Future Generation Computer Systems 161 (2024) 454–466

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

MURE: Multi-layer real-time livestock management architecture with unmanned aerial vehicles using deep reinforcement learning

Xinyu Tian [a], Mahbuba Afrin [b,*], Sajib Mistry [b], Redowan Mahmud [b], Aneesh Krishna [b], Yan Li [c]

[a] Australian National University, Canberra, 2601, ACT, Australia
[b] School of Electrical Engineering, Computing and Mathematical Sciences, Curtin University, Perth, 6102, WA, Australia
[c] School of Mathematics, Physics and Computing, University of Southern Queensland, 4350, QLD, Australia

ARTICLE INFO

Keywords:
Precision livestock management
Unmanned aerial vehicle
Deep reinforcement learning
Wireless sensor network
Internet of Things

ABSTRACT

In recent years, the combination of unmanned aerial vehicles (UAVs) and wireless sensor networks (WSNs) has gained popularity in livestock management (LM) due to energy constraints and network instability. Limited energy storage of sensor nodes (SNs) and the possibility of packet loss contribute to fast energy consumption and unstable networks, respectively. UAVs serve as relay nodes and data sinks, addressing these issues by temporarily storing data to reduce SN workload and establishing mobile nodes for network stability. We propose two innovations based on previous work: 1) We introduce a multi-layer wireless network architecture, categorizing UAVs into two layers based on their functions including data collection and data processing. This enhances task parallelization, bridging performance gaps among multiple UAVs; 2) We overcome the mobility limitation of SNs, considering their real-time movement in the network. Through deep reinforcement learning, UAVs learn to cooperatively locate moving SNs. This accounts for the inevitable mobility of livestock in the industry. Additionally, we simulate the environment and compare our approach to traditional methods, evaluating metrics such as collected data per timestep (DCPS), energy consumed per timestep (ECPS), and network stability (NS). Experimental results demonstrate that our method outperforms traditional approaches, achieving a data collecting gain of 4.84% and 8.20% compared to the methods without considering SN mobility or the multi-layer characteristics of WSNs, respectively. Under energy consumption limits, our method yields energy savings of 3.00% and 1.35% respectively. Furthermore, we extensively study and validate our method against other path planning algorithms, including genetic particle swarm optimization (GPSO), modified central force optimization (MCFO), and rapidly-exploring random trees (RRT). Our approach surpasses these methods in terms of data collecting efficiency and network stability.

1. Introduction

Livestock management (LM) has become crucial and promising in the industry due to population growth and changing topography worldwide [1]. The prevalence of the Internet of Things (IoT) in LM has gained traction, as it efficiently connects devices and monitors, reducing farmers' workload. IoT enables automatic sensing of livestock health and reporting anomalies on cloud servers [2,3]. These portable IoT devices, such as wearables or embedded sensors, collect health-related data like body temperature, humidity, and heart rate, determining necessary actions. These devices are distributed as sensor nodes (SNs) in a wireless sensor network (WSN) across the farm. However, SNs have limitations. Firstly, their small size and lightweight design for portability result in low energy storage capacity [4]. Consequently, frequent replacement of SNs is necessary. Secondly, the cost of data collection from SNs is relatively high, highlighting the need for an automated pipeline for data collection and analysis [5]. Thus, the rapid development of unmanned aerial vehicles (UAVs) has inspired the use of mobile IoT devices to assist SN management and enable smart decision-making.

UAVs have diverse applications across industries. For example, in data communication, UAVs act as communication nodes to connect separate WSN [6]. Equipped with cameras, UAVs capture images from various angles to reconstruct 3D scenes [7]. In the field of LM, UAVs serve multiple purposes. They can: (1) act as data sinks and transmit collected data to cloud servers [8], (2) preprocess and aggregate data from different SNs [9,10], and (3) provide real-time farm monitoring by periodically capturing frames of the livestock [11].

In this paper, we propose a multi-layer real-time architecture comprising SNs and UAVs, each with distinct roles. We assume that multiple

* Corresponding author.
E-mail address: mahbuba.afrin@curtin.edu.au (M. Afrin).

https://doi.org/10.1016/j.future.2024.07.038
Received 10 July 2023; Received in revised form 16 July 2024; Accepted 20 July 2024
Available online 23 July 2024

Table 1
Overview of related work in LM.

Author	Task	Methodology
Bouchekara et al. [14]	Livestock monitoring	A classification task is proposed on the images taken by UAVs to distinguish livestock.
Li et al. [15]	Livestock monitoring	Establish a point cloud system using 3d reconstruction to estimate the size and weight of livestock.
Awan et al. [16]	Communication & Routing	A nature-inspired cluster-based routing considering the activity of herd is proposed to minimize the energy consumption in WSNs.
Alanezi et al. [10]	Communication & Routing	Based on the existing cluster-based routing method, a fault detection and recovery mechanism is proposed to reduce the risk of connection failure in WSNs.
Behjati et al. [17]	UAV path planning	A genetic PSO algorithm is proposed to iteratively find the shortest path across all SNs (Hamiltonian path).
Pan et al. [18]	UAV path planning	A deep learning based genetic algorithm is proposed to find the optimal path for multiple UAVs.
Salehi et al. [12]	UAV cooperation	A decision-making mechanism depending on different situations is proposed to find out the minimum number of UAVs required satisfying the specified parameters
Dineva and Atanasova [8]	Interaction & Visualization	A pipeline processing and analyzing data to interactively provide information to users is proposed

UAVs may vary in parameters and performance. Hence, we implement hierarchical management of UAVs based on their data storage capacity and processing capability. This approach involves developing different types of UAVs for data collection and aggregation, as well as data preprocessing and relaying. We believe that this hierarchical management strategy will effectively leverage the strengths of individual UAVs and parallelize various stages in the pipeline. While previous work focuses on structural deployment and partitioning of SNs, such as cluster-based approaches [9,10], we emphasize that functional division of UAVs would be more efficient than deploying them uniformly. Our experimental results demonstrate that our method outperforms traditional approaches in terms of data collection effectiveness and energy efficiency.

Moreover, existing research often assumes static SNs [9,12] or imposes strong restrictions on their movement [13], which is impractical in LM. In reality, livestock follow a random walk model as they roam across pastures, influenced by water resources and grassland density. In our work, we account for livestock movement by updating their coordinates in real-time. We propose a cluster-based partition using stream K-means. At each timestep, we evenly distribute the entire pasture area among UAVs based on the current SN locations, considering the stability of this partition over time. In other words, SNs can dynamically switch between UAVs as they move. To enable efficient path planning in this dynamic scenario, we employ deep reinforcement learning for UAVs. Our approach liberates SNs from fixed partitions in the cluster-based method, allowing them to collect data efficiently under any walk model.

The main contribution of this paper are summarized as:

- To enhance efficiency, we introduce a multi-layer architecture utilizing ad-hoc UAVs tailored to their specific functions. Unlike conventional UAVs that sequentially perform distinct tasks, our architecture enables parallelization of data collection and processing, resulting in improved efficiency. Moreover, this practical approach acknowledges the performance disparities often observed among multiple UAVs in the industry, in contrast to the assumption of ideal uniformity.
- We address real-time mobility of SNs, surpassing previous limitations and restrictions on node movement. At each timestep, we update the area partition using stream K-means, guaranteeing optimal node allocation efficiency. By maintaining stable area partitions, UAVs can avoid overlapping or overloading their coverage, even when nodes move arbitrarily.
- Through simulation, we evaluate and quantitatively compare our method with traditional approaches. The evaluation encompasses data collection effectiveness, energy efficiency, and network stability. Our experimental results demonstrate significant improvements provided by our method. Specifically, compared to methods that do not consider SN mobility and the multi-layer UAV architecture, our method achieves a data collection gain of 4.84% and 8.20%, as well as an energy saving gain of 3.00% and 1.35% respectively. Furthermore, our visualization of network stability reveals increased operational longevity without failures as the number of learning episodes increases, highlighting the enhanced robustness of our method in comparison to others.
- To validate our method, we compare it with three popular path planning strategies: genetic particle swarm optimization (GPSO) [17], modified central force optimization (MCFO) [19], and rapidly-exploring random trees (RRT) [20]. With a consistent setting, we observe significant improvements in data collection gain and network stability compared to these methods. Specifically, our method achieves a data collection gain of 12.53%, 9.44%, and 16.84%, as well as a network stability gain of 1.63%, 0.37%, and 6.31% respectively, surpassing the aforementioned methods. Furthermore, our method demonstrates lower energy consumption, ranking second only to GPSO.

Section 2 provides a detailed analysis of the related work. In Section 3, we present our proposed multi-layer architecture, which encompasses the overall definition and energy model. Section 4 provides our contribution to the improvement of the system including cluster configuration and evaluation metrics. The experimental settings and validation of our approach are discussed in Section 5, where we compare it to previous methods. Finally, in Section 6, we conclude our proposed methods and highlight their contributions.

2. Related work

2.1. UAVs for livestock management

Currently, livestock management (LM) can be categorized into two types: traditional LM (TLM) and precision LM (PLM)[1] [21]. TLM involves manual supervision of farms or labor-intensive farm management practices [22], which is characterized by high costs and demands a high level of management expertise and technical knowledge from farmers.

PLM achieves high productivity through automated management and advanced technology. Electronic tools, particularly communicative sensors, are extensively used in PLM for data collection and transmission [21]. Research efforts in PLM are increasingly focused on

[1] In the following section, we will refer to LM as PLM for brevity.

leveraging diverse electronic and communication devices. Early PLM employed sensors, GPS, and other tools to collect health data from livestock [23]. Technologies like RFID were utilized for livestock identification [24]. However, these methods were insufficient in efficiently collecting data for preprocessing and timely upload. The advent of IoT revealed the effectiveness of WSNs in organizing data and establishing automated pipelines that encompass data collection, aggregation, analysis, and visualization [25].

UAVs have gained popularity in LM as relay nodes, capable of establishing connections across multiple networks and storing/processing data [26]. The research encompasses various areas such as battery life, collision avoidance, navigation and control systems, and communication technology.

The battery life of a UAV plays a crucial role in determining its operational capabilities [27]. In LM, the energy consumption of UAVs is influenced by factors such as UAV movement, information transmission frequency, and the number of SNs in WSNs [28]. While optimizing energy allocation through efficient network design and path planning can be helpful, certain fundamental energy consumption, including circuit dissipation and information transmission, cannot be completely eliminated. Recent research has focused not only on maximizing UAV battery performance [29], but also on exploring energy consumption optimization for UAVs in WSNs [30].

In LM, the collision avoidance capability of UAVs is typically considered to be robust due to the absence of high-altitude structures or obstacles on farms. UAVs are often operated at a fixed elevation, allowing for maximized communication range [31]. However, it is important to acknowledge that collision avoidance remains a critical factor, and efforts have been made to enhance UAV robustness in LM [32]. Early UAV navigation primarily relied on GPS technology, yet its limitations, such as indoor inoperability, have been recognized [33]. Alternative navigation techniques, including inertial navigation or terrain aided navigation, have their own drawbacks, such as reduced accuracy or susceptibility to extreme weather conditions [34]. In LM scenarios, it is commonly believed that UAV navigation and control are confined within a limited range of latitude, longitude, and altitude, allowing for the utilization of GPS and other methods for more precise localization [35].

Recent research efforts have focused on developing automatic control algorithms and path planning techniques to enhance the overall efficiency of the network system [36]. The communication of UAVs in LM is flexible yet susceptible to failures due to their swift movement [37]. This characteristic becomes more prominent in multi-UAV and large-scale networks. To enhance network robustness, recent research has focused on minimizing packet loss and improving fault tolerance through information encoding and decoding techniques [10,38]. Table 1 provides an overview of the recent work.

2.2. Reinforcement learning

Reinforcement learning (RL) involves an agent learning to decide actions within an environment based on received rewards. Representative RL algorithms encompass Q-value network optimization [39–41] and policy gradient optimization [42,43], each based on parameterized functions. Q-value network optimization estimates the total expected reward given the current state and action, selecting the action with the highest expected reward during inference. Policy gradient optimization directly approximates the probability function of the agent's actions given a state. Despite the promise of RL, a major challenge is delayed feedback [44,45], where rewards are received only at the end of an episode, leading to high time and training costs. Temporal difference learning [46] addresses this by approximating the ground truth reward, enabling updates at each step.

In LM, system optimization aims to discover the most efficient interaction mode among various IoT devices by optimizing path finding and communication strategies in an assumed environment, echoing the core motivation of RL. Early attempts in this field come from [47], which proposes a sense-and-send protocol using reinforcement learning to address the decentralized UAV pathfinding problem by optimizing sensing and transmission efficiency together. [48] advances this by introducing an interference-aware path planning scheme for cellular UAV networks, balancing efficiency and effectiveness. [49] presents a generalizable online reinforcement learning framework tailored for large-scale, complex environments by converting UAV's raw sensory measurements into control signals. [13] brings reinforcement learning in UAVs to the next level by solving path planning for 3d deployment and dynamic movement of multiple UAVs. Recently, [50] introduces a distributed Q-learning algorithm for multi-agent scenarios, surpassing benchmarks in energy efficiency and stability. Concurrently, [51] presents the UAV-MEC network, aiming to establish proxy connection between mobile devices and distant base stations.

However, a common assumption in prior work is that UAVs are equally divided and deployed, performing assigned tasks in a sequential manner. This work proposes a novel hierarchy by assigning distinct tasks to different UAV layers, parallelizing the system, and significantly improving energy efficiency and network stability.

3. System model

3.1. System assumptions

Consider in a fixed size farm, we establish a WSN with $\mathcal{N}$ nodes randomly distributed in the farm and every node is denoted as $s_i \in \{s_1, s_2, \dots, s_{\mathcal{N}}\}$. To evaluate the sustainability of the system, we define that each node has identical and limited buffers to store sensed data from livestock. The buffer capacity is denoted as $\mathcal{B}$, and at timestep t, the stored data in the buffer of s_i is denoted as b_i. Therefore the system should satisfy the inequality below

$$\forall\ i \in [1, \mathcal{N}],\ \ b_i(t) \le \mathcal{B}. \tag{1}$$

In terms of the system evaluation, we say the system is sustainable if for a pre-defined threshold $\mathcal{T}$,

$$\forall\ t \le \mathcal{T} \quad \forall\ i \in [1, \mathcal{N}],\ \ b_i(t) \le \mathcal{B}. \tag{2}$$

This will be used to determine if the episode should be terminated or not in DQN. In other words, an episode will be over if any of the nodes exceeds the storage of buffers. Whenever this happens, we say the buffer of the node expires and some information will be lost. The details will be discussed in Section 4.1. Besides, we define the location of the node s_i at timestep t as $x_i(t) \in \mathbb{R}^2$. As we consider the mobility of nodes for each timestep, we perform a random walk model on each node. For simplicity, we define a parameter $Pr \in [0, 1]$, where for each timestep, the node s_i has a probability of Pr to move towards a random direction with a pre-defined velocity $v_i(t)$. Note that in this case $v_i(t)$ is a randomly generated 2d vector with a constant magnitude. Therefore we have

$$x_i(t+1) = \begin{cases} x_i(t) + v_i(t), & \sigma \le Pr, \\ x_i(t), & \sigma > Pr. \end{cases} \tag{3}$$

where σ is a uniformly generated random number ranged in $[0, 1]$.

This random walk model will ensure that when $t \to \infty$, the location of the node $x_i(t)$ is independent of the initial location $x_i(0)$, which means UAVs should find their best way to distribute and approach these nodes despite their locations in our method.

Now we consider adding UAVs into our environment. Specifically, we consider $\mathcal{K}$ UAVs are deployed to collect data from nodes and name the set of UAVs Type I. Each UAV of Type I is denoted as $u_i \in \{u_1, u_2, \dots, u_{\mathcal{K}}\}$. We will give the initial deployment of UAVs as the cluster centers by performing K-means regarding the initial location of nodes. If we denote the location of UAV u_i at timestep t as $x_i^u(t)$, then we have

$$x_i^u(0) = c_i(0), \tag{4}$$

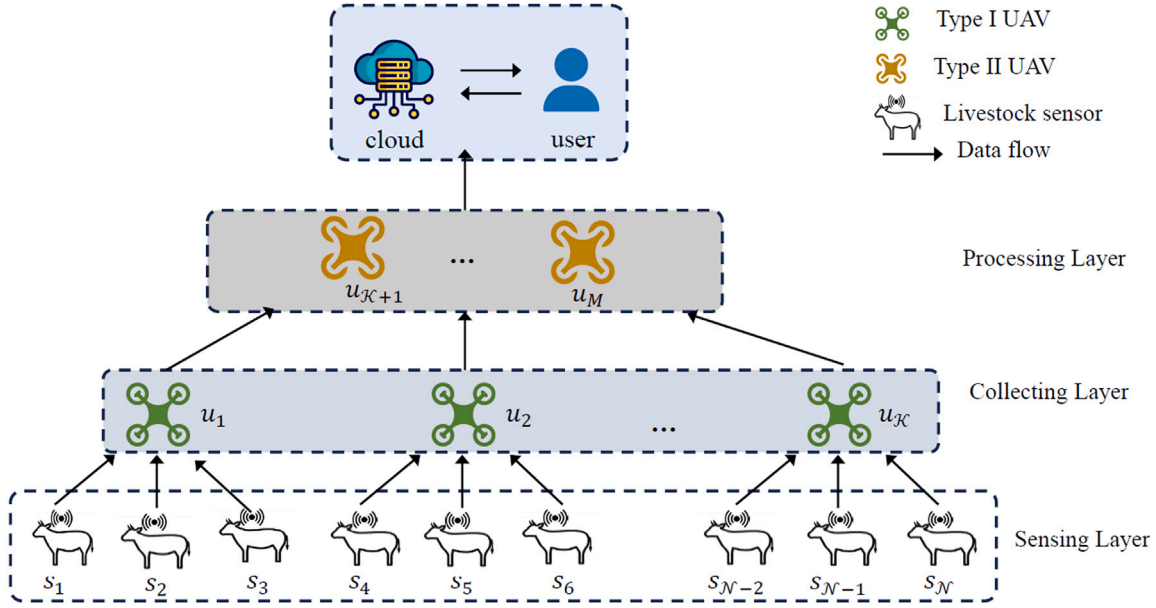


Fig. 1. Multi-layer architecture of WSNs.

where c_i is the center of the cluster C_i. In this context, each cluster C_i is denoted as a set of SNs partitioned by clustering algorithms, and every cluster will be assigned to a single UAV of Type I. The configuration and update of the cluster will be introduced in Section 4.4.

We consider another type of UAVs. We denote the UAVs we have added to collect data from nodes as Type I, similarly, the other type of UAVs defined is called Type II. We denote each UAV of Type II as $u_i \in \{u_{\mathcal{K}+1}, u_{\mathcal{K}+2}, \dots, u_{\mathcal{K}+\mathcal{M}}\}$, where $\mathcal{M} \ll \mathcal{K}$, and typically $\mathcal{M} = 1$. The UAVs in Type I will visit the nodes in their vicinity, and attempt to relay the raw data to UAVs in Type II. That is, UAVs in Type II will mainly focus on preprocessing raw data from livestock, then transferring processed data to the cloud server.

Here we further clarify the difference between Type I and Type II. Generally, UAVs in WSNs will play multiple roles including relay nodes, computational nodes, and sometimes storage sink. We notice that this requires relatively high comprehensive performance of UAVs, and it is difficult for a single UAV to complete multiple tasks sequentially in a short time. Therefore, we propose to divide UAVs into two categories, one with high storage space and the other with strong data processing capability, namely Type I and Type II. This approach allows for the parallelization of both data collecting and data processing and takes advantage of the performance benefits of different UAVs, which is more practical in the industry. More details of this multi-layer architecture will be discussed in Section 3.2.

Without losing generalization, we consider a simple signal channel model to estimate the optimal altitude of UAVs for the maximum coverage radius of UAVs in the farm. Following Al-Hourani et al. [52], we consider two typical propagation groups $\xi \in \{\text{LoS}, \text{NLoS}\}$. We denote θ as the elevation angle between the UAV and the ground node, therefore we have the probability of line-of-sight (LoS) and non-line-of-sight (NLoS) respectively

$$P_{LoS}(\theta) = (1 + a\exp(-b\,[\theta - a]))^{-1}, \tag{5}$$

$$P_{NLoS}(\theta) = 1 - P_{LoS}(\theta), \tag{6}$$

where a and b are two constants affected by the environment such as the mean height of the buildings around. To maximize the coverage radius $\mathcal{R}$, similar to Al-Hourani et al. [52], we pre-define the maximum allowed path loss PL_{max} and set

$$PL_{max} = P_{LoS} \times PL_{LoS} + P_{NLoS} \times PL_{NLoS}, \tag{7}$$

where PL_{LoS} and PL_{NLoS} are the path loss of LoS and NLoS respectively. That is, we determine the allowed path loss as the spatial expectation value of two propagation groups. Combining the free space path loss with the additive loss incurred by man-made structures, we have

$$PL_{LoS} = FSPL + \eta_{LoS}, \tag{8}$$

$$PL_{NLoS} = FSPL + \eta_{NLoS}. \tag{9}$$

Finally, substitute these path loss in terms of the coverage radius and the altitude, we can formulate an optimization problem

$$\begin{aligned} &\max\ \mathcal{R} \\ &s.t.\quad PL_{max} = A(1 + a\exp(-b\left[\arctan(\tfrac{h}{\mathcal{R}}) - a\right]))^{-1} \\ &\qquad\qquad + 10\log(h^2 + \mathcal{R}^2) + B, \end{aligned} \tag{10}$$

where A and B are in terms of η and environment factors, h and $\mathcal{R}$ are the altitude and radius, respectively. By implicit differentiation, the value of h satisfying $\frac{\partial \mathcal{R}}{\partial h} \approx 0$ is considered an acceptable solution and used in the experiment.

3.2. Multi-layer architecture

In this section, we propose a multi-layer architecture as depicted in Fig. 1 based on the assumption in Section 3.1. We parallelize the two tasks by assigning data collecting and data processing to different types of UAVs and take advantage of the different performance benefits of the UAVs.

In order to effectively distinguish and utilize the characteristics of different categories of UAVs, we quantitatively evaluate their data capacity, transmission capability, and processing capability. We denote the maximum load of certain UAV u_i as l_i, then the data collecting capability (DCC) regarding l_i is determined as

$$DCC_i = \frac{l_i}{\max l}. \tag{11}$$

Now we consider a simple signal transmission model defined in Friis [53]. The relationship of the transmission power between the sender and the receiver could be defined as

$$P_r = \frac{P_t A_r A_t}{d^2 \lambda}, \tag{12}$$

where P_r is the received signal power, P_t is the transmission power from the sender. A_r and A_t are the effective area of the antenna from the receiver and sender respectively. d is the distance between and λ is the wave length. Therefore, given a fixed distance d_0, we want to evaluate the signal strength emitted by the UAVs. For simplicity, we only consider the transmission power per unit area on the receiver. Therefore we have

$$P_0 = \frac{P_t A_t}{d_0^2 \lambda}. \tag{13}$$

In this scenario, we term P_0 as the data transmission capability (DTC) for a certain u, and thereby d_0 refers to the distance between u and the corresponding receiver.

In addition, we directly refer to the data processing capability as floating-point operations per second (FLOPS), denoted as DPC. We take these three metrics into consideration in the selection of UAVs. Given a pre-defined threshold ρ, we say these UAVs belong to Type I if they satisfy

$$\frac{\beta DCC + (1-\beta) DTC}{\varphi DPC} \geq \rho. \tag{14}$$

where β and φ are pre-defined parameters between 0 and 1 to balance the weight of capabilities. Otherwise, we say they belong to Type II and they should be responsible for pre-processing raw data, converting, and transferring to the cloud. Thus the WSN is organized into a three-layer architecture as depicted in Fig. 1. All the SNs embedded on livestock are grouped into the bottom layer. To cooperatively manage and collect data from these individuals, UAVs in Type I are responsible for approaching assigned SNs and collecting data from them. The main challenge for Type I is to dynamically find the shortest path to visit the nodes in the sub-area without failure. The other type, Type II allows for more computation and processing capability according to the definition above, and is told to pre-process the aggregated data received from Type I and transfer to the cloud. The top layer could be the cloud server, base station, or any other data warehouse to store the data from the bottom for further analysis.

3.3. Energy consumption model

To better evaluate the system from the perspective of energy efficiency, we define a simple energy consumption model based on the assumption in Bouguera et al. [28]. We will explore different phases during the waking-up of sensors and determine the total energy consumption regarding sensors.

Without losing generalization, the total energy consumption E_{tot} until timestep t is defined as

$$E_{tot}(t) = E_{slp}(t) + E_{wk}(t), \tag{15}$$

where $E_{slp}(t)$ and E_{wk} are the energy consumption during sleeping and waking-up mode of nodes. In this scenario, for simplicity, we assume the system is continuous and ignore the charging and sleeping phases. Therefore we have

$$E_{tot}(t) \approx E_{wk}(t), \tag{16}$$

Now we consider four phases sequentially taking place in the system including (1) start-up phase, (2) sensing phase, (3) processing phase, and (4) transmission phase. We assume a constant energy dissipation of waking-up states E_c per timestep, therefore during the start-up phase from time t to $t + \Delta t$, we have

$$E_{sp} = E_c \Delta t. \tag{17}$$

When performing the sensing operation within the nodes, we consider the extra energy consumption E_{adc} for analog-to-digital converting (ADC) and data storage. Therefore the energy consumption in this stage E_s is

$$E_s = (E_c + E_{adc})\Delta t. \tag{18}$$

Data processing by nodes is unlikely mentioned in UAVs, it is an ultra-low cost but not negligible process. For simplicity, we only consider the constant energy dissipation similar to waking-up states, i.e., $E_p = E_c \Delta t$.

The final phase is to transfer collected data to UAVs nearby for further processing. Here we consider the extra energy cost for package sending and receiving and term it as E_{sr}, the energy consumption in this stage is

$$E_{tr} = (E_c + E_{sr})\Delta t. \tag{19}$$

Therefore, the total energy cost during the whole process can be represented as

$$E_{tot} = E_{sp} + E_s + E_p + E_{tr}. \tag{20}$$

4. Problem formulation and proposed solution

4.1. Deep Q network

In this section, we will give the essentials of reinforcement learning including states, actions, and rewards. Because of the sparsity of nodes in contrast to the farm, we will also give some ideas on how to design the reward function in terms of the result and process, namely reward shaping. In fact, the design of the reward function, even the change of the hyperparameters, has a crucial influence on the degree of convergence and performance of the model.

Generally, we have two methods to maximize our rewards in deep reinforcement learning. We can directly model the long-term reward function Q by taking the input as the defined states and output as the Q-value of each action [39]. We say the reward function modeled as a deep neural network is called deep Q network. Another method is to model the policy of the agents, specifically approximate the state value function $V_\pi(s_t) = \mathbb{E}_{\mathcal{A}}\ [Q_\pi(s_t, \mathcal{A})]$ using the neural network. The policy gradient method will be used to update the parameters in the network [54]. Nowadays in LM, deep reinforcement learning is mainly used in the path planning of UAVs on how to design the shortest path to visit the nodes in WSNs [13,55]. The reward is mainly out of two motivations: data collecting rate and energy consumption rate. Both 2d [56] and 3d [13] action space are studied in these work depending on altitude setting. For 3d action space, UAVs should have a dynamic coverage radius affected by the altitude. And for 2d action space, UAVs are typically located at an optimal altitude with maximum coverage prior to the learning.

We consider each UAV in Type I as a separate agent to learn its own network parameters. These networks share the environment and get their own rewards and new states after each timestep. For simplicity, we do not add learnable parameters into Type II. This is because (1) generally UAVs of Type I are controlled by neural nets and have a more deterministic behavior to be easily approached; (2) due to the small number of UAVs of Type I, the efficiency gain by deployment of DQN on Type II is not evident, but consumes extra energy. Specifically, a dynamic shortest path finding algorithm is used in Type II and we do not discuss the details here since GPSO, MCFO and other algorithms are applicable.

States. We define a state $S_j(t)$ for UAV u_j at step t as

$$S_j(t) = \{(x_j^u(t),\ b_i(t),\ x_i(t)) \mid 1 \leq i \leq \mathcal{N}\}, \tag{21}$$

where x_i and b_i are the location and buffer load of the sensor s_i, x_j^u is the current location of the corresponding UAV. That is, we consider returning two types of information by the environment including the current location and data storage of the nodes along with the status of the agent itself. The data storage of each node will inform the agents which node is more imperative and needed to be visited urgently. The location of the current nodes will be the tips to help agents find their own shortest path through the network. The agent's location is crucial for calculating its distance to each node and determining the direction for the next step. Here it is worth mentioning that because

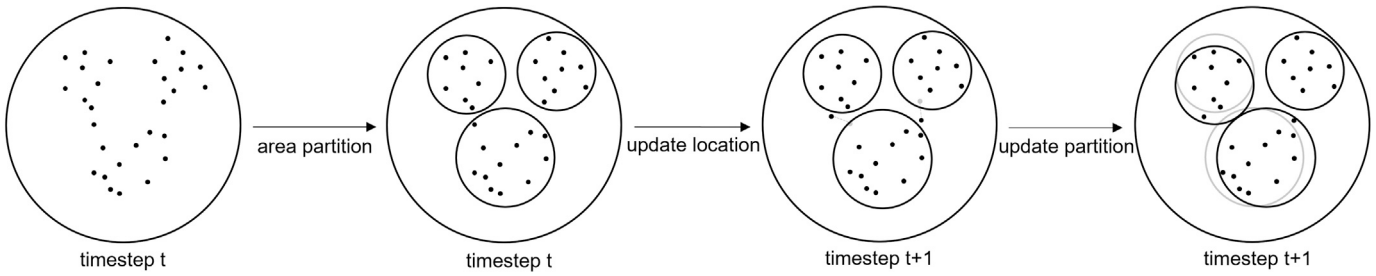


Fig. 2. The cluster update from time t to $t+1$.

of the dynamic assignment due to the mobility of nodes, every node is possibly to be assigned to any agent so that we have to make sure every agent has access to the information of all the nodes.

Action Space. Now we discuss the action space $\mathcal{A}$. For simplicity, we consider a 2d space with a fixed altitude optimized for the maximum coverage radius.

$$\mathcal{A} = \{(-1,0),(1,0),(0,1),(0,-1),(0,0)\}, \tag{22}$$

where $(-1,0)$, $(1,0)$, $(0,1)$, $(0,-1)$, $(0,0)$ refer to *left*, *right*, *forward*, *backward* and *idle*, respectively. In each timestep, the agent u_j will perform one of the actions according to the policy $a_j(t) \sim \pi(\cdot \mid S_j(t))$ and get the reward returned by the environment depending on the current state.

Reward Function. Since the setting of reward function plays a crucial role in reinforcement learning, here we discuss how to design the reward function, namely reward shaping. A simple thought is that we give agents a reward if they successfully communicate with nodes and collect data from them. However, in a large scale farm, nodes are usually sparse in contrast to the whole area, which means this is a sparse reward environment and we have to design some extra rewards to guide agents to complete the task. Therefore we define $r_j(t)$ as the reward function of u_j and let

$$r_j(t) = \begin{cases} -\infty, & x_j^u(t) \nprec \mathcal{L} \\ \lambda\ b_i(t), & \forall\ s_i \in C_j,\ \|x_j^u(t) - x_i(t)\| \le \mathcal{R} \\ -0.1, & \forall\ s_i \in C_j,\ b_i(t) \ge \eta \mathcal{B} \end{cases} \tag{23}$$

where $\mathcal{L}$ is the boundary of the farm, λ is a reward scalar to ensure the reward range between 0 and 1, $\mathcal{R}$ is the maximum coverage radius of the UAV, η is a pre-defined threshold to give the agents a penalty when it is imperative and typically $\eta = 0.75$. In other words, when the load of any nodes exceeds the threshold, we must push the agent to resolve the emergency. Besides, we will strongly restrict the agents to be within the farm by giving an infinite large penalty in case of connection lost and out of control. Whenever the agents approach the nodes under the coverage radius, we will give the reward by the collected data. This prompts agents to look for nodes with a large amount of data that has not yet been collected.

4.2. Problem statement

In this section, we establish the final optimization problem, i.e., Q-value function. Upon formulating an instant reward function r_i that reflects our objectives in Section 4.1, our ultimate aim is to maximize long-term reward function

$$\begin{aligned} &\max\ Q_\pi \\ &s.t.\quad Q_\pi = \mathbb{E}\,[\Sigma_{i=t}^{n} \gamma^{i-t} r_i | S_t = s_t, \mathcal{A}_t = a_t], \end{aligned} \tag{24}$$

where γ is the discount factor and the policy π refers to the probability density function taking action a given the state s. Practically, given the environment, every agent will feedback on the action a to the environment and keeps informed of the corresponding state s and reward r from the environment. Upon deriving Q_π in the current episode, the system maximizes Q_π by determining gradients with respect to the parameters of the deep Q network and updating these parameters using stochastic gradient descent.

In summary, optimizing Q_π indicates that we encourage the model to learn a path finding strategy that (1) covers the signals of assigned nodes to maximize the data collecting effectiveness and (2) is short as much as possible to save system energy. We introduce the metrics to quantitatively evaluate the above attributes in Section 4.3. Since the optimization problem is dependent on the mechanism in UAVs, we also propose an efficient node assignment strategy and the multi-layer architecture to achieve a higher optimal Q_π in Section 4.

4.3. Evaluation metrics

We measure the system in terms of data collecting effectiveness, energy saving efficiency, and network stability. We define the collected data per timestep (CDPS) as the following

$$CDPS = \frac{\Sigma_i b_i}{\mathcal{T}}, \tag{25}$$

where $\mathcal{T}$ is the total timestep in the current episode. CDPS could effectively evaluate how much data the system could collect from the nodes in the unit time. Due to the limitation of the rate at which the nodes collect data, CDPS has a fixed peak $CDPS_{max}$ at which we consider the system to have reached stability.

We use the assumed energy consumption model to calculate the energy consumed by SNs in the system at each timestep. We define the energy consumed per timestep (ECPS) as

$$ECPS = \frac{E_{tot}}{\mathcal{T}}, \tag{26}$$

where E_{tot} is the total energy consumption until $\mathcal{T}$ timesteps. Similarly, we make $\mathcal{T}$ as large as possible to get the value of ECPS when the system is relatively stable. When compared, the system with lower ECPS consumes less energy per unit of time and is preferred.

We measure the stability of a network by how long it lasts without failure. We term network stability (NS) the ratio of the current time that the network can sustain to the maximum time that the network can theoretically sustain. The maximum time that the network can theoretically maintain is determined by the limited energy storage of the SNs and calculated by the current energy consumption strategy.

4.4. Cluster configuration

Traditional LM assumes that SNs are static, thereby a well-designed initial deployment of clusters is sufficient. However, in dynamic scenarios, the changing locations of SNs can lead to UAV coverage overlapping with other clusters, resulting in inefficient transmission. To address this, we propose a new scheme for dynamic node allocation based on the dynamic locations of clusters. Our goal is to ensure that at each step, UAV coverage remains within the range of its assigned cluster, achieving minimum overlap among multiple UAVs.

Algorithm 1 A real-time cluster update algorithm considering the mobility of SNs

Require: Initial clusters $C_i(0), i = 1, 2, ..., \mathcal{K}$, forget rate α, maximum timestep $\mathcal{T}$
Ensure: The clusters $C_i(t)$ at arbitrary timestep $t \leq \mathcal{T}$, $i = 1, 2, ..., \mathcal{K}$

$t = 0$
 # iterate every step
while $t \leq \mathcal{T}$ **do**
  SNs generate new data from livestock
  # check the condition of termination
  if buffer of any SNs is exceeded **then**
    # episode terminated
    break
  end if
  # update the location of SNs
  SNs perform random walk for one timestep
  for all $i = 1, 2, ..., \mathcal{K}$ **do**
    reallocate SNs to cluster centers to get $OC_i(t)$ and $NC_i(t)$
    $RC_i(t+1) = C_i(t) \setminus OC_i(t)$
    determine the cluster centers $rc_i(t+1)$ and $nc_i(t)$ respectively.
    $c_i(t+1) = (\alpha \cdot rc_i(t+1) \left|RC_i(t+1)\right| + nc_i(t) \left|NC_i(t)\right|)/(\alpha \left|RC_i(t+1)\right| + \left|NC_i(t)\right|)$
    $C_i(t+1) = RC_i(t+1) \cup NC_i(t)$
  end for
  # update the location of UAVs
  UAVs perform action given current states
  # if any SNs are in the coverage
  UAVs collect data from SNs if available
  # update parameters of Q-network
  UAVs update policy based on current rewards
  $t = t + 1$
end while

Table 2
Notations and simulation parameters. Note that some system parameters are variables and their values change during the run, indicated as not applicable (–).

Parameter	Description	Value
LEN	The border length of the farm	200 m
UAVs_K	The number of Type I	3
UAVs_M	The number of Type II	1
UAV_V	The velocity of the UAVs	5 m/s
TypeI_R	The coverage radius of Type I	20 m
TypeI_A	The altitude of Type I	35 m
TypeII_R	The coverage radius of Type II	50 m
TypeII_A	The altitude of Type II	46 m
SNs_N	The number of sensor nodes	200
SNs_B	The buffer capacity of SNs	1000 bytes
SNs_CUR_B	The current buffer of SNs	–
SNs_CR	The collection rate of SNs	2.3 bytes/step
SNs_G	The data generation rate of SNs	1.0 bytes/step
SNs_VAR	The data generation variance of SNs	0.25 $bytes^2$/step
SNs_V	The velocity of SNs	2.5 m/s
DCC	The data collecting capability	–
DTC	The data transmission capability	–
DPC	The data processing capability	–
α	The forget rate of the old cluster center	1
Pr	The probability for the SNs to move	0.8
PL_{max}	The maximum path loss allowed	45 dB
PL_{LoS}	The probability of line-of-sight	–
PL_{NLoS}	The probability of non-line-of-sight	–
β	The weight to balance DCC and DTC	0.7
φ	The weight to balance DPC	1
ρ	The threshold to determine UAVs to be Type I or Type II	0.6
E_c	The constant energy consumption for waking-up	10 uJ/step
E_{adc}	The energy consumption for data processing	3 nJ/bit
E_{sr}	The energy consumption for data transferring	1 uJ/bit
LR	The learning rate	0.01
ϵ	The randomness of the greedy policy	0.9
γ	The reward discount factor	0.9
BS	Batch size	32
λ	The reward scalar	0.3

In this section, we introduce a real-time method for tracking the location of nodes and updating the records of the assignment. In Section 3.1 we give the initial assignment of nodes by K-means. Now we consider a set of clusters at time t, $C_i(t) \in \{C_1(t), C_2(t), \ldots, C_{\mathcal{K}}(t)\}$ satisfying $\forall\ i \neq j,\ C_i(t) \cap C_j(t) = \emptyset$.

When we come to $t+1$, we track the mobility of nodes from time t to $t+1$. Here the cluster distribution will constantly change over time. Therefore we discuss two cases: new nodes coming and old nodes leaving. Specifically, in each timestep, we update the assignment of all the nodes in WSNs depending on the proximity. Regarding certain cluster C_i, we say a new node is coming when a node assigned to C_i at time $t+1$ is originally assigned to C_j at time t where $i \neq j$. Similarly, we say an old node is leaving if a node assigned to C_j at time $t+1$ is originally assigned to C_i at time t where $i \neq j$. Hence we further denote the set of new nodes coming regarding cluster C_i at time t as $NC_i(t)$, and the set of old nodes leaving regarding cluster C_i at time t as $OC_i(t)$. Therefore we have

$$C_i(t+1) = C_i(t) \cup NC_i(t) \setminus OC_i(t). \tag{27}$$

Now we consider the update rule of clusters in terms of the location change of nodes in every timestep. In this sense, we will determine the new cluster center affected by the old nodes remaining in the cluster and new nodes coming from other clusters. Therefore we denote the remainder of C_i at time $t+1$ as $RC_i(t+1)$ which is the set of remaining nodes from $C_i(t)$ and have

$$RC_i(t+1) = C_i(t) \setminus OC_i(t). \tag{28}$$

In other words, we consider the process as a stream with new data points introduced in every timestep. We need to update the original distribution towards the new data points. Here we denote the cluster center of $RC_i(t+1)$ and $NC_i(t)$ as $rc_i(t+1)$ and $nc_i(t)$ respectively. Inspired by stream K-means, we can determine the new cluster center $c_i(t+1)$ as

$$c_i(t+1) = \frac{\alpha \cdot rc_i(t+1)\left|RC_i(t+1)\right| + nc_i(t)\left|NC_i(t)\right|}{\alpha\left|RC_i(t+1)\right| + \left|NC_i(t)\right|}, \tag{29}$$

where $|\cdot|$ refers to cardinality of a set and α refers to the forget rate of the remainder. That is, we determine the weighted average of the remainder and the new nodes affected by α. When the fluctuation of nodes is small, that is, the magnitude of velocity is small, we set a large α. Instead, we need a small α to quickly adapt to node location changes. Fig. 2 visualizes the overall update process. Algorithm 1 provides the step-by-step real time cluster update mechanism considering the mobility of the sensor nodes.

5. Experiment

In this section, we will give the details of the implementation and discuss the results by evaluation and comparison.

5.1. Simulation settings

Table 2 presents the key notations and parameters used in the experiment. Notably, the UAV's altitude and coverage radius are determined based on the specified maximum acceptable path loss. This decision involves a trade-off between path loss and coverage radius since excessive path loss requires stronger transmission signals, leading to increased energy consumption, while lower path loss limits the maximum coverage radius. Additionally, in the training process of DQN, we introduce a certain level of randomness by allowing a probability for random sampling from the action space. This randomness enhances the robustness of the neural network.

For data generation, To cope with dynamic and unbalanced data storage in SNs and mimic real-world conditions, we model the data generation rate of each SN at every step as a Gaussian variable. The data generation rate is set to 1 byte/step, with an additional variance of 0.25 bytes^2/step. This approach results in varying data generation amount for each SN, fluctuating between 0.5 and 1.5 bytes/step. Moreover, while Type I enters the sensing range of SNs, the data in their buffer is transmitted at a static rate of 2.3 bytes/step. It is worth mentioning that this transmission does not hinder SNs' data collection, as both processes may run concurrently, resulting in an effective data reduction rate of less than 2.3 bytes/step. We do not consider partial transmission, i.e., each transmission continues until the buffer is empty.

In our simulation environment, we utilize Python to construct the environment and PyTorch to implement the DQN. For Type II, the path planning is performed using GPSO. In the wireless network architecture, we deploy a total of 200 sensor nodes and 4 UAVs. Among the UAVs, three are responsible for data collection from the nodes, while the remaining UAV is tasked with processing the aggregated data. This configuration enables a pipeline-like transmission of data within the multi-layer wireless network architecture, allowing for seamless data flow from bottom to top.

5.2. Result & Comparison

In this section, we present the evaluation results using CDPS, ECPS, and NS metrics. Additionally, we compare our method with two traditional approaches that neglect the mobility of livestock in their random walk model or assume a homogeneous layer of UAVs without considering their distinct roles.

Fig. 3 illustrates the progression of CDPS as DQNs undergo learning. Initially, the CDPS value is nearly 0, signifying that UAVs exhibit random movement due to the absence of reward guidance. As the learning process continues, UAVs gradually acquire the ability to locate and collect data from SNs. After approximately 100 episodes, the CDPS approaches its peak, indicating that the system enters a stable phase.

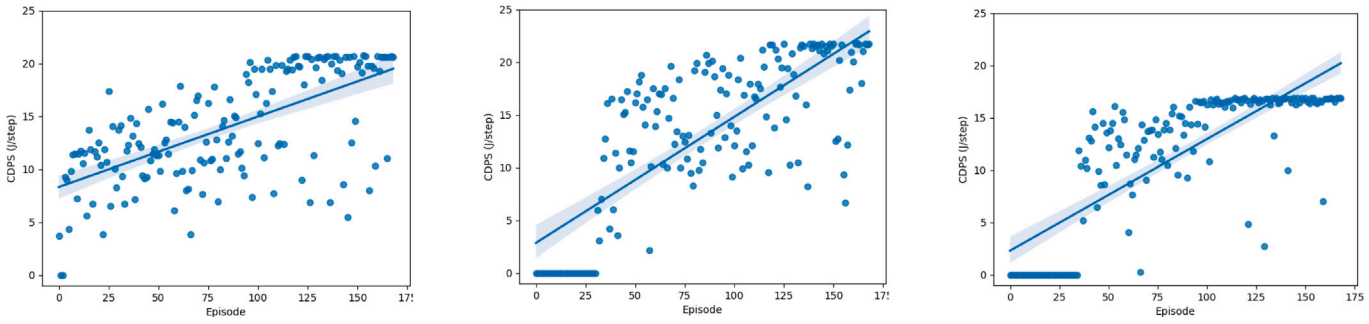
Fig. 4 exhibits a comparable trend, albeit with a distinction in the initial value of ECPS. Unlike CDPS, the ECPS value is not 0 initially, as the system incurs energy consumption from circuit dissipation even when UAVs are in an idle state. With an increasing number of episodes, ECPS gradually reaches its peak value, signifying system stabilization. It is important to note that system stability is a prerequisite for sustainability.

Fig. 5 presents the evolution of NS during the initial 10 episodes following system stabilization. The graph illustrates that NS continues to increase as episodes progress. However, after a brief period, it tends to converge to a value close to 1, indicating system sustainability. In practice, as episodes persist, DQNs approach convergence, implying that UAVs consistently discover the shortest path through all nodes, even when the SNs undergo arbitrary movement.

Next, we compare our approach with two traditional methods. The first method neglects the mobility of SNs, resulting in a single partitioning of the area without considering real-time location changes. The second method treats all UAVs uniformly, assigning them sequential tasks of data collection and processing without differentiation.

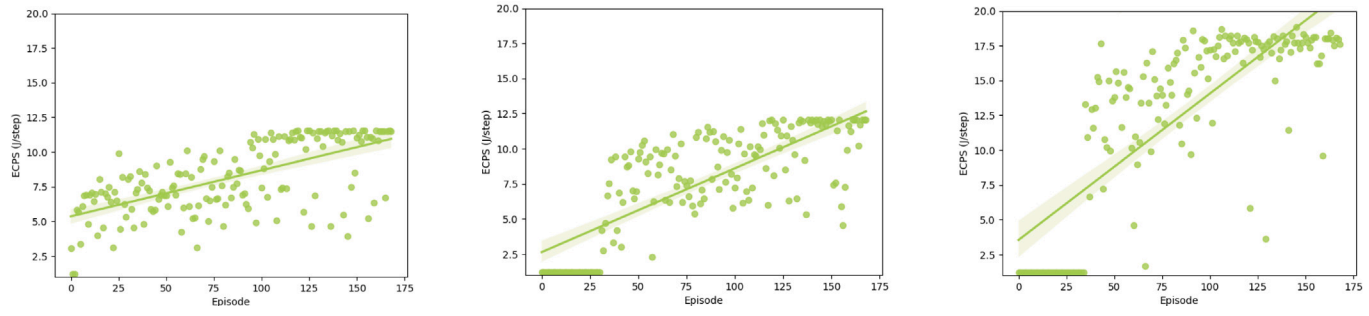
Fig. 6 illustrates the mean NS of all UAVs in the first 10 episodes after system stabilization. Notably, NS shows an increasing trend across all three methods, reaching a plateau shortly after. Besides, our method demonstrates a faster convergence rate compared to the other two methods, and the peak NS value is close to 1, indicating enhanced robustness. In contrast, the method disregarding SN mobility may lead to ineffective UAV movement due to SN location changes, thereby compromising network stability. Similarly, the method with uniform task assignment fails to account for UAV differences, potentially resulting in task failures and data processing delays due to task serialization.

In Fig. 7, the mean values of all UAVs in the steady state provide further evidence supporting our earlier observations. Our method exhibits significant advantages over the two traditional methods in terms of data collecting effectiveness and energy-saving efficiency. Specifically, our method achieves a CDPS gain of 4.84% and 8.20%, as well as an ECPS reduction of 3.00% and 1.35%, when compared to the aforementioned traditional methods, respectively. These results highlight the superior performance of our method in optimizing both data collection and energy consumption.



(a) UAV 1 (b) UAV 2 (c) UAV 3

Fig. 3. CDPS trend over the episode by each UAV. Because each UAV owns different number of SNs, there may be a different peak value of CDPS.



(a) UAV 1 (b) UAV 2 (c) UAV 3

Fig. 4. ECPS trend over the episode by each UAV. Similar to CDPS, each UAV may own a different peak value of ECPS.

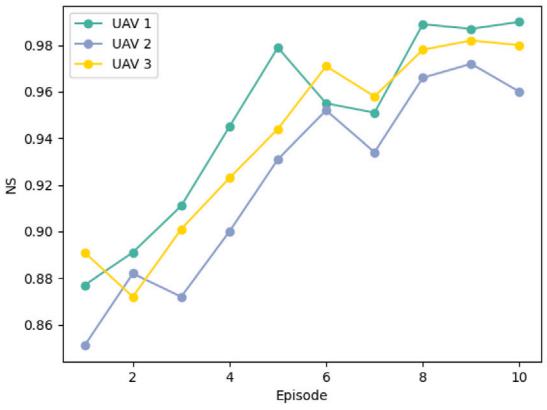


Fig. 5. NS trend over the episode by each UAV. For simplicity, we only consider NS for 10 episodes right after CDPS reaches peak value, that is, when the system reaches relatively stable.

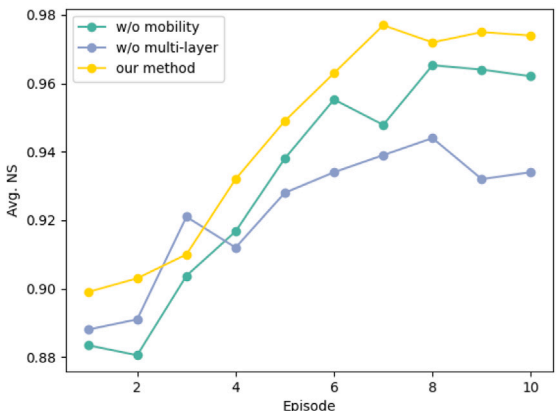


Fig. 6. NS comparison between our method and traditional method which does not consider the mobility of SNs or use unified layers without dividing UAVs.

Table 3
Experiment results on different methods. The bolded result represents the best result for the column.

Method	CDPS (bytes/step)	ECPS (J/step)	NS (%)
DQN + mobility + multi-layer	**18.32**	14.89	**97.38**
DQN + mobility	17.49	15.68	96.19
DQN + multi-layer	17.01	15.12	94.41
genetic PSO	16.28	**14.48**	95.82
MCFO	16.74	15.99	97.02
RRT	15.68	16.28	91.60

5.3. Extensive study

Through our ablation study, we have demonstrated the advantages of our method in terms of collecting efficiency, energy saving efficiency, and network stability, leveraging dynamic assignment and a multi-layer architecture. In this section, we will provide a comprehensive validation of our method by comparing it with different path planning algorithms, encompassing GPSO [17], MCFO [19], and RRT [20].

We maintain the settings described in Section 5.1 and keep the parameters of UAVs consistent across different methods, including altitude and velocity. However, it should be noted that while other methods have a continuous action space, our method has a discrete action space with only five actions. This decision is made to avoid the need for complex networks and lengthy convergence times associated with expanding the output layer of DQN. The consideration of expanding the action space (e.g., 3d space) while preserving the dynamic configuration are areas that will be addressed in future work.

In order to ensure a fair comparison among the different methods, we establish a consistent criterion for convergence by unifying the

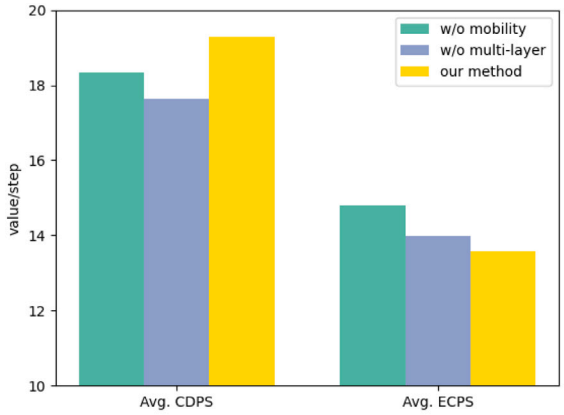


Fig. 7. CDPS and ECPS comparison between our method and traditional method which does not consider the mobility of SNs or use unified layers without dividing UAVs.

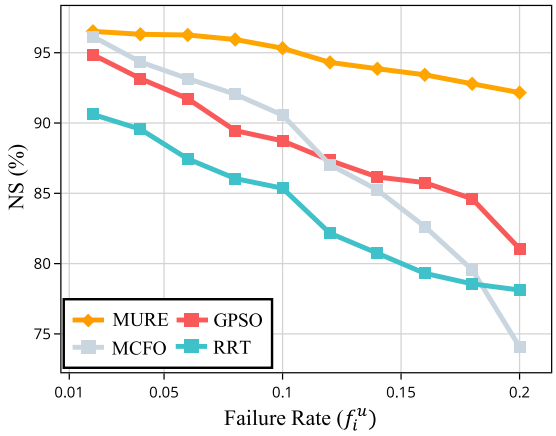


Fig. 8. The results upon employing varying failure rate with our methods and baselines. The failure rate f_i^u is varied uniformly from 0.02 to 0.2 and $f_i^s = 2f_i^u$. Note that we evaluate NS on 100th episodes where all listed methods are ensured to converge.

number of iterations[2] required for all methods to reach a certain level of convergence. As in the previous experiment, we mitigate the impact of result fluctuations by averaging the results obtained from the first 10 iterations after convergence.

Table 3 demonstrates the performance of our method compared to GPSO, MCFO, and RRT. Our method achieves the highest CDPS among the methods, with gains of 12.53%, 9.44%, and 16.84% over GPSO, MCFO, and RRT, respectively. This indicates that DQN outperforms the other methods in finding the shortest path across the farm. The improvement in CDPS is also reflected in NS, which shows gains of 1.63%, 0.37%, and 6.31% respectively, indicating a more efficient path that reduces the risk of missing nodes. However, our method exhibits slightly higher energy consumption compared to genetic PSO. This can be attributed to the computational intensity of DQN during backward propagation, especially when attempting to expand the action or state space. It implies that when considering energy consumption, the use of DQN for path planning in a 3D space may result in higher energy expenditure.

[2] Unless explicitly specified, we use the terms *iteration* and *episode* interchangeably.

5.4. Fault tolerance

In a large-scale agricultural setting, ensuring the uninterrupted operation of each individual component within the system can be a formidable challenge, given the potential for unforeseen issues leading to operational failures. Consequently, this section focuses on evaluating the system's fault tolerance, specifically examining the repercussions of component failures on the overall system. To facilitate this analysis, we introduce a failure rate f_i^s for each sensor in the system during every episode. This rate signifies the probability of a sensor completely losing functionality for the duration of the episode, which encompasses essential tasks like data collection and communication. Consequently, when a sensor experiences failure, the WSN loses a node, necessitating a recalculation of UAV actions during subsequent updates. Furthermore, UAVs themselves might encounter temporary incapacitation due to factors like energy depletion or mechanical malfunctions. To address this concern, we introduce failure rate f_i^u for UAVs, specifically Type I, representing the probability of a UAV's temporary inoperability within the current episode. In such scenarios, we expect the system to promptly reassign the nodes previously managed by the malfunctioning UAV to the remaining operational UAVs. Our evaluation of the system's robustness continues, utilizing the NS metric, and we subsequently compare our system against baseline models.

Empirically, we set $f_i^s = 2f_i^u$ and vary f_i^u from 0.02 to 0.2. Fig. 8 illustrates the robustness of our method compared to the baselines under varying failure rates, as measured by the NS metric. We observe that (1) Irrespective of the fluctuating failure rates, our method consistently outperforms the NS values of the baselines; (2) As the failure rates increase, the superiority of our approach becomes more pronounced. Specifically, while the NS values of the baselines exhibit a more rapid decline, our system's NS remains stable at above 90%. This indicates that our system retains more than 90% of the data even under high failure rates.

5.5. Further discussion

Ease of Deployment. In real-world applications, apart from prioritizing system efficiency and robustness, cost and deployment complexity are also significant considerations. In our proposed system, alongside standard and indispensable expenses (e.g., the acquisition and installation of sensors and UAVs), the primary divergence in our deployment strategy is the categorization of UAVs into Type I and Type II based on their performance attributes to facilitate parallelized management. The evaluation of UAVs is a one-time expenditure and is relatively modest, rendering it nearly negligible. Another prospective cost pertains to the training of models for UAV path planning. However, our experiments indicate minimal training time due to our utilization of parallel training involving multiple agents, with each run being completed in a matter of minutes. Furthermore, in terms of computational resources, even in a scenario involving 200 sensors, a typical consumer-grade GPU is sufficient to accommodate the system's requirements.

Practical Issues. Upon the initial deployment of drones, certain observations have highlighted potential impacts on livestock, such as increased heart rates and subtle alterations in behavior. To mitigate these effects, elevating altitude of UAVs serves to prevent operational noise from disturbing the natural activities of the livestock. Our experimental determination identifies an optimal altitude of 35 m, assuming this position maintains a safe distance to avoid livestock disturbance. Besides, a transitional phase precedes the initiation of drone operations, allowing livestock to gradually get used to the presence of drones. Over time, this approach aids in the normalization of their behavior. We also clarify that in real scenarios, a single drone typically manages several tens or even hundreds of livestock. Thus, maintaining an adequate distance from individual animals is ensured as a standard practice.

6. Conclusion

In this article, we propose a multi-layer real-time wireless network architecture. This architecture divides the wireless network into three layers: SNs at the bottom layer, and UAVs divided into two layers based on their functions for data collection and data processing. The advantages of this architecture are twofold. Firstly, it leverages the performance advantages of different UAVs, aligning with industry requirements. Secondly, it enables parallelization and pipeline processing of data collection and processing tasks, enhancing task efficiency. Additionally, we consider the mobility of SNs by updating area partitions using stream K-means and incorporating SN status into DQNs. This enables real-time SN localization and dynamic learning of shortest paths by UAVs. Our experimental results demonstrate that our method outperforms traditional approaches in terms of data collection effectiveness, energy efficiency, and network stability. Furthermore, we compare our method with popular path planning algorithms (GPSO, MCFO, and RRT), and find that our method excels in data collection efficiency and network stability, ranking second only to GPSO in terms of energy saving efficiency.

CRediT authorship contribution statement

Xinyu Tian: Writing – original draft, Methodology, Data curation. **Mahbuba Afrin:** Writing – review & editing, Writing – original draft, Visualization, Validation, Supervision, Software, Resources, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Conceptualization. **Sajib Mistry:** Writing – review & editing, Supervision, Funding acquisition. **Redowan Mahmud:** Writing – review & editing, Supervision, Funding acquisition. **Aneesh Krishna:** Writing – review & editing, Supervision, Funding acquisition. **Yan Li:** Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

References

[1] P.K. Thornton, Livestock production: recent trends, future prospects, Phil. Trans. R. Soc. B 365 (1554) (2010) 2853–2867.

[2] B.I. Akhigbe, K. Munir, O. Akinade, L. Akanbi, L.O. Oyedele, IoT technologies for livestock management: a review of present status, opportunities, and future trends, Big Data Cogn. Comput. 5 (1) (2021) 10.

[3] M. Afrin, J. Jin, A. Rahman, Y.-C. Tian, A. Kulkarni, Multi-objective resource allocation for edge cloud based robotic workflow in smart factory, Future Gener. Comput. Syst. 97 (2019) 119–130.

[4] K.H. Kwong, T.-T. Wu, H.G. Goh, K. Sasloglou, B. Stephen, I. Glover, C. Shen, W. Du, C. Michie, I. Andonovic, Practical considerations for wireless sensor networks in cattle monitoring applications, Comput. Electron. Agric. 81 (2012) 33–44.

[5] Q. Li, Z. Liu, J. Xiao, A data collection collar for vital signs of cows on the grassland based on LoRa, in: 2018 IEEE 15th International Conference on E-Business Engineering, ICEBE, IEEE, 2018, pp. 213–217.

[6] Q. Zhang, M. Jiang, Z. Feng, W. Li, W. Zhang, M. Pan, IoT enabled UAV: Network architecture and routing algorithm, IEEE Internet Things J. 6 (2) (2019) 3727–3742.

[7] S. Zhao, F. Kang, J. Li, C. Ma, Structural health monitoring and inspection of dams based on UAV photogrammetry with image 3D reconstruction, Autom. Constr. 130 (2021) 103832.

[8] K. Dineva, T. Atanasova, Cloud data-driven intelligent monitoring system for interactive smart farming, Sensors 22 (17) (2022) 6566.

[9] M.A. Alanezi, A.F. Salami, Y.A. Sha'aban, H.R. Bouchekara, M.S. Shahriar, M. Khodja, M.K. Smail, UBER: UAV-based energy-efficient reconfigurable routing scheme for smart wireless livestock sensor network, Sensors 22 (16) (2022) 6158.

[10] M.A. Alanezi, A.F. Salami, Y.A. Sha'aban, H.R. Bouchekara, RUBER: Recoverable UAV-based energy-efficient reconfigurable routing scheme for smart wireless livestock sensor network, Front. Energy Res. 10 (2022).

[11] Q.M. Ilyas, M. Ahmad, Smart farming: an enhanced pursuit of sustainable remote livestock tracking and geofencing using IoT and GPRS, Wirel. Commun. Mob. Comput. 2020 (2020).

[12] S. Salehi, J. Hassan, A. Bokani, An optimal multi-UAV deployment model for UAV-assisted smart farming, in: 2022 IEEE Region 10 Symposium, TENSYMP, IEEE, 2022, pp. 1–6.

[13] X. Liu, Y. Liu, Y. Chen, Reinforcement learning in multiple-UAV networks: Deployment and movement design, IEEE Trans. Veh. Technol. 68 (8) (2019) 8036–8049.

[14] H.R. Bouchekara, B.O. Sadiq, S. O Zakariyya, Y.A. Sha'aban, M.S. Shahriar, M.M. Isah, SIFT-CNN pipeline in livestock management: A drone image stitching algorithm, Drones 7 (1) (2022) 17.

[15] J. Li, W. Ma, Q. Li, C. Zhao, D. Tulpan, S. Yang, L. Ding, R. Gao, L. Yu, Z. Wang, Multi-view real-time acquisition and 3D reconstruction of point clouds for beef cattle, Comput. Electron. Agric. 197 (2022) 106987.

[16] K.M. Awan, H.H.R. Sherazi, A. Ali, R. Iqbal, Z.A. Khan, M. Mukherjee, Energy-aware cluster-based routing optimization for WSNs in the livestock industry, Trans. Emerg. Telecommun. Technol. 33 (3) (2022) e3816.

[17] M. Behjati, A.B. Mohd Noh, H.A. Alobaidy, M.A. Zulkifley, R. Nordin, N.F. Abdullah, LoRa communications as an enabler for internet of drones towards large-scale livestock monitoring in rural farms, Sensors 21 (15) (2021) 5044.

[18] Y. Pan, Y. Yang, W. Li, A deep learning trained by genetic algorithm to improve the efficiency of path planning for data collection with multi-UAV, Ieee Access 9 (2021) 7994–8005.

[19] Y. Chen, J. Yu, Y. Mei, Y. Wang, X. Su, Modified central force optimization (MCFO) algorithm for 3D UAV path planning, Neurocomputing 171 (2016) 878–888.

[20] S. Lavalle, Rapidly-Exploring Random Trees: a New Tool for Path Planning, Research Report 9811, Department of Computer Science, Iowa State University, 1998.

[21] D. Berckmans, General introduction to precision livestock farming, Anim. Front. 7 (1) (2017) 6–11.

[22] J. Boyazoglu, Livestock farming as a factor of environmental, social and economic stability with special reference to research, Livest. Prod. Sci. 57 (1) (1998) 1–14.

[23] L.A. González, G.J. Bishop-Hurley, R.N. Handcock, C. Crossman, Behavioral classification of data from collars containing motion sensors in grazing cattle, Comput. Electron. Agric. 110 (2015) 91–102.

[24] A.S. Voulodimos, C.Z. Patrikakis, A.B. Sideridis, V.A. Ntafis, E.M. Xylouri, A complete farm management system based on animal identification using RFID technology, Comput. Electron. Agric. 70 (2) (2010) 380–388.

[25] N. Wang, Z. Li, Wireless sensor networks (WSNs) in the agricultural and food industries, in: Robotics and Automation in the Food Industry, Elsevier, 2013, pp. 171–199.

[26] M. Afrin, J. Jin, A. Rahman, A. Gasparri, Y.-C. Tian, A. Kulkarni, Robotic edge resource allocation for agricultural cyber-physical system, IEEE Trans. Netw. Sci. Eng. 9 (6) (2021) 3979–3990.

[27] M. Afrin, J. Jin, A. Rahman, A. Rahman, J. Wan, E. Hossain, Resource allocation and service provisioning in multi-agent cloud robotics: A comprehensive survey, IEEE Commun. Surv. Tutor. 23 (2) (2021) 842–870.

[28] T. Bouguera, J.-F. Diouris, J.-J. Chaillout, R. Jaouadi, G. Andrieux, Energy consumption model for sensor nodes based on LoRa and LoRaWAN, Sensors 18 (7) (2018) 2104.

[29] B. Galkin, J. Kibilda, L.A. DaSilva, UAVs as mobile infrastructure: Addressing battery lifetime, IEEE Commun. Mag. 57 (6) (2019) 132–137.

[30] M. Angurala, M. Bala, S.S. Bamber, Wireless battery recharging through UAV in wireless sensor networks, Egypt. Inform. J. 23 (1) (2022) 21–31.

[31] M. Mozaffari, W. Saad, M. Bennis, M. Debbah, Efficient deployment of multiple unmanned aerial vehicles for optimal wireless coverage, IEEE Commun. Lett. 20 (8) (2016) 1647–1650.

[32] M.A. Alanezi, Z. Haruna, Y.A. Sha'aban, H.R. Bouchekara, M. Nahas, M.S. Shahriar, Obstacle avoidance-based autonomous navigation of a quadrotor system, Drones 6 (10) (2022) 288.

[33] G. Balamurugan, J. Valarmathi, V. Naidu, Survey on UAV navigation in GPS denied environments, in: 2016 International Conference on Signal Processing, Communication, Power and Embedded System, SCOPES, IEEE, 2016, pp. 198–204.

[34] E. Petritoli, F. Leccese, M. Leccisi, Inertial navigation systems for UAV: Uncertainty and error measurements, in: 2019 IEEE 5th International Workshop on Metrology for AeroSpace (MetroAeroSpace), IEEE, 2019, pp. 1–5.

[35] C. Aquilani, A. Confessore, R. Bozzi, F. Sirtori, C. Pugliese, Precision livestock farming technologies in pasture-based livestock systems, Animal 16 (1) (2022) 100429.

[36] M.A. Alanezi, M.S. Shahriar, M.B. Hasan, S. Ahmed, Y.A. Sha'aban, H.R. Bouchekara, Livestock management with unmanned aerial vehicles: A review, IEEE Access (2022).

[37] H. Nawaz, H.M. Ali, A.A. Laghari, UAV communication networks issues: a review, Arch. Comput. Methods Eng. 28 (3) (2021) 1349–1369.
[38] M.A. Alanezi, A. Mohammad, Y.A. Sha'aban, H.R. Bouchekara, M.S. Shahriar, Auto-encoder learning-based UAV communications for livestock management, Drones 6 (10) (2022) 276.
[39] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, M. Riedmiller, Playing atari with deep reinforcement learning, 2013, arXiv: 1312.5602.
[40] V. Mnih, K. Kavukcuoglu, D. Silver, A.A. Rusu, J. Veness, M.G. Bellemare, A. Graves, M. Riedmiller, A.K. Fidjeland, G. Ostrovski, et al., Human-level control through deep reinforcement learning, Nature 518 (7540) (2015) 529–533.
[41] B. Varga, B. Kulcsár, M.H. Chehreghani, Deep Q-learning: A robust control approach, Internat. J. Robust Nonlinear Control 33 (1) (2023) 526–544.
[42] J. Achiam, D. Held, A. Tamar, P. Abbeel, Constrained policy optimization, in: International Conference on Machine Learning, PMLR, 2017, pp. 22–31.
[43] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, O. Klimov, Proximal policy optimization algorithms, 2017, arXiv preprint arXiv:1707.06347.
[44] M. Grzes, Reward Shaping in Episodic Reinforcement Learning, ACM, 2017.
[45] B. Howson, C. Pike-Burke, S. Filippi, Optimism and delays in episodic reinforcement learning, in: International Conference on Artificial Intelligence and Statistics, PMLR, 2023, pp. 6061–6094.
[46] R.S. Sutton, Learning to predict by the methods of temporal differences, Mach. Learn. 3 (1988) 9–44.
[47] J. Hu, H. Zhang, L. Song, Reinforcement learning for decentralized trajectory design in cellular UAV networks with sense-and-send protocol, IEEE Internet Things J. 6 (4) (2018) 6177–6189.
[48] U. Challita, W. Saad, C. Bettstetter, Cellular-connected UAVs over 5G: Deep reinforcement learning for interference management, 2018, arXiv preprint arXiv: 1801.05500.
[49] C. Wang, J. Wang, Y. Shen, X. Zhang, Autonomous navigation of UAVs in large-scale complex environments: A deep reinforcement learning approach, IEEE Trans. Veh. Technol. 68 (3) (2019) 2124–2136.
[50] S. Lee, H. Yu, H. Lee, Multiagent Q-learning-based multi-UAV wireless networks for maximizing energy efficiency: Deployment and power control strategy design, IEEE Internet Things J. 9 (9) (2021) 6434–6442.
[51] W. Lee, T. Kim, Multi-agent reinforcement learning in controlling offloading ratio and trajectory for multi-UAV mobile edge computing, IEEE Internet Things J. (2023).
[52] A. Al-Hourani, S. Kandeepan, S. Lardner, Optimal LAP altitude for maximum coverage, IEEE Wirel. Commun. Lett. 3 (6) (2014) 569–572.
[53] H.T. Friis, A note on a simple transmission formula, Proc. IRE 34 (5) (1946) 254–256.
[54] R.S. Sutton, D. McAllester, S. Singh, Y. Mansour, Policy gradient methods for reinforcement learning with function approximation, Adv. Neural Inf. Process. Syst. 12 (1999).
[55] L. Benos, A.C. Tagarakis, G. Dolias, R. Berruto, D. Kateris, D. Bochtis, Machine learning in agriculture: A comprehensive updated review, Sensors 21 (11) (2021) 3758.
[56] C. Yan, X. Xiang, C. Wang, Towards real-time path planning through deep reinforcement learning for a UAV in dynamic environments, J. Intell. Robot. Syst. 98 (2) (2020) 297–309.

Xinyu Tian is currently pursuing a Ph.D. in the College of Engineering and Computer Science at the Australian National University. He obtained his Bachelor of Advanced Computing degree at the Australian National University in 2023. His research focuses on the areas of internet of things, reinforcement learning, and domain adaptation.

Dr. Mahbuba Afrin is a Lecture in the School of Electrical Engineering, Computing and Mathematical Sciences (EECMS) at Curtin University, Australia. She has a diverse academic and research background, having worked as a Vice-Chancellor's Postdoctoral Research Fellow at the University of Southern Queensland and completed her Ph.D. at Swinburne University of Technology. With expertise in Networked Robotics and the Internet of Things-enabled Cyber–Physical Systems, her research focuses on developing efficient learning models for intelligent systems.

Dr. Sajib Mistry is a Senior Lecture at the School of EECMS in Curtin University, Australia. He was a Postdoctoral Research Fellow at the School of Computer Science in the University of Sydney, Australia. He was awarded Ph.D. from the School of Science (Computer Science), RMIT University, Melbourne, Australia. He completed his Masters (MS) and Bachelor (BS) in Computer Science from the University of Dhaka, Bangladesh. He has teaching/research experience in the University of Sydney, RMIT University, Monash University, University of Dhaka, and University of Liberal Arts. His primary research area is Service Computing, Cloud/Edge computing, Machine Learning, Augmented Reality, and the Internet of Things (IoT). He has authored and edited several books and published in several top journals and conferences such as ACM TOIT, CACM, IEEE TSC, IEEE TKDE, ICSOC, ICWS, SCC, etc. He won the Best Research Paper Award in ICSOC 2016 and RMIT Publication Award 2016. One of his journal papers was selected as the spotlight paper for IEEE TSC. He also contributed significantly with his community services as PC Chair in ASSRI 2018, PC member in ICWS 2019–2020, ICSOC 2019, and WISE 2018–2019. He is a regular reviewer of top journals and conferences in the TCSVC field. He is a member of the prestigious Sydney IoT Hub.

Dr. Redowan Mahmud is currently working as a Lecturer (Computing Discipline) in the School of Electrical Engineering, Computing and Mathematical Sciences, Curtin University. He received Ph.D. from the School of Computing and Information Systems the University of Melbourne in 2020. His primary research areas are Fog/Edge computing, Secure Internet of Things, Software-defined networking, and Mobile cloud computing. Dr. Mahmud is one of the main contributors to the iFogSim simulator, FogBus framework and Con-Pi software systems, used extensively for resource management research in Fog/Edge computing.

Dr. Aneesh Krishna is currently an Associate Professor with the School of Electrical Engineering, Computing and Mathematical Sciences, Curtin University, Australia. He holds a Ph.D. in computer science from the University of Wollongong, Australia. He was a lecturer in software engineering at the School of Computer Science and Software Engineering, University of Wollongong, Australia (from February 2006–June 2009). His research interests include AI for software engineering, model-driven development/evolution, requirements engineering, agent systems, formal methods, data mining, computer vision, machine learning, bio-informative and renewable energy systems. He has published more than 130 articles in reputed journals and international conferences. His research is (or has been) funded by the Australian Research Council (ARC), and various Australian government agencies (like NSW State Emergency Service) as well as companies such as Woodside Energy, Amristar Solutions, Autism West Incorporated, BW Solar Australia, Western Australia Dementia Training Center and Andrew Corporation. He serves as an assessor (Ozreader) for the ARC. He has been on the organizing committee, served as invited technical program committee member of many conferences and workshops in the areas related to his research.

Prof Yan Li received her Ph.D. degree from the Flinders University of South Australia, Australia. She is a full Professor in Artificial Intelligence (AI) in Computer Science discipline, the Associate Head (Research) and the Chair of Research Committee in the School of Mathematics, Physics and Computing at the University of Southern Queensland (USQ), Australia. Her research interests lie in the areas of AI, Machine Learning, Big Data and Internet Technologies, Security, Signal/Image Processing and EEG Research, etc. Prof Yan Li has published 200+ high quality publications, supervised dozens of Ph.D. completions, and obtained more than 3 million research grants from Australia government, industry and through international collaborations. Prof Yan Li is the leader of USQ AI Research and the recipient of many research and teaching excellence awards, including

2012 Australia prestigious National Learning and Teaching Citation Award, 2008 Queensland Government Smart State-Smart Women Award, 2009 USQ Teaching Excellence Award, 2009 USQ Research Excellence Award, 2015–2018 Research Publication Excellence Awards, and 2021 HES Faculty Research Award. Prof Yan Li has served as an elected academic leader in many high-level university committees, such as USQ Academic Board Executive Committee and USQ Research Committee etc. Prof Yan Li is a current member in Australian Research Council College of Experts.