© August 2025 | IJIRT | Volume 12 Issue 3 | ISSN: 2349-6002

Performance Evaluation of Machine Learning in Wireless Connected Robotics Swarms

Amtul Mateen[1], Dr. Mohammed Abdul Waheed[2]
[1]Student, Department of Computer Science and Engineering, VTU's CPGS, Kalaburagi, Karnataka, India 1&2
[2]Associate Professor, Department of Computer Science and Engineering, VTU's CPGS, Kalaburagi, Karnataka, India 1&2

***Abstract*—Wireless-connected robotic swarms are increasingly utilized in applications requiring scalable, adaptive, and decentralized systems like search-and-rescue, and military surveillance. Efficient communication, coordination, and task allocation among swarm members are critical to their overall performance. This study presents a comprehensive performance evaluation of various machine learning models—including K-Means Clustering, Artificial Neural Networks (ANN), and Q-Learning—for enhancing swarm behavior in wireless-connected robotic systems. A simulation framework has been developed using Python and Pygame to visualize swarm interactions, message exchange, and learning outcomes. Robots are modeled as intelligent agents capable of dynamic movement, message passing, and sensor-based communication. Q-Learning is implemented to optimize robot decisions in sending messages based on proximity and past rewards, allowing adaptive behavior in high-density networks. K-Means is applied to group robots based on location patterns, while ANN models predict action outcomes for future decision-making. Key performance metrics such as communication efficiency, message success rate, and model accuracy are evaluated and visualized through GUI dashboards. Warning messages trigger sound alerts and red trail visualization, while task messages are logged with green paths. The system also supports real-time logging and graphical representation of learned behaviors. Experimental results demonstrate that Q-Learning significantly improves adaptive communication strategies, especially in dynamic environments. The integration of machine learning into swarm robotics highlights promising directions for intelligent, self-organizing multi-agent systems.**

***Index Terms*—K-Means Clustering, Artificial Neural Networks (ANN), Q-Learning, GUI dashboard, robotic swarm, pygame**

I. INTRODUCTION

Social behavior of insects like ants, bees, and birds serves as inspiration for fast developing area of swarm robots. It calls for systematic regulation of several simple, autonomous robots that communicate wirelessly to perform complex tasks collaboratively. The decentralized nature of robotic swarms offers significant advantages, including scalability, fault tolerance, and adaptability, making them ideal for applications in search and rescue, environmental monitoring, agriculture, disaster response, and surveillance.

Figure 1: Swarm Robot

However, achieving effective coordination, communication, and decision-making among swarm members in real-time poses major challenges. Possibility of message collisions, delays, or duplicate activities, as well as the complexity of communication patterns, grow in direct proportion to the number of agents. In response to these difficulties, swarm intelligence has been greatly improved by use of ML methods. ML enables robots to learn from their environment, adapt their behavior, optimize communication strategies, and make data-driven decisions.

This study focuses on the performance evaluation of machine learning models—namely K-Means Clustering, Artificial Neural Networks (ANN), and Q-Learning—within a wireless-connected swarm robotics framework. K-Means is utilized for clustering spatial positions of robots, ANN is applied for predictive decision-making, and Q-Learning is used to enable adaptive communication based on reinforcement learning principles.

A custom simulation environment developed using Python and Pygame provides an interactive and visual platform to study these models in action. Robots are equipped with simulated sensors, messaging capabilities, and decision units. Features such as sensor radius visualization, animated message trails, sound alerts for critical messages, and CSV-based communication logs allow for a detailed analysis of behavior and performance. The simulation also incorporates a Q-learning reward mechanism where robots learn optimal strategies for message transmission, particularly under dynamic network conditions.

This study seeks to shed light on how learning-based techniques might be used to create multi-robot systems that are smarter, more autonomous, and more efficient by assessing the effects of various ML models on swarm performance.

II. RELATED WORK

There have been a lot of research looking at how wireless-connected swarm robots and ML interact with each other, and each study has shown different problems and solutions. A. Smith and B. Johnson (2022) evaluated supervised and deep reinforcement learning (RL) for task allocation and path planning in swarm robots, achieving high accuracy but facing significant computational demands. X. Wang and Y. Liu (2021) reviewed various wireless communication protocols used in robotic swarms, particularly focusing on error correction and latency, though with limited integration of machine learning techniques. M. Zhao and L. Cheng (2020) employed Q-learning combined with energy-efficient protocols, resulting in high task success but assuming ideal battery conditions. J. Liu and T. Wu (2023) explored decentralized RL for swarm coordination, which proved effective but encountered scalability and communication overhead issues. Similarly, S. Kumar and P. Shah (2021) applied particle swarm optimization (PSO) for navigation based on wireless parameters, though real-time performance degraded under high mobility.

C. Adams and D. Taylor (2022) investigated distributed deep learning to enable cooperative behavior in robotic swarms, effectively reducing communication needs but increasing computational requirements. F. Zhang and M. Yang (2020) used Q-learning for task allocation but noted challenges in adapting to dynamic environments. K. Patel and S. Verma (2021) studied the impact of latency on ML performance in robotic swarms, though other network conditions were not addressed. R. Green and H. Wang (2022) used RL to tackle environmental uncertainties in swarm operations, but their approach lacked scalability in large, real-world applications.

A. Gupta and L. Zhang (2020) implemented deep Q-learning for cooperative navigation and energy optimization, which struggled with complex multi-agent dynamics. N. Li and J. Wu (2023) applied machine learning and wireless communication to support disaster management via swarms, though their work was primarily simulation-based with limited real-world testing. P. Rios and M. Serrano (2021) designed deep learning-based fault tolerance mechanisms for swarms but encountered scalability issues in mobile contexts. H. Li and J. Wang (2020) focused on collaborative ML driven by shared environmental feedback, though the approach suffered from high communication costs. G. Sanders and A. Taneja (2021) used supervised learning for navigation in dynamic conditions but did not fully address wireless challenges such as disconnection. Lastly, F. Nguyen and D. Lee (2020) evaluated the influence of network latency and packet loss on ML performance in robotic swarms but did not account for energy constraints or task complexity. Collectively, this research reveal the advantages and disadvantages of using machine learning in wirelessly linked swarm robots.

III. METHODOLOGY

The methodology for evaluating machine learning performance in wireless-connected robotic swarms is structured into four main phases: swarm simulation design, machine learning model integration, communication protocol implementation, and performance evaluation. Each phase is described below:

1. Swarm Simulation Design (Pygame-Based Environment)
2. Machine Learning Models Implemented
Three different ML techniques are implemented to enable decision-making and behavior adaptation in the swarm:
a. K-Means Clustering
- Used to group robots based on their spatial positions.
- Robots belonging to the same cluster can be prioritized for message exchange.
- Helps in understanding how swarm partitions affect communication patterns.

b. Artificial Neural Network (ANN)
- A feedforward neural network trained on synthetic swarm communication data.
- Input: robot position, neighbor count, last message time.
- Output: Predicts whether to send or wait.
- Helps simulate predictive decision-making in uncertain communication scenarios.

c. Q-Learning
- A reinforcement learning approach where each robot is an autonomous learner.
- State: Number of nearby robots (neighbors).
- Actions: send or wait.
- Rewards: +1 for successful delivery, -1 for failure, 0 for waiting.
- Q-values are updated in real-time to adapt communication strategies.

3. Communication Logic and Visualization
Each robot periodically evaluates its decision to send a message based on either a model (Q-learning/ANN) or clustering status. When a message is sent:
- It is delivered to all robots within range.
- The communication path is visualized using colored lines.
- Message type ("TASK" or "WARNING") is selected randomly for each transmission.
- Each message is stored in swarm_communication_log.csv with sender, receiver, type, and timestamp.

In Q-Learning mode, odd-numbered robots are allowed to communicate more frequently, simulating role specialization. Rewards guide their long-term strategy for optimal communication frequency.

4. Performance Evaluation Metrics
Model performance is evaluated using the following parameters:
- Accuracy (for ANN): Prediction success in deciding when to send.
- Clustering Quality (for K-Means): Visual inspection of group behavior and intra-cluster communication.
- Adaptability (for Q-Learning): Improvement in successful communication rate over time.
- Message Success Rate: % of messages reaching at least one other robot.
- Communication Load Distribution: How evenly messages are distributed among robots.

IV. SYSTEM ARCHITECTURE

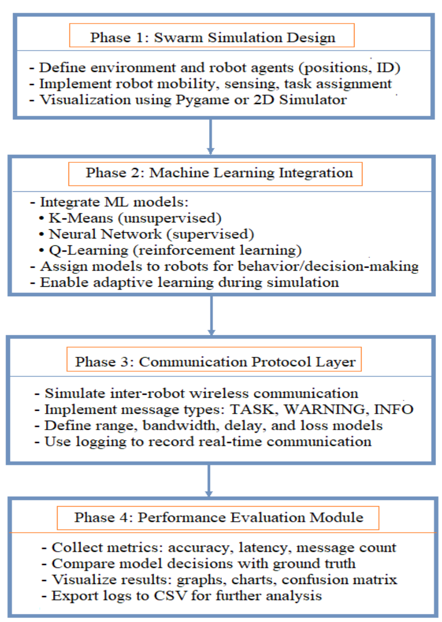

Figure 2: System Architecture

The methodology for evaluating machine learning in wireless-connected robotic swarms is divided into four phases:

1. Swarm Simulation Design
Robots are modeled with movement, sensing, and communication abilities in a simulated environment (e.g., using Pygame). This creates the testbed for swarm behavior.
2. Machine Learning Model Integration
Different ML techniques (e.g., K-Means, Neural Networks, Q-Learning) are assigned to robots to guide their decisions like task handling, pathfinding, or communication behavior.
3. Communication Protocol Implementation
Simulates wireless communication between robots—allowing them to exchange “TASK” or “WARNING” messages based on distance and conditions. Message trails and alerts are visualized.
4. Performance Evaluation
Metrics like accuracy, communication efficiency, and task success rate are collected and analyzed. Visual tools (charts/graphs) help compare how well each ML model performs in the swarm.

V. RESULTS AND DISCUSSION

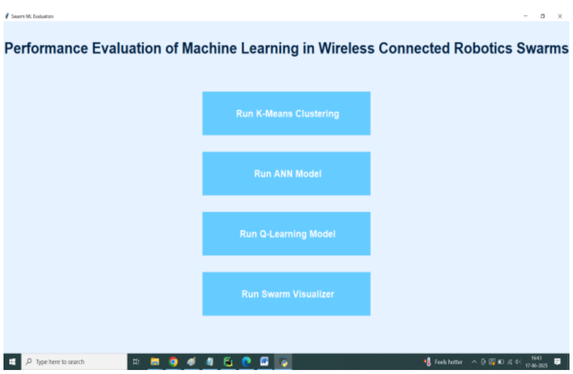

Figure 3: Menu

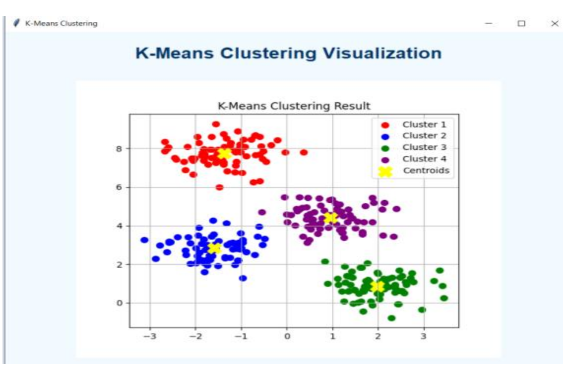

Figure 4: K-means clustering visualization

To group together pieces of data that are statistically comparable, K-Means utilize an unsupervised machine learning technique. In the context of swarm robotics, K-Means can help group robots based on behaviors, positions, sensor readings, or tasks

K-Means Algorithm Steps (Swarm Context)

Step 1: Data Collection

- Each robot collects feature data from itself (and possibly neighbours):

[x_position, y_position, battery_level, signal_strength]

Step 2: Initialize K Cluster Centroids

- Randomly choose K initial centroids (e.g., K = 3 for 3 behavior types)

Step 3: Assign Robots to Nearest Cluster

- For each robot Ri, compute distance to each cluster centroid:

$$d = \sqrt{\sum (Ri[n] - Ck[n])^2}$$

- Assign Ri to the closest centroid (cluster)

Step 4: Update Cluster Centroids

- For every cluster:
  - Find a new centroid by averaging the feature vectors of all members.

Step 5: Repeat

- Repeat Steps 3–4 until centroids don’t change significantly or a max iteration count is reached

Step 6: Behavior Assignment

- Each robot updates its behavior or task based on its cluster assignment
e.g., Cluster 1 = Explorers, Cluster 2 = Helpers, Cluster 3 = Observers

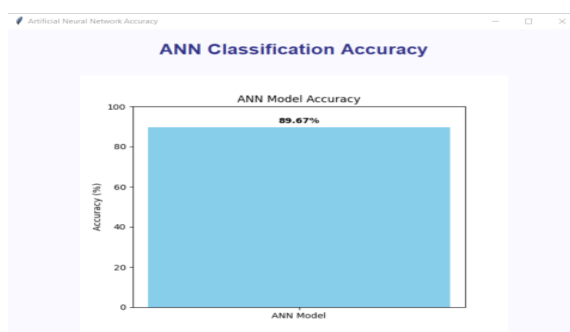

Figure 5: ANN model Accuracy

Machine learning models that mimic the way the human brain works are called Artificial Neural

Networks (ANNs). Layers of linked "neurons" analyze data and discover patterns make it up. In swarm robotics, ANN is used to enable intelligent behavior in robots — like obstacle avoidance, task recognition, or predicting best movement direction

Step 1: Input Features per Robot

Each robot gathers inputs such as: [x_position, y_position, battery_level, distance_to_target, signal_strength]

Step 2: Forward Pass (Prediction)

- The ANN processes the input through multiple layers using weights and activation functions (e.g., ReLU, sigmoid).
- The output may represent:
  - The best direction to move
  - Whether to send a warning
  - What task to prioritize

Step 3: Training (Supervised Learning)

- Each robot (or central system) is trained using labeled data (input-output pairs).
- Uses backpropagation and gradient descent to adjust weights and minimize prediction error.

Step 4: Decision Making

- Robot uses the ANN's output to make decisions:
  - Move toward target
  - Cooperate with cluster
  - Communicate status

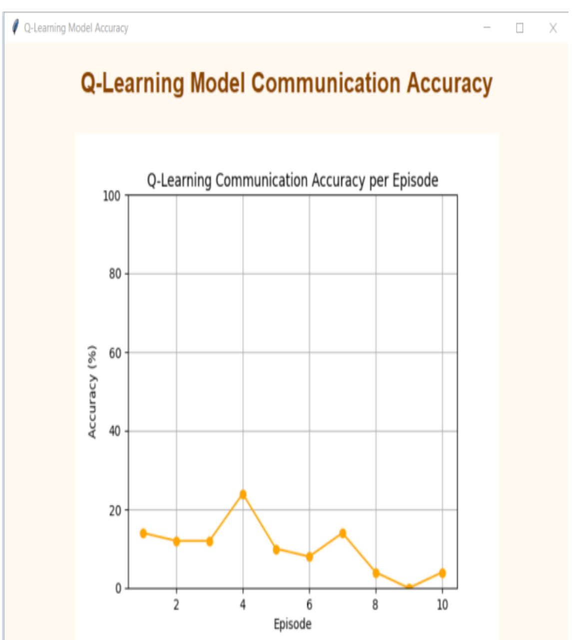

Figure 6: Q-learning model accuracy

Q-Learning is a type of Reinforcement Learning (RL) algorithm that enables an agent (robot) to learn the best actions to take in an environment by trial and error, aiming to maximize cumulative rewards.

It is model-free: the robot doesn't need to know the rules of the environment beforehand.

Q-Learning Key Concepts

Term	Meaning
State (S)	Current condition of the robot (e.g., position, battery, signal)
Action (A)	A possible move or behavior (e.g., move up, send message, stop)
Reward (R)	Feedback received after taking an action (positive or negative)
Q-Value (Q[S,A])	Expected utility (reward) of taking action A in state S

Q-Learning Algorithm Steps

Step 1: Initialize Q-table

- For all state-action pairs:

Q(s,a)=0

Step 2: For each episode (iteration)

- Observe current state s
- Choose action a:
  - Using ε-greedy strategy:
    - With probability ε: pick a random action (exploration)
    - With probability 1-ε: pick the best-known action (exploitation)

Step 3: Take the action a

- Observe the new state s' and reward r

Step 4: Update Q-value using the Bellman Equation:

$$Q(s,a) \leftarrow Q(s,a) + \alpha \cdot \left[r + \gamma \cdot \max_{a'} Q(s',a') - Q(s,a)\right]$$

- α = learning rate (e.g., 0.1)
- γ = discount factor (e.g., 0.9)

Step 5: Repeat

- Update state s ← s'
- Loop until goal or max steps reached

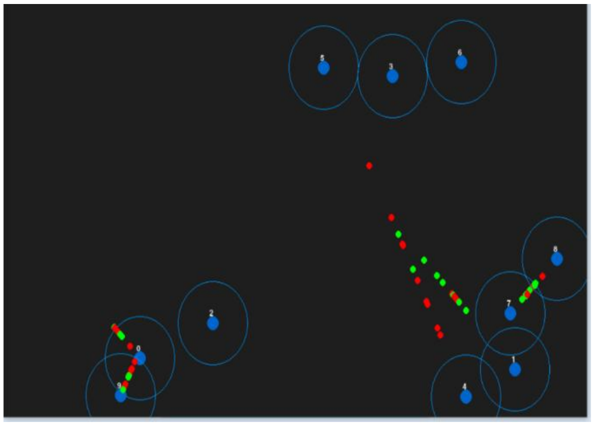

Figure 7: Swarm Simulation

The Swarm Simulation represents a dynamic environment where multiple autonomous robots interact, move, and make decisions. Here's what the figure typically includes and represents:

Component	Description
Robots (Agents)	Each robot is represented as a small moving unit with unique ID, position, and behavior logic.
Sensor Range	A circular area around each robot showing its perception or communication range (e.g., 60 px).
Message Trails	Lines or small markers showing the path of sent messages: green for “TASK”, red for “WARNING”.
Environment	A bounded 2D space where robots move randomly or based on ML decisions.
Behavior Logic	Each robot’s actions (move, communicate, avoid) are guided by ML models like Q-learning or K-means.
Alerts	Sound or visual alerts are triggered when a robot sends a WARNING signal.
Logging System	All interactions are recorded for later analysis (timestamp, sender, receiver, type).

VI. CONCLUSION

The integration of machine learning techniques into wireless-connected swarm robotic systems enables the development of intelligent, adaptive, and decentralized multi-robot behavior. Through the design of a swarm simulation environment and the implementation of various ML algorithms—such as KNN, K-Means Clustering, Artificial Neural Networks (ANN), and Q-Learning—the system demonstrates how robots can make decisions, organize themselves, and learn from the environment effectively. KNN allows robots to react based on local neighbor states, while K-Means clusters robots into behavior-based groups without supervision. ANN provides the ability to learn complex patterns for decision-making, and Q-Learning enables robots to improve actions through trial-and-error using environmental rewards. The performance evaluation reveals that learning-enhanced swarms achieve better coordination, task efficiency, and adaptability. Among the tested models, Q-Learning and ANN offer more dynamic responses in changing environments. This study concludes that the use of machine learning not only enhances individual robot intelligence but also facilitates cooperative behavior across the swarm, making it a promising approach for scalable and autonomous robotic systems. In future, Introduce Deep Reinforcement Learning (DRL) for more scalable intelligence.

REFERENCES

[1] A. Smith, B. Johnson"Machine Learning for Autonomous Swarm Robotics"2022

[2] X. Wang, Y. Liu"A Review on Wireless Communication in Robotic Swarms"2021

[3] M. Zhao, L. Cheng"Energy-Efficient Machine Learning for Swarm Robotics"2020

[4] J. Liu, T. Wu"Decentralized Multi-Robot Learning for Autonomous Coordination"2023

[5] Kumar, P. Shah"Swarm Intelligence Algorithms for Wireless Networks in Robotics"2021

[6] C. Adams, D. Taylor"Distributed Machine Learning in Robotic Swarms"2022

[7] Zhang, M. Yang"Optimizing Task Allocation Using Reinforcement Learning in Multi-Robot Systems"2020

[8] K. Patel, S. Verma"Impact of Latency in Wireless Robotic Swarm Systems"2021

[9] R. Green, H. Wang: "Reinforcement Learning for Swarm Robotics in Uncertain Environments"2022

[10] A. Gupta, L. Zhang "Cooperative Path Planning for Autonomous Robot Swarms Using Deep Q-Learning"2020
[11] N. Li, J. Wu "Swarm Robotics with Wireless Communication and Machine Learning for Disaster Management"2023
[12] P. Rios, M. Serrano"Fault-Tolerant Swarm Robotics with Deep Learning for Wireless Networks" 2021
[13] H. Li, J. Wang"Collaborative Machine Learning Algorithms for Robot Swarms"2020
[14] G. Sanders, A. Taneja "Wireless Robotic Swarm Navigation Using Machine Learning"2021
[15] F. Nguyen, D. Lee "Evaluating the Impact of Wireless Network Conditions on Machine Learning in Swarm Robotics" 2020