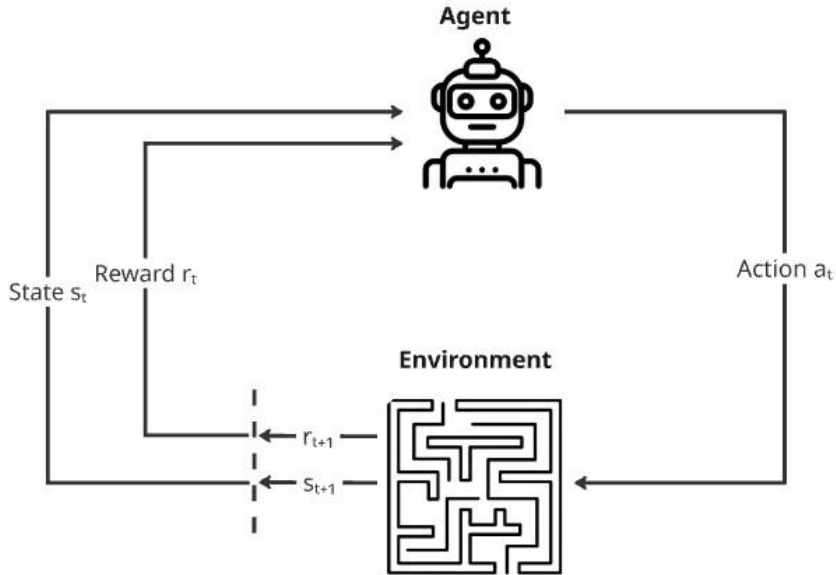


Figure 2 : General Reinforcement Learning Loop

4.2.1. Key Components of RL

In the anomaly detection framework, Reinforcement Learning plays a central role by enabling an agent to learn optimal classification strategies through interaction with the environment. The agent continuously updates its policy based on rewards received from its decisions, allowing it to adapt and improve over time without relying solely on predefined labels. The core components of the RL formulation in this study are defined in **Table 4**.

Table 4 : Reinforcement Learning components as defined in the UR5 anomaly detection framework.

Component	Definition in This Project
Agent	The RL algorithm responsible for learning and identifying anomalous patterns within the robot's operational data.
Environment	The UR5 robot dataset, encompassing various system states captured under different payload, speed, and temperature conditions.
State (S)	A high-dimensional feature vector describing the robot's condition at a specific time step, including raw and engineered features.
Action (A)	A binary decision made by the agent: classify the current state as normal (0) or anomalous (1).
Reward (R)	The feedback signal guiding learning: +1 for a correct classification, -1 for an incorrect classification.

This dynamic setup allows the RL agent to refine its anomaly detection policy iteratively, making it particularly effective in environments where patterns evolve over time. Unlike static supervised models, RL-based systems can better capture and respond to changing behavioral patterns in real-world robotic systems.

4.2.2. Q-Learning vs. Deep Q-Network

Q-Learning:

Q-Learning is a fundamental model-free reinforcement learning algorithm that learns the optimal action-selection policy by iteratively updating a Q-table using the Bellman equation (Eq. 1):

$$Q(s, a) \leftarrow Q(s, a) + \alpha [r + \gamma \max_{a'} Q(s', a') - Q(s, a)] \quad \text{Eq. 1}$$

Where:

- $Q(s, a)$ is the estimated value (expected future reward) of taking action a in state s .
- α is the learning rate, controlling how much new information overrides old.
- r is the immediate reward received after performing action a .
- γ is the discount factor, determining the importance of future rewards.
- $\max_{a'} Q(s', a')$ is the maximum expected future reward for the next state s' over all possible actions a' .

Q-Learning is effective for small, discrete state spaces, but it becomes impractical when handling large, high-dimensional problems like our 92-feature dataset.

Deep Q-Network:

To overcome the limitations of Q-Learning, Deep Q-Networks replace the Q-table with a deep neural network (DNN) that approximates the Q-values. The network takes a state vector as input and outputs Q-values for each action. Instead of explicitly storing all Q-values, DQN learns through backpropagation and gradient descent, making it scalable to complex environments. A key improvement in DQN is the use of:

- Experience Replay: Stores past experiences and randomly samples them to break correlations in training data.
- Target Network: Maintains a separate, slowly updated network to stabilize learning and prevent divergence.

Key Differences Between Q-Learning and DQN:

To select an appropriate reinforcement learning algorithm for our anomaly detection task, it is important to compare traditional Q-Learning with Deep Q-Networks. Given the high-dimensional nature of our input data (92 features), this comparison helps justify the use of DQNs over simpler tabular methods. **Table 5** presents a summary of key differences between Q-Learning and DQN in terms of state space handling, memory requirements, training strategies, and their suitability for high-dimensional feature spaces.

Table 5 : Comparison between Q-Learning and DQN for high-dimensional anomaly detection tasks.

Feature	Q-Learning	DQN
State Space Size	Small, discrete	Large, continuous
Memory Requirements	Stores a Q-table	Uses a deep neural network
Training Method	Bellman equation updates	Backpropagation with gradient descent
Stability	Unstable in large dimensions	More stable with experience replay and target network
Suitability for this project	Not suitable for 92 features	Efficient for high-dimensional feature spaces

4.2.3. Reinforcement Learning Training Pipeline

In this study, we adopt a DQN approach for anomaly detection, structured around a five-step learning pipeline. First, in the **data preprocessing** stage, we select all 92 available features, comprising operational, 7D system, and engineered data, followed by normalization to ensure stable training dynamics. Any missing values are handled by imputing the mean of the corresponding feature. Second, the **state representation** is defined as a 92-dimensional feature vector representing the robot’s operational condition at a given time step t . This compact yet informative representation allows the agent to learn from the robot's real-time condition. Third, the **action space** is binary: the agent chooses between classifying an input as normal (action 0) or anomalous (action 1), simplifying the decision-making process. Fourth, the **reward function** is designed to reinforce correct classification. The agent receives a reward of +1 for a correct decision and -1 for an incorrect one, encouraging continuous learning from its environment. Finally, during the **training phase**, the DQN model is initialized with random weights. At each step, the agent observes the current state s_t , selects an action a_t using an ϵ -greedy strategy, receives a reward r_t , and observes the next state s_{t+1} . These experiences (s, a, r, s') are stored in a replay buffer. A mini-batch is randomly sampled from this buffer to update the Q-network using the loss derived from the Bellman equation (Eq. 2):

$$Q(s, a) = r + \gamma \max_{a'} Q_{target}(s', a') \quad \text{Eq. 2}$$

The DQN process for anomaly detection in this study is illustrated in the **figure 3**. The target network is periodically updated to stabilize learning.

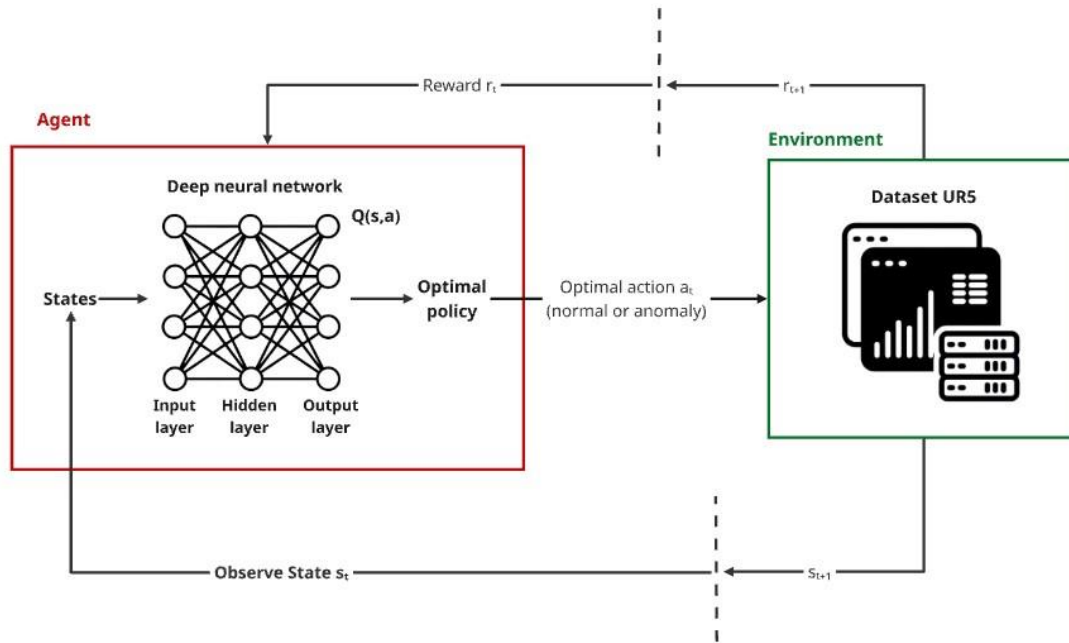


Figure 3 : Architecture of Deep Q-Network used for anomaly detection

5. Experimental results & Discussion

5.1. Quantitative Results

This section presents the performance of the Deep Q-Network model applied to the detection of anomalies on four different targets: the general anomaly, position anomaly, velocity anomaly, and current anomaly. The goal is to assess how well the model learns to identify such anomalies based on robotic time-series data by leveraging the capabilities of deep reinforcement learning. We evaluate the model using common binary classification metrics: test accuracy, precision, recall, and F1-score. These metrics are computed on the test set and are reported separately for each class (normal and anomalous). Given the class imbalance in the anomaly datasets, we also report macro-averaged scores, which treat all classes equally and provide a more balanced assessment of the model's performance, particularly in detecting rare anomalies. The following **table 6** summarizes the performance for each anomaly target. The values shown correspond to macro-averaged metrics, ensuring a balanced representation of both classes.

Table 6 : Performance comparison of DQN model across different anomaly detection targets using macro-averaged metrics.

Target	Accuracy	Precision (macro)	Recall (macro)	F1-score (macro)
y	0.863	0.86	0.86	0.86
y_pos	0.906	0.88	0.90	0.89
y_velocity	0.88	0.86	0.83	0.84
y_current	0.932	0.77	0.75	0.76

- **General Anomaly (y):** For the general anomaly target, the model achieves an accuracy of 86.3% with balanced precision and recall scores of 0.86. This demonstrates the model's capacity to distinguish between normal and abnormal states across all types of behavior, confirming that deep reinforcement learning can generalize well even when not trained to detect specific anomaly categories.
- **Position Anomaly (y_pos):** The best results are observed on the y_pos target, where the model reaches a high accuracy of 90.6% and a macro F1-score of 0.89. This indicates excellent detection capability for anomalies in the robot's position. The high recall (0.90) suggests that the model is particularly effective at identifying actual anomalous behaviors in this category. This strong performance may stem from the fact that position-related anomalies often result in clear, sustained deviations in robot trajectories, which are more easily detected through temporal patterns learned during training.
- **Velocity Anomaly (y_velocity):** For velocity anomalies, the DQN model achieves a test accuracy of 88.1%, with a macro-averaged F1-score of 0.84. The performance reveals a slight imbalance in class-wise recall: while the model correctly identifies normal instances with a high recall of 0.94, its recall for anomalous cases drops to 0.72. This indicates that some velocity-related anomalies are still missed during prediction. Such behavior may stem from the complex, time-varying nature of velocity signals in robotic movements, where transient anomalies are harder to detect. However, the precision for anomalous cases remains relatively strong at 0.82, suggesting that when the model does predict an anomaly, it is often correct. The overall macro precision and recall values (0.86 and 0.83, respectively) confirm the model's reliable, though not perfect, ability to detect velocity-related faults.
- **Current Anomaly (y_current):** The y_current target yields the highest overall accuracy (93.2%), yet this result is misleading. The precision and recall for the anomalous class are significantly lower (0.59 and 0.53, respectively), resulting in a macro F1-score of only 0.76. This indicates a bias toward predicting the normal class and difficulty in detecting current-related anomalies. This challenge might be due to the subtle and short-lived nature of current fluctuations or an insufficient number of anomaly examples in the dataset. Such findings suggest the need for either data augmentation, improved feature representation, or model refinement to better capture current-related behaviors.

5.2. Learning Behavior and Loss Analysis

To better understand the performance and training dynamics of the DQN model for velocity anomaly detection, we present and analyze the key training curves and evaluation metrics obtained under the first experimental condition (payload = 1.6 kg, half speed, and normal temperature).

This figure (**figure 4**) illustrates the evolution of validation accuracy for the velocity anomaly detection task over 10,000 timesteps. The curve shows a sharp increase in performance during the early training phase, followed by stable accuracy fluctuations around 86–88%, indicating consistent learning and convergence of the DQN policy.

The plot in **figure 5** displays the temporal difference (TD) loss during training. The loss drops rapidly from 0.38 to a lower range between 0.15 and 0.2. The absence of large oscillations

or divergence in the loss curve suggests that the learning process is stable and that the model effectively minimizes its error signal across episodes.

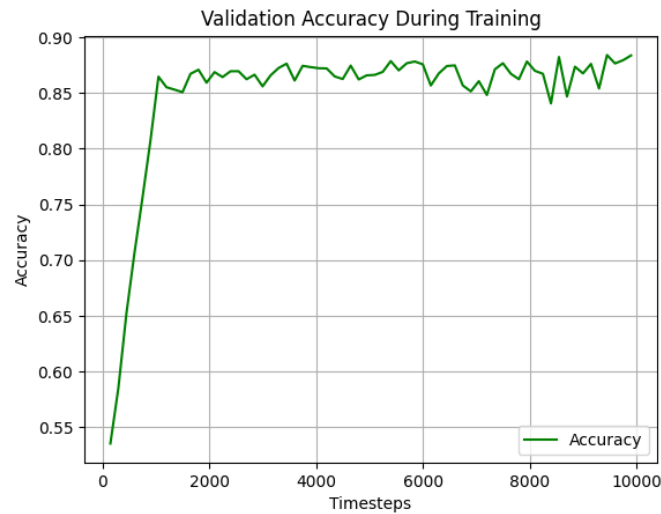


Figure 4 : Validation accuracy during training for velocity anomaly detection using DQN under the first configuration

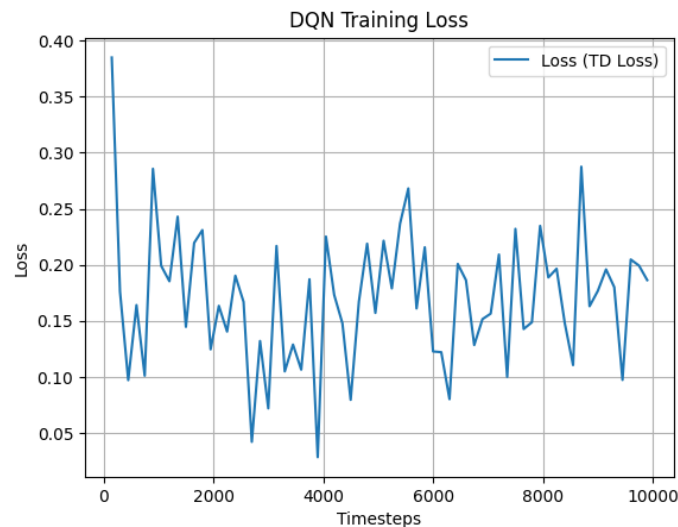


Figure 5 : Training loss curve (TD Loss) of the DQN model for velocity anomaly detection under the first configuration

The confusion matrix in **figure 6** presents the detailed classification performance on the test set. The model correctly classifies 3553 normal instances and 990 anomalous ones. However, it misclassifies 394 anomalies as normal and 220 normal instances as anomalies, which reveals a small imbalance in sensitivity between classes.

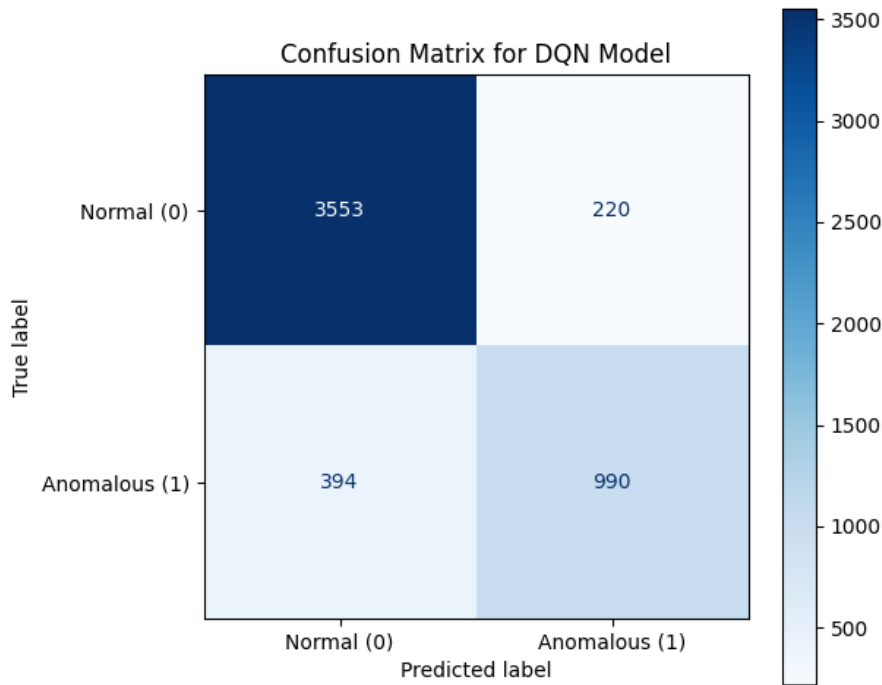


Figure 6 : Confusion matrix of the DQN model for velocity anomaly detection

These figures represent the learning behavior and performance of the DQN model trained for velocity anomaly detection under the first data combination: payload = 1.6 kg, half speed, and normal temperature. The validation accuracy curve and smooth loss decrease confirm that the agent is learning an effective policy. While the confusion matrix shows high performance overall, some velocity-related anomalies remain difficult to detect. This may be due to the dynamic and transient nature of velocity anomalies in robotic motion. Despite these challenges, the DQN model achieves strong classification metrics with a test accuracy of 88.1%, and macro-averaged F1-score of 0.84. These results confirm that deep reinforcement learning is a promising direction for detecting subtle temporal anomalies in robotic systems. Future improvements could involve advanced RL methods or hybrid architectures to increase recall on challenging cases.

6. Limitations

Despite the promising results obtained with the deep reinforcement learning approach, several limitations should be acknowledged:

- **Class Imbalance:** One of the main challenges in robotic anomaly detection lies in the strong imbalance between normal and anomalous classes. In our dataset, normal behaviors are significantly more frequent, which can bias the model during training. This is particularly reflected in a lower recall for the minority (anomalous) class, even when precision remains relatively high. As a result, the model tends to correctly identify normal states but may fail to detect subtle or rare faults that deviate from typical patterns.
- **Lack of Real or Physical Feedback:** All experiments were conducted offline, using pre-recorded data, without real-time interaction with a physical robot or virtual environment. This means the model is not exposed to dynamic feedback loops,

environmental noise, or real-world perturbations. Consequently, its generalizability to real-time deployment scenarios remains untested, and its behavior in unpredictable or noisy environments is still uncertain.

- **Limited Temporal Modeling:** Although time-series data were used as input, the DQN architecture does not fully exploit long-range temporal dependencies. Actions are often chosen based on short-term observations, which may limit the model's ability to detect evolving or delayed anomalies. This is a limitation in tasks where subtle faults unfold over extended periods and require deeper temporal understanding.

7. Future Work

To further improve the performance and robustness of anomaly detection using reinforcement learning, several future research directions are worth exploring:

- **Use of Temporal-Aware RL Models:** Future investigations could benefit from RL algorithms that are better suited for sequential data, such as Recurrent DQNs, Double DQNs, or actor-critic methods like A2C or PPO. These methods can explicitly incorporate time dependency, allowing the model to fully exploit the Robot_time variable and better capture long-range temporal patterns in the data.
- **Incorporating Attention Mechanisms or Hierarchical Modeling:** The addition of an attention mechanism could help the model focus on the most relevant parts of the sequence when detecting anomalies. Likewise, hierarchical modeling, which refers to a structured approach where low-level signals such as joint currents are first processed independently and then combined at a higher decision level, could improve the model's interpretability and robustness when facing complex robotic behaviors.
- **Evaluation in a Simulated Environment:** Coupling the RL agent with a physics-based robotic simulator would enable closed-loop training and evaluation. This setup would provide real-time feedback and allow the model to interactively learn and adapt in dynamic conditions. Such interaction would also help assess the model's response to noise, uncertainty, or previously unseen perturbations, bridging the gap between offline learning and real-world deployment.
- **Data Augmentation and Class Rebalancing:** To address the issue of class imbalance, synthetic data generation or rebalancing techniques like SMOTE (Synthetic Minority Over-sampling Technique) can be used to create additional anomalous samples. This could improve the model's generalization and especially boost recall for rare fault types. Further exploration of generative approaches or domain adaptation could also help expand the training dataset with realistic yet diverse scenarios.
- **Integration of Supervised Sequential Models in Reinforcement Learning:** Another promising direction for future work is the integration of supervised learning models, particularly LSTM networks, into the RL pipeline. Recent trends highlight the growing interest in hybrid approaches that combine the strengths of pretrained sequential models with the adaptive capabilities of RL. Since LSTM networks are inherently designed to capture long-term dependencies in time series data and can be pretrained on large datasets, they provide a valuable prior that can accelerate RL training by starting from an informed representation rather than learning from scratch. Incorporating such models as feature extractors or reward generators within the RL loop could enhance the agent's ability to detect subtle or rare anomalies in robotic systems. This synergy between

supervised temporal modeling and reinforcement learning offers a structured and efficient approach for tackling complex, time-dependent tasks such as fault detection in robotics.

8. Conclusion

This work demonstrates the promising potential of deep reinforcement learning, specifically Deep Q-Networks, in detecting anomalies in industrial robotic systems. By learning to identify deviations from normal behaviors based on reward signals and temporal interaction, the model shows strong performance across multiple targets, including position, velocity, and current-based anomalies. The results reveal that the RL-based approach achieves high classification accuracy and competitive macro-averaged precision, recall, and F1-scores, despite the challenges posed by imbalanced data and noisy signals. Notably, the model performs particularly well on position anomalies, while velocity and current faults remain more difficult to detect, highlighting the importance of temporal variability and subtle feature shifts in anomaly detection. While the model is trained and validated under a single combination of operating conditions, its architecture and methodology open the door for more generalizable and adaptive anomaly detection strategies. Future work will focus on integrating advanced RL variants, temporal modeling, attention mechanisms, and interactive simulation environments to further enhance performance, interpretability, and deployment readiness. In summary, this study positions reinforcement learning as a viable and scalable alternative to traditional supervised learning approaches for anomaly detection in robotics, especially in settings where labeled data is scarce and dynamic behavior needs to be captured over time.

Acknowledgements

This material is based upon work supported by the U.S. Department of Defense under the Office of Local Defense Community Cooperation (OLDCC) Award Number MCS2106-23-01. The views expressed herein do not necessarily represent the views of the U.S. Department of Defense or the United States Government. We would like to acknowledge the National Institute of Standards and Technology (NIST) for making their raw data available.

References

- [1] **Bendaouia, A.; Messaoudi, S.; Abdelwahed, E. H. and Li, J. (2025).** Robots Performance Monitoring in Autonomous Manufacturing Operations Using Machine Learning and Big Data. In Proceedings of the 14th International Conference on Data Science, Technology and Applications, ISBN 978-989-758-758-0, ISSN 2184-285X, pages 84-98.
- [2] National Institute of Standards and Technology (NIST), *Degradation Measurement of Robot Arm Position Accuracy*, [Online]. Available: <https://www.nist.gov/el/intelligent-systems-division-73500/degradation-measurement-robot-arm-position-accuracy>
- [3] **P. Ajidarma and S. Y. Nof,** “*Human-Robot Collaborative Reinforcement Learning in Semi-Automated Manufacturing Operations,*” *IFAC-PapersOnLine*, vol. 58, no. 19, pp. 528–532, 2024. <https://doi.org/10.1016/j.ifacol.2024.09.266>

- [4] **M. Matulis and C. Harvey**, *A robot arm digital twin utilising reinforcement learning*, *Computers & Graphics*, vol. 95, pp. 106–114, 2021. <https://doi.org/10.1016/j.cag.2021.01.011>
- [5] **D. Kim, M. Choi, and J. Um**, “*Digital twin for autonomous collaborative robot by using synthetic data and reinforcement learning*,” *Robotics and Computer–Integrated Manufacturing*, vol. 85, 2024. <https://doi.org/10.1016/j.rcim.2023.102632>
- [6] **Koch, D., Kaiser, J.-P., Stamer, F., Stark, R., & Lanza, G. (2025)**. *Enhancing visual inspection in remanufacturing: A reinforcement learning approach with integrated robot simulation*. *Procedia CIRP*, 134, 939–944. <https://doi.org/10.1016/j.procir.2025.02.228>
- [7] **Pico, N., Montero, E., Alvarez-Alvarado, M. S., Amirbek, A., Jamil, B., Auh, E., Jeon, J., Algabri, R., & Moon, H. (2025)**. *Human and environmental feature-driven neural network for path-constrained robot navigation using deep reinforcement learning*. *Engineering Science and Technology, an International Journal*, 64, 101993. <https://doi.org/10.1016/j.jestch.2025.101993>
- [8] **T. Johannink, S. Bahl, A. Nair, J. Luo, A. Kumar, M. Loskyll, J. A. Ojea, E. Solowjow, and S. Levine**, “*Residual Reinforcement Learning for Robot Control*,” 2019 International Conference on Robotics and Automation (ICRA), pp. 6023–6029, 2019. <https://doi.org/10.48550/arXiv.1812.03201>
- [9] **S. Wang, J. Zhang, P. Wang, J. Law, R. Calinescu, and L. Mihaylova**, “*A Deep Learning-Enhanced Digital Twin Framework for Improving Safety and Reliability in Human–Robot Collaborative Manufacturing*,” *Robotics and Computer-Integrated Manufacturing*, vol. 85, 2024. <https://doi.org/10.1016/j.rcim.2023.102608>
- [10] **Wang, J., Yan, Y., Hu, Y., Yang, X., & Zhang, L. (2025)**. *A transfer reinforcement learning and digital-twin based task allocation method for human-robot collaboration assembly*. *Engineering Applications of Artificial Intelligence*, 128, 110064. <https://doi.org/10.1016/j.engappai.2025.110064>
- [11] **Sun, S., Li, C., Zhao, Z., Huang, H., & Xu, W. (2024)**. *Leveraging large language models for comprehensive locomotion control in humanoid robots design*. *Biomimetic Intelligence and Robotics*, 4, 100187. <https://doi.org/10.1016/j.birob.2024.100187>
- [12] **Habibkhah, F., & Moallem, M. (2025)**. *Application of machine learning for seam profile identification in robotic welding*. *Machine Learning with Applications*, 20, 100633. <https://doi.org/10.1016/j.mlwa.2025.100633>
- [13] **Kuo, P.-H., Yeh, L.-C., Chang, C.-W. & Feng, P.-H. (2025)**. *Developing an automatic gripping learning system for a robotic arm by integrating a convolutional neural network and optimization algorithms*. *Results in Engineering*, 105026. <https://doi.org/10.1016/j.rineng.2025.105026>

- [14] **Castellano-Queroa, M., Castillo-López, M., Fernández-Madrigan, J.-A., Arévalo-Espejo, V., Voos, H., & García-Cerezo, A. (2023).** *A multidimensional Bayesian architecture for real-time anomaly detection and recovery in mobile robot sensory systems.* *Engineering Applications of Artificial Intelligence*, 125, 106673. <https://doi.org/10.1016/j.engappai.2023.106673>
- [15] **Souma, D., Khan, S., & Mori, A. (2023).** *Towards Online Health Monitoring of Robotic Arm.* National Institute of Advanced Industrial Science and Technology, Kansai, Japan.
- [16] **Chen, J., Al-Nussairi, A. K. J., Khosravi, M., Jin, K., Zhang, J., & Chyad, M. H. (2025).** *Advanced multi-loop control for 4DOF robotic arms: Integrating digital twins, neural networks, and model predictive control.* *Energy Reports*, 13, 4261–4279. <https://doi.org/10.1016/j.egy.2025.03.052>
- [17] **Nowak, T., Große-Kreul, A., Boshoff, M., & Kuhlenkötter, B. (2024).** *Enhancing mobile robot position estimation with machine learning methods using camera-based tracking.* *Procedia CIRP*, 130, 964–968. <https://doi.org/10.1016/j.procir.2024.10.192>
- [18] **El Kalach, F., Farahani, M., Wuest, T., & Harik, R. (2025).** *Real-time defect detection and classification in robotic assembly lines: A machine learning framework.* *Robotics and Computer-Integrated Manufacturing*, 95, 103011. <https://doi.org/10.1016/j.rcim.2025.103011>
- [19] **Natarajan, R., Reddy, S. P., Bose, S. C., Gururaj, H. L., Flammini, F., & Velmurugan, S. (2023).** *Fault detection and state estimation in robotic automatic control using machine learning.* *Array*, 19, 100298. <https://doi.org/10.1016/j.array.2023.100298>
- [20] **Sabry, A. H., & Ungku Amirulddin, U. A. B. (2024).** *A review on fault detection and diagnosis of industrial robots and multi-axis machines.* *Results in Engineering*, 23, 102397. <https://doi.org/10.1016/j.rineng.2024.102397>
- [21] **Shah, R., Arockia Doss, A. S., & Lakshmaiya, N. (2025).** *Advancements in AI-enhanced collaborative robotics: towards safer, smarter, and human-centric industrial automation.* *Results in Engineering*, 27, 105704. <https://doi.org/10.1016/j.rineng.2025.105704>