International Journal of Computer Information Systems and Industrial Management Applications
ISSN 2150-7988 Volume 18(2) 2026 pp. 668-681
https://doi.org/10.70917/ijcisim-2026-3007
Article

Efficient Data Collection in UAV Sensors using Optimized Double Deep Q-learning Network (ODDQN)

Sachin Paranjape[1], Barani S.[2], Mukul Sutaone[3]

[1] Department of Electronics and Telecommunication Engineering, MKSSS' Cummins College of Engineering for Women, Pune, Maharashtra, India.
Email: sachin.paranjape@cumminscollege.in
[2] Department of Electronics and Control Engineering, Sathyabama Institute of Science and Technology, Chennai, Tamil Nadu, India.
Email: baraniselvaraj77@gmail.com
[3]Indian Institute of Information Technology (IIIT) Allahabad, Prayagraj, Uttar Pradesh, India.

Abstract: Introduction: In recent years, Unmanned Aerial Vehicles (UAVs) have developed as a highly promising solution for enhancing data-collection efficiency of Wireless Sensor Networks (WSNs). Most existing works on data collection by UAVs concentrate on clustering, trajectory planning, or resource allocation, ignoring congestion caused by massive simultaneous transmissions, which becomes critical in large-scale IoT deployments. Existing MAC and scheduling schemes fail to provide low latency, high throughput, and conflict-free access when the number of sensors grows. **Materials and Methods:** In order to solve these issues, this paper outlines the design for a Double Deep Q-learning Network (DDQN) model to solve the problem of determining optimal data collection schedules for UAV sensors. The DDQN utilizes two identical deep neural networks to mitigate the overestimation bias inherent in standard DQN: the current Q-Network for action selection and the target Q-Network for value evaluation. The DDQN parameters are further optimized by applying backpropagation technique. The reward function is formed to maximize data freshness while minimizing operational costs. **Results:** The performance of ODDQN is evaluated in terms Landing Success percentage, data collection ratio, data collection delay, packet drop rate and average residual energy. **Discussion:** Experimental results show that the ODDQN model attains maximum data collection ratio with reduced data collection delay and packet drop rate. **Conclusion:** Hence the ODDQN framework determines optimal data collection schedules for UAV sensors.
Keywords: Unmanned Aerial Vehicles (UAVs), Wireless Sensor Networks, Data collection, Double Deep Q-learning Network (DDQN).

1. Introduction

Wireless Sensor Networks (WSNs) comprise large numbers of low-cost sensor nodes that function on severely constrained energy supplies, characteristically small batteries that are difficult or impossible to substitute once deployed. As these nodes have to support continuous sensing, processing, and communication, energy efficiency is a fundamental design objective to extend network lifetime and preserve reliable operation. Traditional WSN deployments are affected by energy wastage because of long-distance transmissions from SNs to fixed sink nodes, ineffective routing paths, and redundant sensing activities. Consequently, the integration of intelligent data-collection strategies and energy-aware mechanisms has become essential in modern WSN research [1].

In recent years, unmanned aerial vehicles (UAVs) have developed as a highly promising solution for enhancing data-collection efficiency in WSNs. With their high mobility and flexible deployment, UAVs can fly near to ground sensors and serve as mobile data collectors. UAV-based WSNs significantly lower transmission power requirements by minimizing the communication distance between SNs and the data collector, allowing the sensors to preserve energy, still maintaining reliable data delivery. UAVs can dynamically alter their trajectories, visitation order, and flying locations to exploit short-distance, line-of-sight (LoS) communication links, which have been shown to significantly improve the performance of aerial-ground wireless systems [2]. This mobility also allows UAVs to adapt to event-driven sensing patterns, uneven sensor distributions, or varying network conditions.

For further improving energy efficiency, several UAV–WSN systems integrate sleep–wake mechanisms, in which sensor nodes remain in deep-sleep mode until they detect an adequately strong wake-up beacon from the passing UAV. When activated, the sensors send their data and then immediately return to sleep, reducing idle listening and redundant energy expenditure. Even though effective, this model presents new design challenges. Initially, the wake-up schedule should be optimized [3]. Hence, each SN completes its transmission with less energy usage.

The mutual problem of improving UAV flight trajectory and SN wake-up schedules is complex and has only recently started to obtain research attention [4]. Previous works have studied cyclical multiple-access schemes, multi-UAV deployments, and energy-efficient trajectory design, but most of these approaches treat UAV motion or SN behaviour separately rather than as a joint optimization problem. Additionally, existing studies often simplify channel conditions, supervise fading effects, or overlook sensor-side energy consumption.

Machine learning (ML) offers powerful new opportunities for addressing these limitations and improving UAV-assisted WSN performance. ML models can optimize UAV trajectories, predict channel quality, cluster SN activation patterns, or dynamically adapt sleep–wake mechanisms based on observed network behaviour. Learning-driven UAV control can also lessen communication overhead, forestall sensor energy depletion, and adapt to non-uniform traffic conditions more efficiently than handcrafted models [5][6]. Through reinforcement learning, supervised learning, and predictive analytics, UAVs can make intelligent real-time decisions that lessen SN energy consumption, attaining high data-collection reliability.

1.1 Problem Statement

Current **UAV-assisted data collection frameworks** in Wireless Sensor Networks (WSNs) fail to maintain reliable, low-latency, and robust performance in large-scale Internet of Things (IoT) deployments due to three major limitations: [1]

- **Network Congestion**: Existing MAC and scheduling schemes ignore data traffic bottlenecks caused by massive simultaneous transmissions from scaled sensor nodes. This leads to a severe degradation of throughput and an escalation in latency
- **Location Uncertainty**: Most operational models operate under the impractical assumption of precise, static sensor coordinates. This assumption fails in GNSS-denied, eco-sensitive, or energy-constrained environments (such as ocean or agricultural monitoring), as joint sensing, positioning, and data collection models remain unexplored.
- **Environmental Instability**: Current architectures lack online optimization and machine learning adaptation. This makes them highly vulnerable to real-world mobility constraints, signal fading channels, and incomplete network state information.

Consequently, there is a critical need for a holistic, intelligent, and adaptive UAV–WSN data collection framework that concurrently manages channel access, handles node position uncertainty, and dynamically optimizes system choices under fluctuating channel states.

1.2 Objectives and Proposed Solution

To design, develop, and evaluate a holistic and intelligent UAV-assisted WSN data collection framework that minimizes latency, maximizes throughput, and maintains robustness under sensor location uncertainty and highly dynamic network environments.

In order to meet these objetives, this paper outlines the design for a DDQN model for determining the optimal data collection schedules for UAV sensors.

2. Related Works

Recent progresses in UAV-assisted WSNs have introduced diverse strategies to improve energy management, data collection efficiency, and communication reliability. Study in [6], where a comprehensive aerial data-collection framework is developed. This approach includes five core components such as sensor-node positioning, network deployment, anchor-point identification, UAV path planning, and data retrieval. They propose targeted solutions, such as the Fast Path Planning with Rules (FPPWR) algorithm, which utilizes grid-based division to significantly lessen path-planning complexity, maintaining short and feasible flight paths. For validating the design, a dedicated simulation platform is built, assessing performance across key indicators like UAV flight distance, data-collection duration, and the total volume of retrieved data. The results prove that the FPPWR-enabled framework attains fast and effective aerial data collection under diverse network conditions.

Energy efficiency in heterogeneous WSNs is investigated in [7], where adaptive working modes for sensor nodes are proposed based on their a dynamically optimized energy threshold and residual energy (RE). A

UAV is employed for collecting buffered data and alleviating the burden on energy-constrained nodes. They introduce a unified optimization framework that jointly finds the energy threshold, cluster-head selection, and node operation modes, with the objective of lessening the weighted sum of energy consumption from both the WSN and the UAV. An efficient search method is developed for identifying the optimal threshold, whereas a penalty-based successive convex-approximation scheme determines cluster heads. A low-complexity iterative algorithm is then utilized to solve the overall optimization problem. Simulation results show that this method significantly lessens the energy consumption of low-RE nodes and enhances overall WSN energy efficiency [7].

A collaborative approach is explored in [8], in which WSN is integrated with UAV receivers through collaborative beamforming (CB) and sensor-based virtual antenna arrays (SVAAs). Multiple sensor clusters create independent SVAAs to send data to selected UAVs, which then send the collected information to a ground control station. A bi-objective optimization problem to maximize transmission rate while preserving battery energy is formulated and solved using an Improved non-dominated sorting genetic algorithm (ENSGA-II). This method efficiently manages the non-convex and NP-hard nature of the problem. Extensive simulations confirm that the ENSGA-II surpasses several benchmark schemes while preserving sidelobe variations, robustness under interference, and UAV mobility effects [8].

ML-driven data-collection strategies are investigated in [9]. A UAV-aided model is proposed using K-means clustering to recognize optimal waypoints within the AERPAW digital-twin environment. This approach lessens flight time while preserving high SNR levels and dynamically adjusts hovering time depending on link quality and data backlog. Digital-twin simulations prove that the ML-based strategy attains improved responsiveness and efficiency in fast data-collection scenarios [9].

In [10], a UAV Fuzzy Travel Path mechanism is introduced by integrating deep reinforcement learning (DRL) with software-defined WSNs (SDWSNs). Gateway nodes collect data from group members and relay it to the UAV, whose trajectory is improved using a DRL-based Deep Q-Learning model. The objective is to lessen UAV energy consumption and exploit ground-to-UAV communication rates. This learning-based optimization significantly decreases energy usage while improving packet-delivery performance [10].

To examine UAV-enabled data collection within dense and dynamic networks, the Dual Cluster Head (UAVDCH) method in [11] allocates two cluster heads per cluster. It improves data-transfer reliability and UAV energy efficiency. Multi-channel operation further lessens hovering time. Results show better data acquisition relative to existing schemes.

Two prioritized contact-duration mechanisms (PCdFS and PMCdFS) and a Balance algorithm are proposed. It helps in ensuring collision-free scheduling across multi-UAV systems. Real-world tests prove improved fairness and communication stability, under highly dynamic network conditions.

The path planning problem is translated into a decentralized partially observable Markov decision process (Dec-POMDP), which is solved through a Deep Q-learning Network (DQN) approach

2.1 Research Gaps

In spite of extensive studies on UAV-assisted data collection in WSNs, several gaps remain unaddressed. Firstly, most existing works highlight clustering, trajectory planning, or resource allocation, but they ignore congestion caused by massive simultaneous sensor transmissions, which becomes critical in large-scale IoT deployments like ocean or agricultural monitoring. Existing MAC and scheduling schemes fail to provide low latency, high throughput, and conflict-free access when the number of sensors grows. Secondly, location uncertainty of sensors is infrequently combined into data collection models. Several algorithms assume precise sensor coordinates, which is unfeasible in scenarios lacking GNSS support or where GPS is too expensive or energy-consuming. Consequently, joint sensing–positioning–collection techniques remain underexplored. Then, existing studies seldom integrate ML-based prediction, adaptive scheduling, or online optimization to enhance robustness under mobility constraints, fading channels, and incomplete state information. These gaps motivate holistic, intelligent UAV–WSN data collection frameworks.

3. Proposed Methodology

This paper outlines the design for a DDQN model to solve the problem of determining optimal data collection schedules for UAV sensors, formulated as a Markov Decision Process (MDP).

3.1 System Model

The system model of the proposed UAV data collection process is depicted in Figure 1.

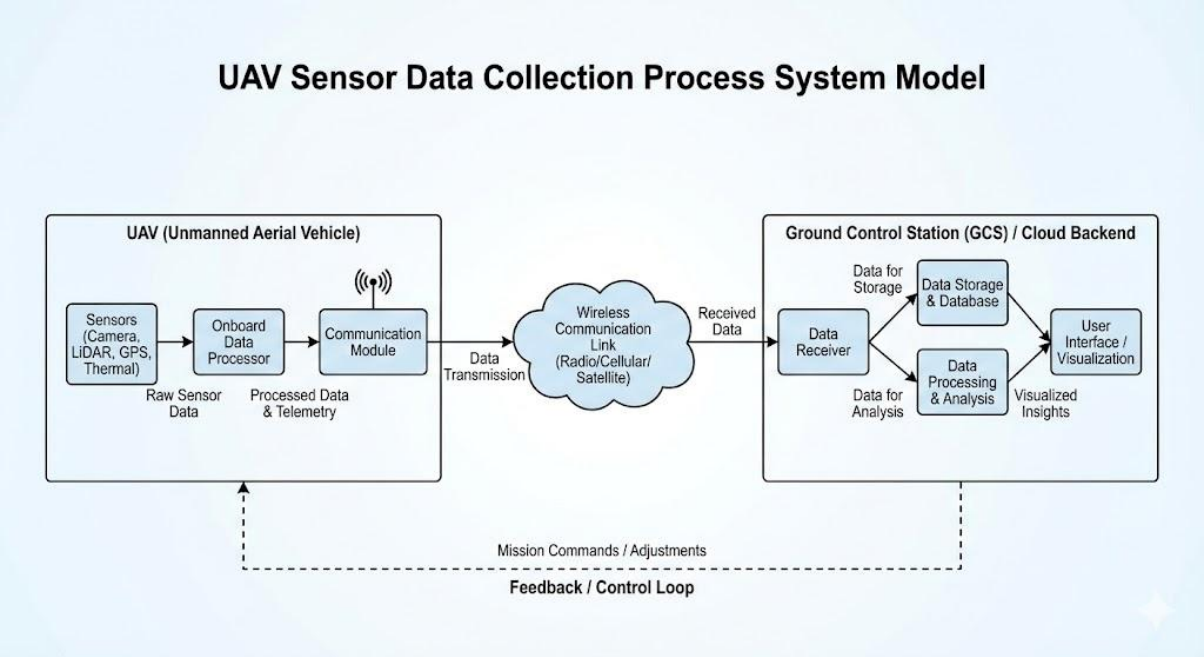

Figure 1 UAV data collection process

The depicted outlines the flow of data from an UAV to a Ground Control Station (GCS)/Cloud Backend, and the subsequent data processing and feedback mechanism.

3.1.1 UAV Components

The initial data collection and processing occur entirely on the UAV. This component consists of three main functional blocks:

- Sensors (Camera, LiDAR, GPS, Thermal):
    - Function: These are the primary sources of information, responsible for capturing environmental data.
    - Output: They generate Raw Sensor Data.
- Onboard Data Processor:
    - Function: This unit receives the raw data from the sensors and performs necessary processing, filtering, and compression.
    - Input: Raw Sensor Data.
    - Output: Processed Data & Telemetry.
- Communication Module:
    - Function: Responsible for transmitting the processed data off the UAV.
    - Input: Processed Data & Telemetry.
    - Output: Data Transmission.

3.1.2 Data Transmission Link

The link between the UAV and the ground-based system facilitates the transfer of the collected information.

- Wireless Communication Link (Radio/Cellular/Satellite):
    - Function: Serves as the medium for transmitting the Data Transmission from the UAV to the ground.
    - Output: Received Data at the ground station.

3.1.3 Ground Control Station (GCS) / Cloud Backend Component

Upon receiving the data, the GCS or Cloud Backend manages its storage, processing, and visualization. This component includes:

- Data Receiver:
    - Function: The initial point of reception for the Received Data from the wireless link.
    - Output: Channels the data to storage and analysis pipelines ("Data for Storage" and "Data for Analysis").
- Data Storage & Database:
    - Function: Persists the received data for future access, archiving, and retrieval.
    - Input: Data for Storage.
- Data Processing & Analysis:
    - Function: Applies advanced algorithms and analytical methods to the stored or received data to extract meaningful insights.
    - Input: Data for Analysis.
    - Output: Visualized Insights (which are passed to the User Interface/Visualization).
- User Interface / Visualization:
    - Function: Presents the processed data and analysis results in a user-friendly, visual format, allowing operators to understand the mission status and collected information.
    - Input: Visualized Insights.

3.1.4 Feedback / Control Loop

The system incorporates a closed-loop mechanism essential for dynamic mission management.

- Mission Commands / Adjustments:
    - Origin: Generated based on the Visualized Insights and operator decisions made at the GCS/Cloud Backend.
    - Path: These commands are sent back to the UAV, specifically interacting with the Onboard Data Processor (indicated by the dashed line).
    - Purpose: This loop allows the ground crew to send real-time adjustments or new mission commands to the UAV based on the data currently being received and analyzed, thereby optimizing the data collection process.

3.2 DDQN Model Design for UAV Scheduling

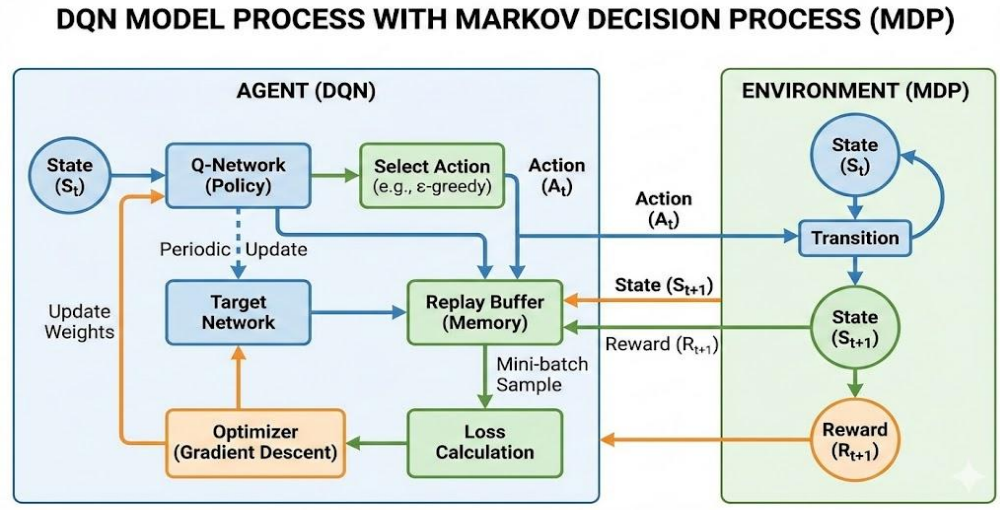

Figure 2 Process of DQN Model

Figure 2 shows he process involved in the DQN-MDP model.

3.2.1 Markov Decision Process (MDP) Formulation

The scheduling problem is defined by the tuple (S, A, R, P), where the UAV network controller acts as the agent.

A. State Space (S)

The state s_t at time t provides all necessary information for decision-making.

- **UAV Status:**
  - $L_{UAV}(t)$: Current **position** (e.g., (x, y, z) coordinates).
  - $E_{UAV}(t)$: Current **energy/battery level**.
  - DataQueue(t): Amount of data buffered, pending offloading.
- **Sensor/Target Status:**
  - $C_{Targets}(t) = \{c_1, \ldots. C_N\}$: Vector of **Age of Information (AoI)** for N target points.
  - $c_i(t)$ is the time since target I was last sampled (measured in time steps).
  - Priority(t) = $\{p_1, \ldots.. p_N\}$: Vector of dynamic or static **priorities** for each target.

$$S_t = (L_{UAV}(t), E_{UAV}(t), DataQueue(t), C_{Targets}(t), Priority(t)) \quad (1)$$

B. Action Space (A)

The action a_t is a **discrete set** of tasks the UAV can perform.

- **Sampling:** a_i: Move to and sample target point I (for I = 1 to N).
- **Maintenance:**
  - a_{charge}: Move to the charging station and recharge**.**
  - $a_{offload}$: Move to the ground station and offload data.
- **Wait:** a_{idle}: Hover/wait in the current location.

C. Reward Function I

The goal is to maximize data freshness (minimize AoI) while minimizing operational costs (energy/movement).

$$R_t = R_{sample} - R_{AoI} - R_{energy} - R_{Latency} \quad (2)$$

D. Transition Dynamics (P)

The transition to $s_{\{t+1\}}$ is deterministic, based on the physics of movement, energy depletion, and the reset of the AoI for the sampled target.

Table 1 shows the illustration of Reward components along with their formula and description.

Component	**Description**	**Formulation (Example)**
R_{Sample} (Positive)	Reward for successful sampling, weighted by target priority.	$R_{Sample}(i) = P_{sample}$. Pi
R_{AoI} (Penalty)	Penalty for the Age of Information across all targets.	AoI . $\sum_{i=1}^{N} c(t)^2$

Component	Description	Formulation (Example)
R_{Energy} (Penalty)	Penalty for energy consumed by the action taken.	$B_E \cdot \Delta E_{UAV}$
$R_{Latency}$ (Penalty)	Penalty for high data queue size (offload urgency).	$B_L \cdot DataQueue(t)^2$

Table 1 Illustration of Reward components

3.2.2. Architecture of DDQN Model

The DDQN utilizes two identical deep neural networks to mitigate the overestimation bias inherent in standard DQN.

A. Network Structure

Two DNNs are used: the Current Q-Network (θ) for action selection and the Target Q-Network (θ') for value evaluation.

- **Input Layer:** Takes the flattened State vector s_t.
- **Hidden Layers:** Multiple Fully Connected (FC) layers with ReLU activation.
- **Output Layer:** An FC layer with A outputs (equal to the number of actions). Each output represents the Q-value Q(s_t, a).

B. Training and Update

The network is trained using experience replay and a specialized target calculation.

1. **Experience Replay:** Transitions (s_t, a_t, $r_{\{t+1\}}$, $s_{\{t+1\}}$) are stored in a buffer and sampled randomly to train the networks.
2. **DDQN Target Q-Value (Yt^{DDQN}):** The target value is calculated by using the current network (θ) to select the best action in the next state $s_{\{t+1\}}$, and the target network **(θ')** to evaluate its value.

$$Yt^{DDQN}=r_t+\gamma\, Q_{\theta^-}(s_{\{t+1\}},\text{argmax}_{a'} Q_{\theta}(s_{\{t+1\}},a')) \quad (3)$$

3. **Loss Function:** The network **θ** is optimized to minimize the Mean-Squared Error (MSE) between the predicted Q-value and the target Q-value:

$$\mathcal{L}(\theta) = \mathbb{E}_{(s_t,a_t,r_t,s_{t+1})\sim\mathcal{D}}\left[\left(Y_t^{\text{DDQN}} - Q_\theta(s_t,a_t)\right)^2\right] \quad (4)$$

4. **Target Network Update:** The weights of the Target Network (**θ'**) are updated periodically (or via soft update) to match the weights of the Current Network (**θ**).

3.2.3 Backpropagation through the Online Network

$$\theta = \theta - \alpha\, \Delta_\theta L(\theta) \quad (5)$$

This updates only the online network parameters.

- gradients flow through $Q_\theta(s_t)$
- Not through the target network
- Not through the argmax selection

3.2.4 Pseudocode for DDQN process

Initialize online network Q_θ

Initialize target network $Q_{\theta'}$

Initialize replay buffer D

repeat for each episode:

s = initial state

repeat for each step:

Choose action a using ε-greedy(Q_θ)

Execute action a → observe r, s'

Store (s, a, r, s', done) in D

Sample minibatch from D

1. Predicted Q

$Q\_pred = Q_\theta(s)[a]$

2. Online network selects action

$a^* = \text{argmax}(Q_\theta(s'))$

3. Target network evaluates it

$target = r + \gamma * (1\text{-done}) * Q_{\theta'}(s')[a^*]$

4. Loss

loss = MSE(Q_pred, target)

5. Optimize online network

Backprop(loss)

6. Soft update target network

$\theta' = \tau\theta + (1-\tau)\theta'$

s = s'

until done

until converged

4. Experimental Results

The proposed system is deployed in two different maps Manhattan32 and Urban50 using Python. Manhattan32 is an urban dense city environment containing 32x32 cells whereas Urban50 is a lesser dense environment with size 50x50 cells. Two UAV agents were deployed to visit and collect the data from the collection points.

The hyperparameters of DDQN are presented in Table 2.

Parameter	**Value**
Boundary penalty	1.0

Empty battery penalty	250.0
Movement penalty	0.1
Batch size	128

Table 2 DDQN Hyperparameters

The common simulation parameters are shown in Table 3.

Parameter	**Value**
Number of UAVs	2
Number of sensor nodes	20 to 100
Altitude	10m
Cell size	10m
Time slots	4

Table 3 Common parameters

Table 4 shows the simulation parameters of Manhattan 32 and Urban50 maps.

Parameter	**Map type**	
	Manhattan32	**Urban50**
cell_edge_snr	-25	-30
path_loss_exp	227	227
los_shadowing_variance	2.0	2.0
Movement range	50 to 150m	100 to 200m

Table 4 Parameters for the given maps

4.1 Performance metrics

The performance of ODDQN is compared to the Fuzzy DRL [10] and UAVDCH [11] methods, in terms of the following metrics:

Landing Success percentage: Records the successful landing of agents at the end of an episode

Data Collection ratio: It is the ratio of total collected data at the end of the mission to the total device data that was available at the beginning of the mission.

Data Collection delay: It indicates the total time taken for an agent to collect the data from a visiting point.

Packet drop rate: It indicates the rate of packets dropped over the specified time interval.

Average Residual energy: It is the average remaining battery energies of all the sensors at the end of data transmission.

The comparison results are presented in the next section.

4.2 Results

In this section, the number of sensors transmitting data to the UAV is varied from 20 to 100.

(i) Data Collection Delay

Table 5 and Figure 3 show the results of data collection delay, when the nodes are varied.

Number of Sensors	**UAVDCH (sec)**	**Fuzzy-DRL (sec)**	**ODDQN (sec)**
20	0.177	0.191	0.112
40	0.195	0.228	0.138

60	0.214	0.264	0.165
80	0.258	0.287	0.185
100	0.276	0.316	0.226

Table 5 Results of data collection delay

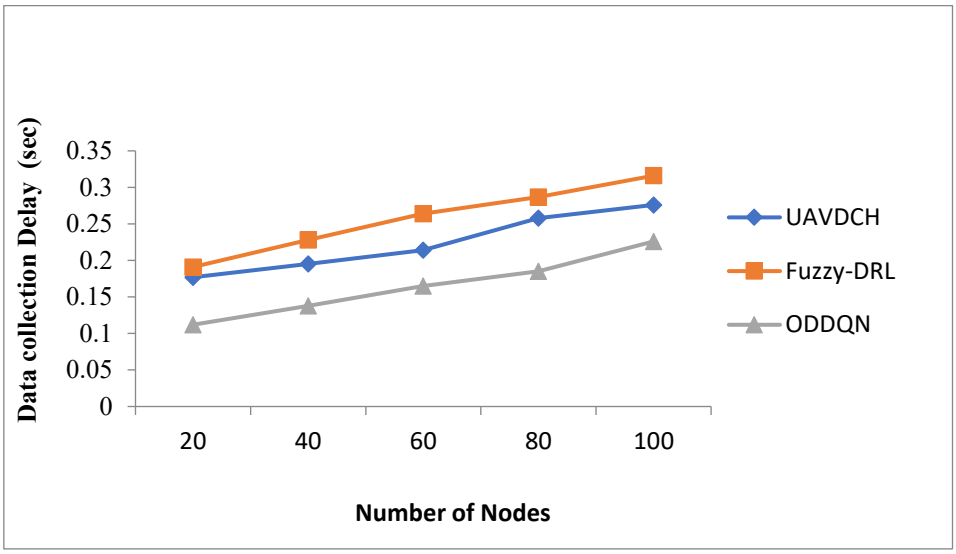

Figure 3 Data collection delay for various nodes

From Table 5 and Figure 3, we can observe that the data collection delay of ODDQN is 27% lesser than UAVDCH and 36% lesser than Fuzzy-DRL

Key Insights & Trends

- **Direct Proportional Trend**: All three methods experience a steady, near-linear increase in data collection delay as more nodes are introduced to the network.
- **Best Performer:** The **ODDQN** algorithm consistently maintains the lowest data collection delay at every single node count, showcasing superior efficiency.
- **Worst Performer**: The **Fuzzy-DRL** approach experiences the highest delay across the entire evaluation spectrum, peaking at over 0.31 seconds at 100 nodes.
- **Middle Baseline: UAVDCH** outperforms Fuzzy-DRL but falls short of ODDQN, acting as a middle-tier baseline algorithm.

(ii) **Data Collection ratio**

Table 6 and Figure 4 show the results of data collection ratio, when the nodes are varied.

Number of Sensors	**UAVDCH**	**Fuzzy-DRL**	**ODDQN**
20	0.9631	0.9561	0.9817
40	0.9583	0.9519	0.9783
60	0.9525	0.9479	0.9716
80	0.9502	0.9437	0.9642
100	0.9481	0.9408	0.9575

Table 6 Results of data collection ratio

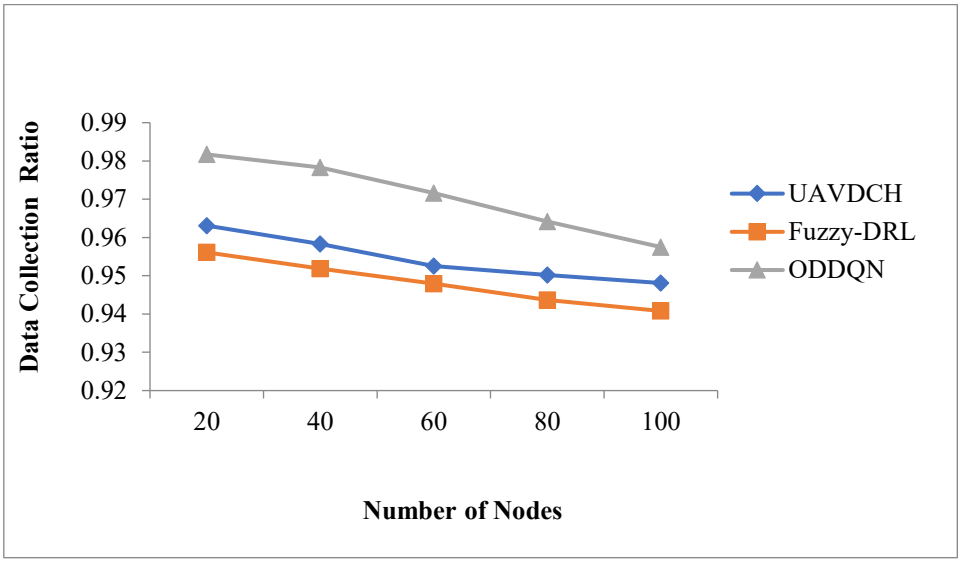

Figure 4 Data collection ratio for various nodes

From Table 6 and Figure 4, we can observe that the data collection ratio of ODDQN is 1.6% higher than UAVDCH and 2.3 higher than Fuzzy-DRL.

Key Insights & Trends

- **Inverse Proportional Trend**: All three methods show a steady decrease in data collection ratio as the number of network nodes increases.
- **Best Performer**: The **ODDQN** algorithm consistently delivers the highest data collection ratio, staying above 95.5% even at full network capacity.
- **Worst Performer**: The **Fuzzy-DRL** algorithm struggles the most, maintaining the lowest success ratio across all tested node counts.
- **Middle Baseline**: **UAVDCH** closely trails ODDQN initially but experiences a slower rate of decline after 60 nodes, stabilizing near 94.8%.

(iii) Packet Drop Rate

Table 7 and Figure 5 show the results of packet drop rate, when the nodes are varied.

Number of Sensors	UAVDCH (%)	Fuzzy-DRL (%)	ODDQN (%)
20	14.85	15.33	13.14
40	15.15	15.54	13.48
60	15.77	16.25	13.88
80	16.38	16.72	14.56
100	16.91	17.45	14.87

Table 7 Results of Packet drop rate

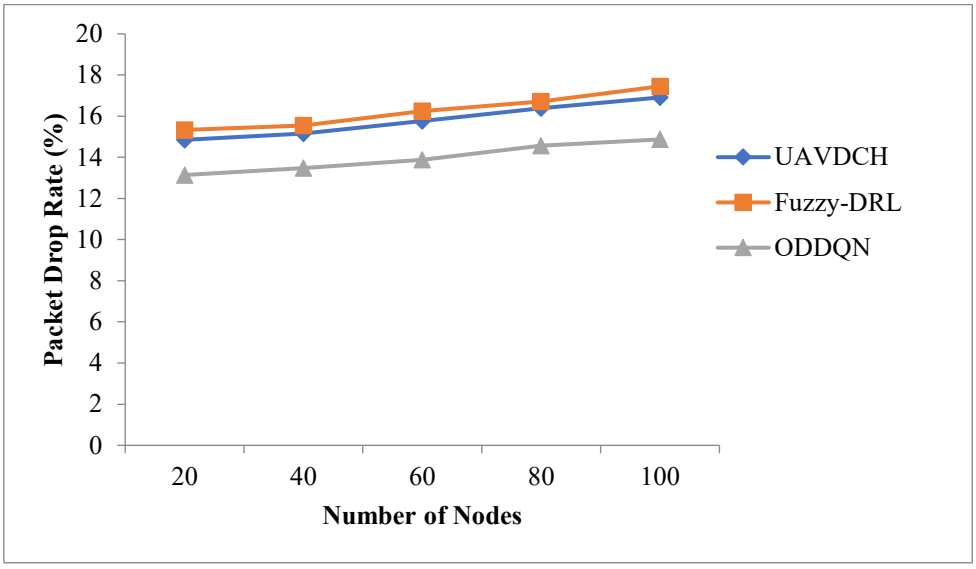

Figure 5 Packet drop rate for various nodes

From Table 7 and Figure 5, we can observe that the packet drop rate of ODDQN is 11% lesser than UAVDCH and 14% lesser than Fuzzy-DRL

Key Insights & Trends

- **Upward Performance Strain**: All three routing algorithms experience a steady increase in packet loss as network density scales from 20 to 100 nodes due to increased network congestion and collision risks.
- **Best Performer**: **ODDQN** consistently achieves the lowest packet drop rate across all configurations, maintaining a clear separation from the other two curves and keeping packet loss below 15%.
- **Worst Performer**: The **Fuzzy-DRL** algorithm shows the highest packet loss throughout the evaluation, reaching a maximum drop rate of approximately 17.5% at 100 nodes.
- **Close Competitors**: The performance curves for **UAVDCH** and **Fuzzy-DRL** track very closely together across all data points, indicating similar performance limitations under dense network environments.

(iv) Average Residual Energy

Table 8 and Figure 6 show the results of average residual energy, when the nodes are varied.

Number of Sensors	UAVDCH (%)	Fuzzy-DRL (%)	ODDQN (%)
20	20.16	19.88	20.87
40	19.82	19.61	20.45
60	19.45	19.35	20.28
80	19.14	18.77	19.72
100	18.82	18.43	19.53

Table 8 Results of Residual energy

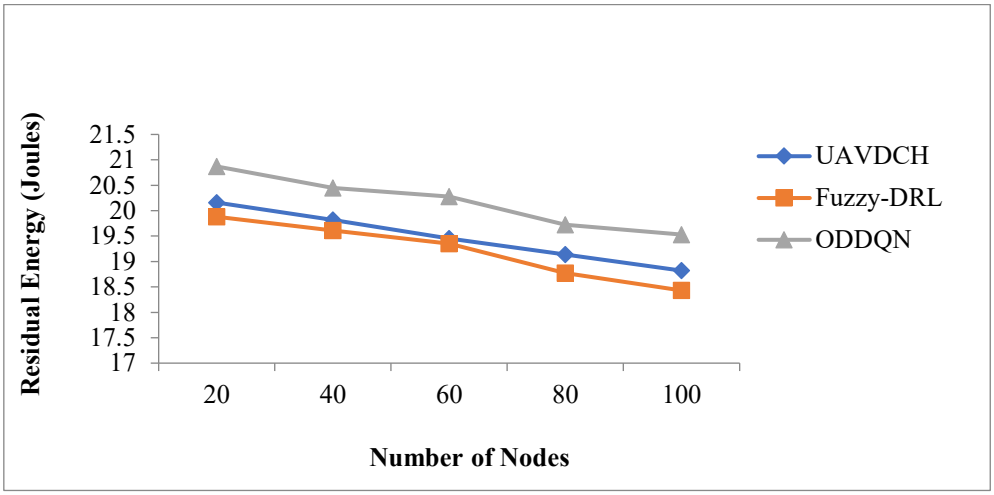

Figure 6 Residual energy for various nodes

From Table 8 and Figure 6, we can observe that the residual energy of ODDQN is 3.4% higher than UAVDCH and 4.7 higher than Fuzzy-DRL.

Key Insights & Trends

- **Downward Energy Trend**: Every algorithm shows a steady decline in residual energy as the network size scales up. More nodes generate higher traffic and routing overhead, causing nodes to consume more power and leaving less energy remaining.
- **Best Performer**: **ODDQN** maintains the highest residual energy across all test scenarios. It preserves approximately 19.5 Joules even at 100 nodes, showing strong energy efficiency.
- **Worst Performer**: **Fuzzy-DRL** finishes with the lowest residual energy (~18.4 Joules at 100 nodes), indicating higher power consumption during network operations.
- **Convergence at 60 Nodes**: The performance of **UAVDCH** and **Fuzzy-DRL** comes very close at 60 nodes (~19.45 J vs ~19.35 J) before widening again at higher node count

5. Conclusion

This paper designs a DDQN model to solve the problem of determining optimal data collection schedules for UAV sensors. The DDQN parameters are further optimized by applying backpropagation technique. The reward function is formed to maximize data freshness while minimizing operational costs. The proposed system is deployed in two different maps Manhattan32 and Urban50 using Python. The performance of ODDQN is compared to the Fuzzy DRL and UAVDCH methods. Experimental results show that the ODDQN model attains maximum data collection ratio with reduced data collection delay and packet drop rate.

References

1. Zhang, H. Liu, Z. Ma, Y. Yang and X. Wan, "Study of UAV Application in Wireless Sensor Networks," 2020 3rd International Conference on Mechanical, Electronics, Computer, and Industrial Technology (MECnIT), Medan, Indonesia, 2020, pp. 343–348, doi: 10.1109/MECnIT48290.2020.9166681.
2. Z. Wei et al., "UAV-Assisted Data Collection for Internet of Things: A Survey," IEEE Internet of Things Journal, vol. 9, no. 17, pp. 15460–15483, Sept. 2022, doi: 10.1109/JIOT.2022.3176903.
3. Y. Du, J. Hao, Z. Chen and Y. Guo, "Indeterministic Data Collection in UAV-Assisted Wide and Sparse Wireless Sensor Network," Sensors, vol. 24, 2024, Art. no. 6496, doi: 10.3390/s24196496.
4. M. Chen, Data Collection Optimization for UAV-Assisted Wireless Sensor Networks, Ph.D. dissertation, Australian National Univ., Canberra, Australia, May 2023.
5. Y. Lu, Y. Hong, C. Luo, D. Li and Z. Chen, "Optimization Algorithms for UAV-and-MUV Cooperative Data Collection in Wireless Sensor Networks," Drones, vol. 7, no. 408, 2023, doi: 10.3390/drones7070408.
6. Wang, F. Ma, J. Yan, D. De and S. K. Das, "Efficient Aerial Data Collection with UAV in Large-Scale Wireless Sensor Networks," International Journal of Distributed Sensor Networks, vol. 2015, Art. ID 286080, 2015.
7. J. Chen and J. Tang, "UAV-Assisted Data Collection for Wireless Sensor Networks With Dynamic Working Modes," Digital Communications and Networks, vol. 10, pp. 805–812, 2024.
8. Sun, X. Zheng, J. Li, H. Kang and S. Liang, "Collaborative WSN–UAV Data Collection in Smart Agriculture: A Bi-objective Optimization Scheme," ACM Transactions on Sensor Networks, 2023.
9. J. J. Sadique, M. R. Khan and A. S. Ibrahim, "UAV-Aided Fast Data Collection via Machine Learning Using AERPAW's Digital Twin," in Proc. WiNTECH '25, Hong Kong, China, 2025.

10. P. A. Karegar, D. Z. Al-Hamid and P. H. J. Chong, "Deep Reinforcement Learning for UAV-Based SDWSN Data Collection," Future Internet, vol. 16, no. 398, 2024, doi: 10.3390/fi16110398.
11. K. Soltani, F. Corò and S. K. Das, "Optimizing UAV-Assisted Data Collection in IoT Sensor Networks Using Dual Cluster Head Strategy," in 2024 IEEE 21st International Conference on Mobile Ad-Hoc and Smart Systems (MASS), 2024.
12. X. Ma, T. Liu, S. Liu, R. Kacimi and R. Dhaou, "Priority-Based Data Collection for UAV-Aided Mobile Sensor Network," Sensors, vol. 20, no. 3034, 2020, doi: 10.3390/s20113034.