

Received 19 June 2024, accepted 2 July 2024, date of publication 9 July 2024, date of current version 22 July 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3425523

RESEARCH ARTICLE

AI-Enabled Collaborative Distributed Computing in Networked UAVs

BASSEM MOKHTAR, (Senior Member, IEEE)

Department of Computer and Network Engineering, College of Information Technology, United Arab Emirates University, Al Ain 15551, United Arab Emirates

e-mail: bassem.mokhtar@uaeu.ac.ae

This work was supported by the Start-Up Research Program—College of Information Technology, United Arab Emirates University, under Grant G00004585/12T050.

ABSTRACT Nowadays, the evolution of AI is noticed in supporting many life applications and manipulating different data types. It helps complete the tasks and get the required information efficiently and precisely. The deployment of AI techniques and machine learning models moves to limited-resources energy-constrained platforms, ranging from simple IoT devices to unmanned aerial vehicles (UAVs). Employing such models on limited-resource UAVs to support a wide range of applications is an inevitable duty and it is at the same time a challenging task. Additionally, obtaining high-accuracy outputs from a single AI-enabled UAV within the operating context of delay-sensitive applications faces a lot of obstacles, and may not be feasible. Accordingly, distributed operations and cooperation among a set of UAVs can provide the required level of accuracy within the time constraints for some applications. This work proposes a distributed computing architecture for networked UAVs based on collaborative learning and edge-of-things computing. Such architecture would help a suite of UAVs to train based on their local ML model and captured data and to collaborate with other UAVs in the same network to generate an aggregated ML model that improves the operation accuracy with acceptable performance speed. Using a networked UAV system and various application scenarios, numerical simulation studies have been presented. The performance analysis and results show how the proposed distributed computing architecture with collaborative learning outperforms the centralized computing architecture with edge and cloud computing paradigms.


INDEX TERMS Artificial intelligence, collaborative learning, lightweight training, networked UAVs, resource allocation, vehicular edge of things computing.

I. INTRODUCTION

In this era and upcoming period, there are plenty of applications and services that run and support users' needs depending on Internet-enabled embedded systems. The Internet of Things (IoT), Internet of Vehicles (IoV), and Internet of Drones (IoD) are clear examples of such networked systems with resource-constrained devices [1]. Those systems are implemented at edges near users to enable real-time high-speed privacy-preserved data communications. Artificial Intelligence (AI) has been involved in diverse real-life applications and deployed on various hardware devices in such systems. Implementing AI-based solutions on limited energy resource-constrained systems, such as unmanned

aerial vehicles (UAVs), is not an easy task. AI engines require high computation and energy power besides their dependence on huge memory sizes. Employing AI/machine learning (ML) algorithms would help tackle challenges when operating these systems such as i) unreliable connectivity, ii) high latency, iii) low data throughput, iv) high power consumption, and v) resource management.

There are unique characteristics in UAV networks that enforce to employ of ML algorithms to optimize operations and consequently the quality of services of offered applications and services [2]. For instance, the highly dynamic networking environment and the intermittency in communication paths are two main characteristics of UAV networks. ML enables drones to learn data patterns related to different network topology-related parameters (wireless channels, traffic rates,), and service-related parameters (required

The associate editor coordinating the review of this manuscript and approving it for publication was Guillermo Valencia-Palomo .

resources, security requirements, ...). Drones can adopt learned patterns autonomously to predict needs in the future to establish reliable communication channels and to transfer data efficiently to meet the quality of services of running applications. Also, drones can adapt to any changes in operating conditions to meet an acceptable level of quality of service.

Embedded machine learning [3] can support employing machine learning algorithms on networked embedded systems, such as UAV networks, with limited resources to offer reliable services to overcome the challenges in these networks. Embedded machine learning can help in planning for efficient data transmission and minimizing power consumption in limited resources Internet of Things devices [4]. In addition, ML can help in handling big data with privacy preservation in IoV environments [5]. In UAV networks, several types of embedded machine learning algorithms can be adopted [2]. Unsupervised learning techniques require training datasets to expect the output, however, the training data is not labeled to a specific output. Drones can learn classes of output to be able to predict patterns and outcomes. Supervised learning techniques require labeled training datasets that can help to learn drones the expected outcomes when having specific input data. Semi-supervised learning techniques depend on labeled and unlabeled training data that can help in predicting output whether without learning a specific related pattern or not. Reinforcement learning techniques enable drones to learn by experience through trial and test processes and to evaluate the obtained outcome targeting a maximized cumulative reward. Deep Reinforcement Learning (RL) algorithms have been used in UAV-related duties, such as UAV target tracking control, to help in learning the best strategy for designing a suitable tracking controller [6]. In addition, the RL-based UAV controller design enables UAV control adaptation in complicated operating environments [7]. It targets more instructive and directional training phases for developing an optimized ML model. In addition, RL algorithms can aid energy-constrained UAV systems to work efficiently in various networking and communication systems targeting optimized energy consumption and prolonging operating time [8].

In the previously discussed ML algorithms, the ML models based on the implemented algorithms have to be implemented locally in each drone and each drone has to employ those models and keep training those models to be able to rely on them and achieve reliable efficient operations. Since employing ML requires utilizing and consuming some UAV resources, UAVs may have limited capability to conduct the required tasks. UAV resources overloading with tasks in a highly dynamic and mobile network is still representing a major problem that should be tackled with efficient solutions. Training efficiently a machine learning model on a UAV requires a huge amount of data and requires a long time. Also, the inference time expected for a UAV when running its trained model may vary and not be acceptable. So, it is required to utilize lightweight machine learning algorithms on UAVs that consider their limited resources, and at the

same time, such algorithms should be efficient and generate accurate outcomes in a timely manner.

To improve the training capability of UAVs, federated learning (FL) architectures are recommended to be deployed. FL is presented to provide an advantage of decoupling training ML models from accessing directly training data, which results in privacy preservation [9]. This leads to leveraging a set of trained ML models in drones where such models can be aggregated to generate an efficient ML model without sharing related locally-maintained training raw datasets between interacting and communicating drones. There are centralized and distributed FL architectures proposed to support cooperating embedded systems [10]. In FL, each drone has its own dataset and is not shared with other drones. Drones can get a trained ML based on decentralized data among various drones where that trained model may be shared by a master drone or a central server.

In this research paper, we propose and discuss a computing architecture for UAV networks based on adopting a distributed collaborative learning algorithm and edge-of-things computing architecture. Such architecture would help a suite of UAVs to train their ML models based on decentralized data and with sharing raw data with other UAVs in the same network. An aggregated ML model will be generated and shared between UAVs in order to support realtime applications where the aggregated model improves the operation accuracy with acceptable performance speed.

The main contributions of this work are outlined as follows.

- Developed system model of a distributed computing architecture for networked UAVs based on collaborative learning and edge-of-things computing.
- Algorithms for task offloading and collaborative learning for distributed UAVs integrated with edge and cloud computing architectures.
- Analytical studies for comparing various computing paradigms organized in a distributed way to serve different types of UAV-related services.

The remainder of this paper is organized as follows. Section II highlights some relevant fields in literature with a focus on some related work. The proposed UAV network system model with distributed collaborative learning capability is presented in section III. The conducted numerical studies are shown and discussed in section IV. The paperwork concludes in section V with a reference to some future directions.

II. RELATED WORK

Due to the high computational requirements of the machine learning algorithms and the resource limitations of the embedded systems, embedded machine learning emerges to fill in that gap and support the running of various real-time applications on mobile devices [3]. The applications include ones related to computer vision, health care, transportation, and environmental monitoring. Self-driving and autonomous vehicles are a domain of interest that attracts researchers to investigate embedded machine learning algorithms that support this industry. Many works have been introduced to address related applications. In [11], the authors discussed

some risks and challenges of applying AI in autonomous vehicles. For sure, there are some aspects to be considered when choosing a machine learning algorithm for a certain embedded system and the targeted application. These aspects might be related to available energy and memory resources, running costs, and the required quality of service.

In [3], the authors overviewed various application-oriented embedded systems that employ specific machine learning models and architectures to suit the hardware specifications and the applications' QoS demands. Multiple machine learning classes can be used. Unsupervised and supervised ML algorithms can be used, such as the k-nearest neighbor algorithm, naïve Bayes, decision tree, random forest, logistic regression, support vector machines, convolutional neural networks, deep neural networks, recurrent neural networks, long short-term memory networks, and hidden Markov models.

For UAV networks, several machine learning algorithms can be used to enable real-time applications, such as pattern recognition and object detection [2]. Any applied machine learning algorithm should be able to support the demands of robotic applications in UAV environments like low latency, low power consumption, and high reliability. Machine learning algorithms include deep Q-network, deep Q-learning (DQL), and liquid state machines. A survey study [12] highlighted some machine learning algorithms that can support and improve definite UAV-based robotic applications. For instance, artificial neural networks would help in enhancing delay, connectivity, and security in aerospace robotics. Moreover, ML approaches can be adopted with UAVs systems and networks to enable/support processes, such as trajectory and placement, path planning, interference management, situational awareness, and motion control.

To overcome the limitation that might exist in the energy and memory resources of embedded systems that employ machine learning models and with the emergence of privacy concerns when using cloud-centric ML, federated learning (FL) [9] is recommended to be used. The capability of machine learning-enabled embedded systems will not enable such systems in many times to update their machine learning models efficiently. This may be because of not having sufficient data or inability due to energy and computation resources. In general, FL would enable multiple embedded systems to collaborate on training an ML model without sharing their own data. FL is applied better with edge computing architectures to shorten the delay and utilize efficiently bandwidth. There are various architectures for FL [10] that can support diverse designs of embedded systems networks and multiple applications. There are centralized and decentralized federated learning architectures that can support operations and applications within the UAV networks. In centralized FL architectures, any implemented FL approach will depend on a central machine or server for generating the aggregated ML model. Some challenges and problems may appear due to this dependency, such as the single point of failure with respect to this central entity. On the other hand, with decentralized FL (or serverless FL) architectures, each UAV will have and train

its ML model and it will share its trained model with one-hop neighboring UAV or a cluster head UAV for aggregating all received ML models and generating a global ML model. Such decentralized FL architectures can be classified as collaborative FL, multi-hop FL, and Fog learning.

Several computing architectures were presented to enable efficient and resource-matching computation and storage services for static and mobile network devices. For UAV systems, vehicular computing architectures would be better to be used to support the highly dynamic environment of communicating UAVs and the running of real-time applications. Different computing paradigms, such as mobile and vehicular edge computing, were discussed to support edge computing collaboration of vehicular devices in various networks [13]. It was recommended to adopt edge of things computing architecture that can provide suitable computing resources near a wide range of vehicular devices [14]. In [15], a vehicular edge of things computing architecture was discussed. Such edge computing architectures are suitable for running real-time video recognition and computer vision applications [16].

Selecting an appropriate computing architecture would help networked and communicating devices and controllers apply efficient resource allocation and task offloading. For instance, vehicular edge computing enables efficient computation and storage capabilities for mobile devices and UAVs via offloading to edges (road side units) or with the capability to decide whether offloading at edges or cloud servers [17]. In [18], two methods were presented to enable efficient bandwidth allocation among a suite of communicating and networked UAVs with limited energy resources. Other research work considered resource allocation as an optimization problem where it can relieve the energy consumption challenge in UAV networks [19], [20]. The authors in [21] presented a distributed Edge-Cloud collaborative framework for accurate and realtime UAV-based object detection. The framework depended on a lightweight object detection algorithm that can be implemented on embedded systems. a fuzzy NN-based mechanism was adopted to enable task allocation in edge or cloud. In [22], a hierarchical UAV-assisted data processing framework was developed to support data processing and computing offloading targeting minimized age of information.

III. SYSTEM MODEL

We developed a system model of a UAVs network that follows a simple star topology. Each UAV will be represented by an embedded ML system/IoT platform equipped with a camera (to capture RGB images) and enough memory. We include an edge server and a central cloud server that can be reached remotely by a specific UAV to provide powerful resources in case of need. Figure 1 shows the proposed framework for the proposed UAV-based distributed computing system.

Various UAV network models were surveyed in [23]. A UAV network may comprise one UAV for a single mission. For large-scale missions, UAV networks comprise multiple interactive UAVs. There are various network architectures as

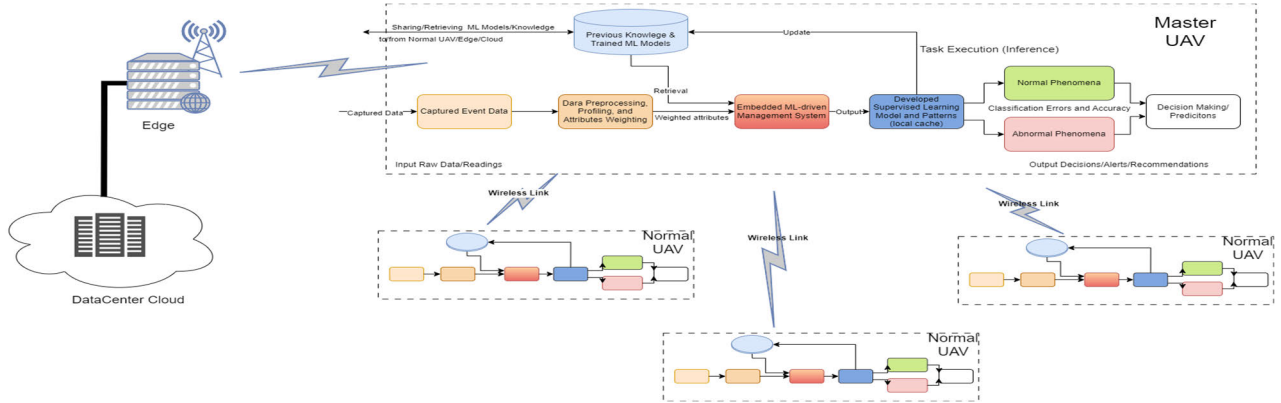


FIGURE 1. UAV-based distributed computing framework.

follows. UAV network of cell level (UCL), UAV network of system-level (USL), and UAV network of the system of system level (USoS). For USL, there are five representative examples of the USL topology; infrastructure-based, single-star, multi-star, flat ad hoc, and hierarchical ad hoc architectures. Considering USoS, there are two discussed architectures, which are LAN-based cloud service architecture and Internet-based cloud service architecture.

A network system model is presented for UAVs where a set of connected UAVs with a cloud can communicate directly with mobile handheld devices on Earth [24].

Different communication and network models were presented to UAVs. In [25], a network cluster model for UAVs was discussed. A mobility model for an ad hoc UAV network was presented in [26]. A network communication model was presented in [27]. In addition, a traffic flow model for UAV was presented in [28].

We consider the system model and the proposed scenario in [17] where we will adapt it to a UAV system. There are different levels of computing: cloud computing, edge computing, edge of things computing with collaborative learning, and local computing.

We adopted the system model of USoS of the Internet-based cloud service architecture presented in [23]. There are sets of UAVs connected to base stations and then to a cloud.

Table 1 shows the system model notation. In each area out of M areas with N UAVs, there will be a cluster C of UAVs led by a master UAV forming a star topology. Accordingly, each UAV cluster comprises m_c UAVs, $m_c \leq N$

So, there are M clusters, and the total number of UAVs in all clusters

$$\sum_{i=1}^M m_{c_i} = N \quad (1)$$

In each cluster, there will be one master UAV and a number of normal UAVs up to $N-1$.

Each task i can be processed on the UAV (i.e., edge of things offloading), or offloaded to the base station server (i.e., edge offloading when $A_UAV_i \leq 0$), or accomplished on the cloud server (i.e., cloud offloading) through the base

TABLE 1. System model notation.

Symbol	Meaning and Formula
N	number of UAVs (drones)
BS_i	base station server i (each base station is equipped with one server)
M	number of base stations or area segments where each base station serves a segment
d	interdisance between UAVs and between UAV and the base station
U_{int_j}	task of UAV offloaded internally to a computing resource j inside UAV
UAV_{t_i}	the whole tasks executed by k computing resources in UAV i , $UAV_{t_i} = \sum_{i=1}^k U_{int_i}$
UAV_{c_i}	Computing capacity of UAV i
UAV_{s_i}	Storage capacity of UAV i
A_UAV_i	The availability of UAV i computing capacity, $A_UAV_i = UAV_{c_i} - UAV_{t_i}$
U_{ext_j}	task of UAV j offloaded externally to a base station
BS_{t_i}	the whole tasks offloaded to base station server i from n UAVs, $n \leq N$, $BS_{t_i} = \sum_{j=1}^n U_{ext_j}$
BS_{c_i}	Computing capacity of a base station server i
BS_{s_i}	Storage capacity of a base station server i
A_BS_i	The availability of a base station i computing capacity, $A_BS_i = BS_{c_i} - BS_{t_i}$
T	number of tasks UAV/base station may accomplish where each task i is characterized by $T_i: \{s_i, c_i, t_i\}$ where s_i is the task size, c_i is the required computing resource speed, and it is the maximum accepted delay

station (i.e., when $A_BS_i \leq 0$). The base station is considered with a capability to communicate with a number of master UAVs. The communication between cluster members (the master UAV and normal UAVs and the base station) is done over an average distance d meters.

Each cluster of M clusters is served by a base station and is formed based on the targeted tasks and events to be detected. In case there is a need to allow or order UAV to join a cluster to provide further computing capabilities, this will be valid as the number of connected UAVs in less than $N-1$. New UAV clusters can be formed and connected to a base station to serve more areas and to execute more tasks.

Each UAV will employ one machine learning model to help in completing the required task (e.g., image classification). The approach is schematically shown in Fig. 2.

A. NETWORK AND COMPUTATION MODEL

As mentioned previously, there are various ways for offloading the tasks. It is proposed that a suite of UAVs will form a star topology with a master UAV that can manage the processes of tasks offloading to base station servers. Also, the master UAV will aggregate the machine-learning models collected from UAVs in the cluster and send them the aggregated model. Also, the master UAV will be responsible for sending collected machine learning models from all UAVs and forward to the base station and then to the cloud to generalize the model, get that model, and forward it to all UAVs. We target a similar data collection system model for UAVs as the one presented in [29]. We assume that each UAV is equipped with sensors (i.e., data sources)

We target the task completion delay and accuracy by considering the used machine learning model detection accuracy (MLA), machine learning model training/updating time (T_{MT}), machine learning model exchange time (T_{ET}), task transmission time (T_{Tt}), task data propagation time (T_{Pt}) in the network, task processing time (T_{Proc}), and UAV connection time with the edge or cloud (T_C) as performance measures to compare the various computing architectures.

In the formation process of each cluster, the assumption is that interdistance d between the master UAV and the normal UAVs and between the master UAV and the base station will encounter delay plus the processing delays by the employed ML engines that would not affect the transmission delay threshold of data and the accomplishment of tasks by UAV members in the cluster.

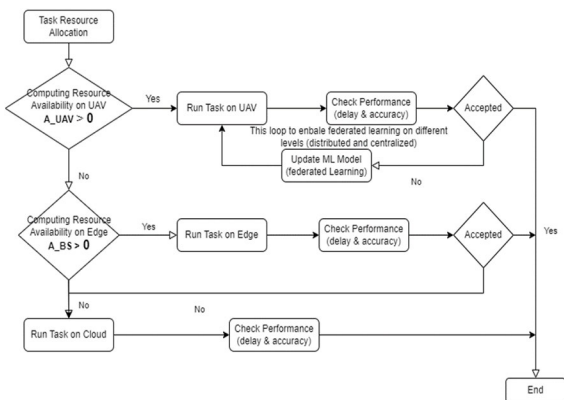


FIGURE 2. Task resource allocation and computation.

Moreover, the energy consumption of UAVs is following the model of hovering battery-powered WiFi-enabled UAV presented in [30] where UAV is stationary in a place in the air. There is a relationship extracted in [30] that considers the energy required for hovering and flying at altitude h for t seconds is as follows $E = (4.917 h + 275.204) t$. It is assumed that before assigning a task to an UAV, a prediction model

would be applied to expect the amount of energy and time needed to accomplish task besides the energy required for landing after completing the task. Based on the current energy level at UAV, the decision to offload the task would be taken.

B. LOCAL COMPUTING (EDGE OF THINGS OFFLOADING)

UAVs can accomplish the tasks on their embedded computing resources and no tasks-related data will be shared with other UAVs/servers. Each UAV is equipped with a set of IoT platforms/computing devices that form an edge of things computing inside the UAV. Based on the utilization and availability of resources in UAV and the number of assigned tasks T . So, if $A_{UAV} > 0$ (there are sufficient resources) or $UAVt == T$ (number of assigned tasks that can be executed equally to the number of all required tasks), the UAV can accomplish tasks locally. Figure 3 shows the algorithm of resource allocation for executing a task at UAVs.

Algorithm 1: Resource Allocation for Task. There are N UAVs that form a cluster; each UAV is indexed by n ; one edge station is indexed by E ; one cloud server is indexed by C .

```

Local Computing:
Initialize UAV resources
for each round  $t = 1, 2, \dots$  do
for each UAV  $n$  do
if task available
measure task data arrival rate  $\lambda$ 
check available service rate  $\mu$ 
LC: if ( $\mu_n \geq \lambda_n$ )
run the local ML model (run the task)
measure performance
if performance is accepted
go to Done
else
Collaborative Learning: Master UAV-get ML models and weights from  $N-1$  UAVs
update weights of the local ML model
go to LC
Edge Computing: else
if ( $\mu_E \geq \lambda_E$ )
run the ML model on the edge
measure performance
if performance is accepted
go to Done
Cloud Computing: run the ML model on the server
measure performance
Done: task completed
    
```

FIGURE 3. Local computing - task resource allocation algorithm.

- Without collaborative learning :** The UAV will depend on its developed and locally updated machine learning models. So, for a task j to be executed on UAV $_i$ in area m , we will use this tuple (j, i, m) In this case, transmission time and propagation time $T_{Tt(j,i,m)} = T_{Pt(j,i,m)} = 0$

$$\begin{aligned}
 & \text{Delay}_{L_withoutFL(j,i,m)} \\
 & = T_{Proc(j,i,m)} = \frac{s_j \text{taskjsize}}{c_j \text{computing resource speed for task j}} \tag{2}
 \end{aligned}$$

$$\begin{aligned}
 & \text{MLA}_{L_withoutFL(j,i,m)} \\
 & = \frac{n_j \text{number of correct objects detections in task j}}{n_{j,total} \text{total number of detections in task j}} \tag{3}
 \end{aligned}$$

It is expected that the number of correct objects detections $n_j \ll n_{j,total}$

- With collaborative learning [9], [31]:** The UAV will depend on its developed machine learning models and

Algorithm 2: Distributed Collaborative Learning for N UAVs that form a cluster; one master UAV and $N-1$ normal UAVs

```

Master UAV:
Initialize an ML model with weights
for each UAV  $t = 1, 2, \dots, N-1$  do
    get the trained ML model and weights from the local ML models
    update weights
    check performance
update weights of the ML model
    
```

FIGURE 4. Collaborative computing - task resource allocation algorithm.

be globally updated with the help of other UAVs and central servers. So, we will consider the following two cases.

- a. *Centralized collaborative learning:* updates on the adopted machine learning model done on a central server in the cloud.

$$\begin{aligned} \text{Delay}_{L_CFL(j,i,m)} &= T_{MT} + T_{Proc(j,i,m)} \end{aligned} \quad (4)$$

$$\begin{aligned} \text{MLA}_{L_CFL(j,i,m)} &= \frac{n_j \text{number of correct objects detections in task } j}{n_{j, \text{total}} \text{total number of detections in task } j} \end{aligned} \quad (5)$$

It is expected that the number of correct objects detections $n_j < n_{j, \text{total}}$

Besides the time required for training, the training model time T_{MT} will include the time overhead to send and receive the model from the cloud.

- b. *Distributed collaborative learning:* updates on the adopted machine learning models done on one UAV or a set of UAVs in a cluster with a star topology and then shared with all UAVs in the cluster. Algorithm 2 in Fig. 4 describes the main processes of distributed collaborative learning.

$$\begin{aligned} \text{Delay}_{L_DFL(j,i,m)} &= T_{MT} + T_{Proc(j,i,m)} \end{aligned} \quad (6)$$

$$\begin{aligned} \text{MLA}_{L_DFL(j,i,m)} &= \frac{n_j \text{number of correct objects detections in task } j}{n_{j, \text{total}} \text{total number of detections in task } j} \end{aligned} \quad (7)$$

It is expected that the number of correct objects detections $n_j < n_{j, \text{total}}$

The model training time in this distributed learning case will be shorter than the one in the centralized learning.

C. EDGE COMPUTING (EDGE OFFLOADING)

UAVs can offload tasks to be executed on base stations (edges) based on the utilization and availability of resources in UAV and the number of assigned tasks T . So, if

$A_UAV < 0$ (there are no sufficient resources) or $UAVt < T$ (number of tasks that can be executed less than the required tasks), UAVs will forward tasks to base stations.

$$\begin{aligned} \text{Delay}_{E(j,i,m)} &= T_{Proc(j,i,m)} + T_{Tt(j,i,m)} + T_{Pt(j,i,m)} + T_{C(j,i,m)} \end{aligned} \quad (8)$$

$$\begin{aligned} \text{MLA}_{E(j,i,m)} &= \frac{n_j \text{number of correct objects detections in task } j}{n_{j, \text{total}} \text{total number of detections in task } j} \end{aligned} \quad (9)$$

D. CLOUD COMPUTING (CLOUD OFFLOADING)

Base station servers can offload tasks to be executed on cloud servers (cloud) based on the utilization and availability of resources in base station servers and the number of assigned tasks T . So, if $A_BS < 0$ (there are no sufficient resources at the base station) or $BSt < T$ (number of tasks that can be executed less than the required tasks), base stations will forward tasks to cloud servers.

$$\begin{aligned} \text{Delay}_{C(j,i,m)} &= T_{Proc(j,i,m)} + T_{Tt(j,i,m)} + T_{Pt(j,i,m)} + T_{C(j,i,m)} \end{aligned} \quad (10)$$

$$\begin{aligned} \text{MLA}_{C(j,i,m)} &= \frac{n_j \text{number of correct objects detections in task } j}{n_{j, \text{total}} \text{total number of detections in task } j} \end{aligned} \quad (11)$$

The task propagation time T_{Pt} will be longer in the case of cloud computing compared with the case of edge computing with respect to the distance the task should travel.

IV. NUMERICAL SIMULATIONS

In this section, we conduct a set of numerical simulation studies based on a system model proposed for networked UAVs as shown in Fig. 5. Mathematica-based simulations are conducted.

A. ASSUMPTIONS

The adopted UAV system design considered the following assumptions.

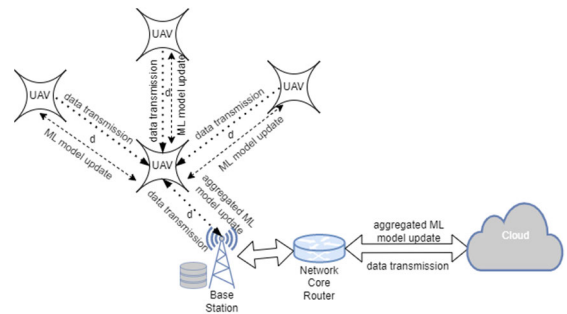


FIGURE 5. Local computing - task resource allocation algorithm.

- Network design
 - One UAV cluster: A set of four UAVs forms a star topology and one of them is the master UAV that other UAVs can communicate with directly.

- Each UAV is equipped with one IoT platform that is attached to one sensor and one computing device.
- The inter-distance d between all UAVs and between the master UAV and the base station is equal and fixed.
- Each UAV is always in proximity to the master UAV and it can communicate with and send data directly to it.
- Data sending
 - Each UAV sensor is collecting data per task with rate λ
 - Each UAV is sending task data with rate equal to task data collection rate λ
 - Normal UAV has a buffer size equals

$$\lambda \times \text{Task Delay}$$

- The master UAV has a buffer size that equals to $\sum_{i=1}^4 \lambda_i \times \text{Task Delay}$
- Communication channel and impact
 - UAVs adopt IEEE 802.11 MAC protocol
 - Each wireless channel between UAVs and the base station is suffering from path loss n .
- Energy consumption
 - Each UAV is assumed to be powered with on-board battery and has energy level with consumption rate and flies on altitude and communicate within a distance d . All these factors enable UAV to be successfully accomplish the assigned offloaded tasks.
- Resource allocation and computing
 - Considering one cluster with small number of connected UAVs to minimize the collaborative learning time among UAVs without centralized training data.
 - Each UAV has its ML model, which can be trained locally based on captured RGB images/dataset.
 - The master UAV that can get ML models from the other UAVs to generate an aggregated ML model and broadcast it to all UAVs.
 - A simple image classification and object identification application (i.e., computer vision application) with time constraints will be targeted. Convolutional neural networks can be used.
 - In the proposed UAV-based distributed computing architecture where energy-efficient inference engines with structured pruning strategies for model compression are applied heavily at local UAV and minimized heavily at the cloud computing (CC). In other words, the number of neural network layers and neurons has the largest value at CC and the smallest value at LC. It was assumed that UAV is equipped with embedded ML models with limited capability.
 - The following computing architectures will be considered in the evaluation studies.
 - **UAV-based computing:** each UAV employs its local ML model to get the result. In the case of normal UAV with Local Computing (LC), an ML model was built using a linear neural network

accompanied by layers for input and output has been adopted. In the case of master UAV with Edge of Things Computing (EoTC), an additional linear neural network was added to the ML model design presented at the LC computing.

- **Without collaborative learning:** Each UAV is locally analyzing data based on its ML model to generate outcomes with rate μ (assuming that $\mu \geq \lambda$) and there are sufficient computing resources (i.e., $A_{\text{UAV}} > 0$)
- **With collaborative learning: Distributed learning:** the master UAV will receive ML models and the initialized weights from all UAVs and it will build an aggregated ML model use and send it back to all UAVs. This learning will enable a computing architecture, we call it edge of things computing, where learning and decision are taking based on capabilities of learning models and hardware at multiple UAVs.
- **Edge Computing (EC):** The edge is equipped with an ML model with more neural network layers than in UAVs with less compression. In that case, tasks will be offloaded to edge where UAVs have no capabilities (i.e., $A_{\text{UAV}} < 0$) or ML models; and accordingly, they send all data to the edge server to analyze and send back the results.
- **Cloud Computing (CC):** The cloud is equipped with an ML model with more neural network layers than in the edge with less compression. In that case, tasks will be offloaded to cloud where UAVs have no ML models and the edge servers do not have the related ML models or there are no sufficient capabilities ($A_{\text{BS}} < 0$). The edge server sends all data to the cloud server to analyze and send back the results.

We configured the UAV system model shown in Fig. 5. Table 2 describes the main simulation parameters.

A dataset was generated and Table 3 shows its main information. The dataset refers to the classification to measurements done by UAVs based on phenomena of interest (image of activity). Every entry in the dataset comprises 4 attributes, namely, measurement class, measurement reading value, targeted phenomena type, and label (output class). We followed a supervised learning approach where each data entry in the dataset includes information about the output decision (classification result) as a label.

B. SCENARIOS

Scenario (1): a simulation study of the effectiveness of using UAVs with linear layer neural network and adopting different computing architectures to help accomplish a non-time-sensitive data classification duty.

We targeted a non-time-sensitive simple classification task where the UAV system shown in Fig. 5 will be employed for it. The task is to classify some collected environment-related measures into specific classes. There are three classes with two different types that can help in interpreting phenomena

TABLE 2. Simulation parameters.

Parameter	Definition and Related Equation	Values Range/Unit
Data arrival rate λ	-----	1 – 5 packets/seconds
Data service rate μ	$\mu \geq \lambda$	Packets/seconds
Packet size	-----	500 – 3000 bytes
Transmission power P_{tran}	-----	80 dBm
UAV interdistance d	-----	2- 10 meters
Path loss index n	-----	On average 7 for distance (2-10 meters) [28]
Average path loss P_{loss}	$10n\log(d)$ [32] [33]	20 – 70 dB
Received signal strength indicator (RSSI)	$RSSI=P_{tran}-P_{loss}$	~ 10 – 60 dBm
ML model loss function: mean squared error (MSE)	$\sum_{i=1}^N \frac{\text{difference between detection } i \text{ and the truth}}{N \text{ total number of detections in task}}$	Variable (0 – 1.0)
ML model performance measure accuracy	$\frac{\text{number of correctly predicted classes}}{\text{total number of predicted classes}}$	Variable (0 – 1.0)
task transmission time (T_T)	The time required to push out the data to the channel (Data packet size/channel bandwidth)	(range of milliseconds)
task data propagation time (T_P) in the network	The time required to transfer data to the computing device (travelled distance d /propagation speed)	(range of milliseconds)
Connection time (T_C)	Time needed to establish a connection between the master UAV and edge/cloud $T_C = T_{P_i} + T_{Proc}$	(range of milliseconds)
Number of collected data points	Data collected by the networked UAV system to learn and classify	500
Training data points	Percentage of data points used for training	70% (350)
Data point profile	Measurement class, reading value, type, label	String (1 st , 2 nd , 3 rd), scalar (range 5 to 80), string (phenomena 1 or 2), string (normal, abnormal)
ML Activation functions	Logistic sigmoid	-----
Machine learning model training/updating time (T_{MT})	The time required to update and train the ML model	(range of seconds)
machine learning model exchange time (T_{ET})	The time required to broadcast the trained ML model	(range of milliseconds)
task processing time (T_{Proc})	The time required to process the task by the ML model $T_{Proc} = T_{MT} + T_{ET}$ including time for data preprocessing, learning patterns and training the model.	(range of seconds)
Execution time (T_{Exec})	The whole time required to accomplish the task (the inference) by the used computing architecture $T_{Exec} = T_C + T_{Proc}$	(range of seconds)

from collected images, such as the possibility of having flooding, thunderstorms, and cyclone. A simple linear layer Neural

Network (NN) is employed in each UAV to help in accomplishing efficiently the classification duty. The linear layer

TABLE 3. UAV dataset information.

Number of dataset entries	500
Number of attributes	4
Number of measurement classes	3
Possible measurement class	1 st , 2 nd , 3 rd
Possible measurement value	An integer value from 5 to 80
Number of phenomena type	2
Number of possible outputs	2
Possible outputs	Normal, Abnormal
Training data percentage	70% (default)

NN model is associated with a mean squared error (MSE) layer and is trained using ADAM Optimizer to minimize the loss function and adjust the hyperparameters. A master UAV, with multiple platforms, which is far d meters from normal UAVs is employing a linear NN that is formed based on the getting weights from NNs at normal UAV using the Identity method. As we discussed in Fig. 1 and Algorithm 1, the model at the master UAV is updated with the NN models and their weights once it is used. In case of unavailability of computing resources or not meeting QoS measures the master UAV, a 2×2 linear NN with MSE layer can be employed at the edge with capability to combine more than one reading at the same input. In case of unavailability of resources at the edge, a 3×3 linear NN with MSE layer can be employed at the cloud with powerful computation to help in getting better results and expectations for the classes of the measurements.

We simulated four levels of computing as discussed previously. The first level is local computing (LC) where each UAV employs the implemented ML model. The second level is edge of things computing (EoTC) where the master UAV employs and initializes an ML model based on defined ML models at the other UAVs in its cluster. The third level is edge computing (EC) where an edge server near to the UAV cluster employs an ML model with high computation capability. The last level is cloud computing (CC) where a cloud server with a powerful ML model will be used. The obtained results were averaged over five runs.

We target the following performance metrics and the related measures to quantitatively compare the performance of the different computing levels.

1. Time complexity: this metric refers to the time overhead needed by the various computing paradigms to execute the assigned tasks of learning data patterns and classifying the outputs.
 - a. The task execution delay (measured in seconds): this measure was considered where it refers to the computing speed at the computing paradigm engine with or without previous learning. This delay considers the connection time and the processing time of the ML model.
2. ML model performance: this metric refers to the ML model's capacity to perform efficient classification tasks

- b. Mean square error: it measures the average squared difference between the predicted and the actual target values by the ML model within the acquired dataset.

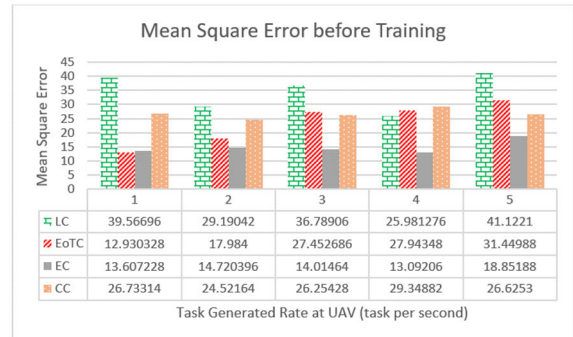


FIGURE 6. Obtained MSE before training and updating ML models.

Figures 6 and 7 show the obtained MSE by the used ML models before and after getting trained at various computing levels when varying the task arrival rate (i.e., task-related data packets generated by UAVs). It is noticed that the local computing (LC) achieved the worst MSE level before training and better MSE levels at the other computing paradigms. However, after training, the computing levels achieved better MSE values. In Figs. 8 and 9, the execution time overhead is given. Longer execution time is achieved at edge and cloud computing levels due to the impact of the data transmission and propagation time complexity besides the time overhead of the implemented ML models. As a conclusion, well-trained ML models at the levels of local computing and edge of things computing can provide good MSE levels with short execution times.

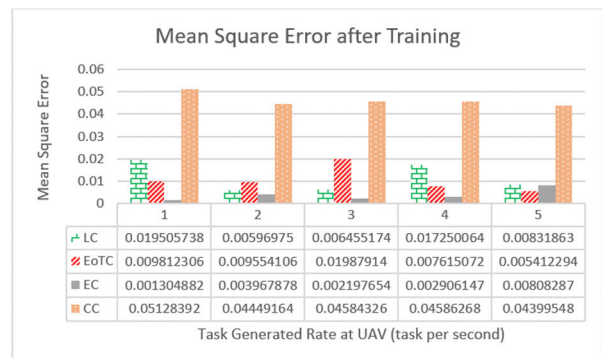


FIGURE 7. Obtained MSE after training and updating ML models.

Figure 10 shows the computation overhead when changing the interdistance between UAVs in the network. Based on the developed system model in Fig. 5, it is noticed that increasing the distance leads to some increase in the computation delay according to the impact of the propagation delay. In the case of using the EC and CC levels, there was an impact when changing the distance where data needs more time to arrive and to be processed.

Scenario (2): a simulation study of the effectiveness of using UAVs with *multilayer neural networks* and adopting different computing architectures to help accomplish UAV sensor quality classification duty in a timely manner.

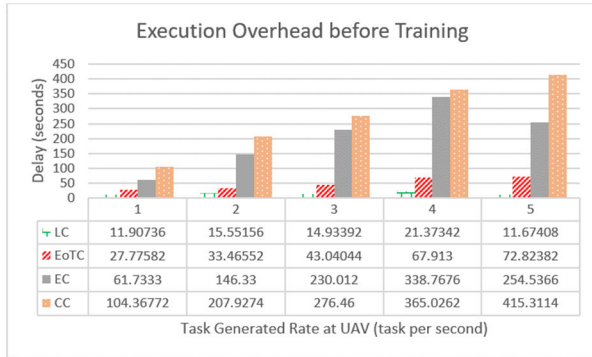


FIGURE 8. Expected delay when applying various computing architectures before training ML models.

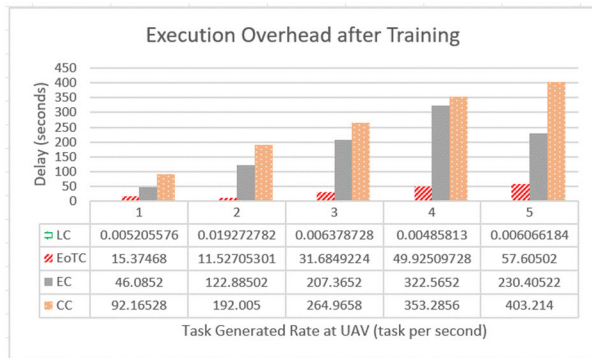


FIGURE 9. Expected delay when applying various computing architectures after training ML models.

We target a duty, which focuses on analyzing and classifying collected data received from three classes of sensors at each UAV. We assume a labeled empirical dataset where it includes three input attributes (sensed parameter, sensor reading, sensor type) and one output (sensor status). Each sensor generates a digital reading within a certain levels range, from 5 to 80. Normal UAVs within the cluster as depicted in Fig. 5 will employ a NN comprising two layers; a linear layer and a logistic sigmoid layer. The logistic sigmoid layer will help in having binary classification for the output, which is normal or abnormal sensor. Based on defined encoders, the data is profiled with various classes. Based on the developed mathematical model, a set of data entries (500 entries) was developed based on collected profiled data from UAV sensors and 70% of these data were used for training and 30% of the data for testing. The master UAV will use an aggregated ML model based on trained models adopted by the UAVs in the cluster. The edge server will get the data from UAVs and its ML model will be trained. The ML at the edge has an extra logistic sigmoid layer to classify the input data and help in categorizing the input data received from UAVs into specific

classes to avoid outliers and abnormal levels. At the cloud, a powerful ML is applied with another NN layer to help in getting finer-tuned relations between input data and output classes. Also, the cloud ML will be trained based on data sent by UAVs.

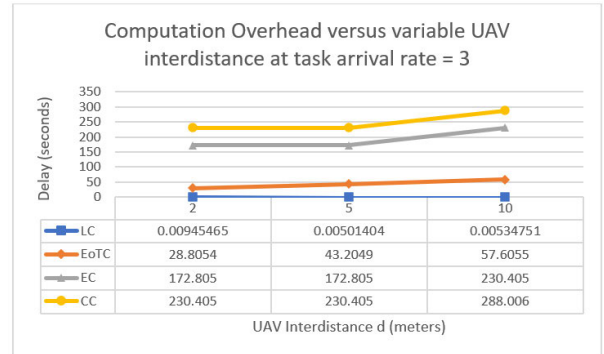


FIGURE 10. Expected Delay when Varying the UAV interdistance at task arrival rate equals 3.

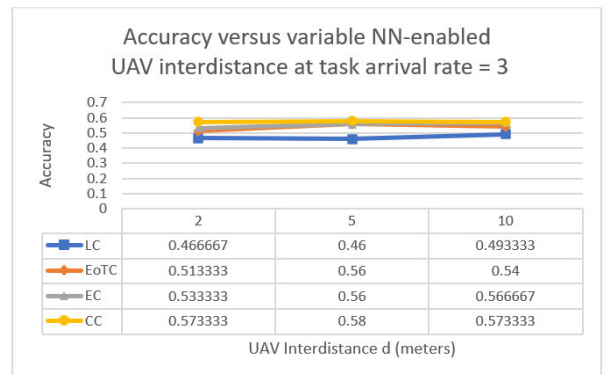


FIGURE 11. Accuracy of ML-based computing tasks at various UAV interdistance.

We target the following performance metrics and the related measures to quantitatively compare the performance of the different computing levels.

1. Time complexity: this metric refers to the time overhead needed by the various computing paradigms to execute the assigned tasks of learning data patterns and classifying the outputs.
 - a. Model computation delay (measured in seconds): the measure refers to the time consumed by the ML models to accomplish the assigned computing tasks.
2. ML model performance: this metric refers to the ML model's capacity to perform efficient classification tasks
 - a. Accuracy: it measures how the ML model often classifies correctly the outcome concerning possible classes of normal and abnormal.
 - b. F1-score: it measures implemented ML models' performance based on the used dataset

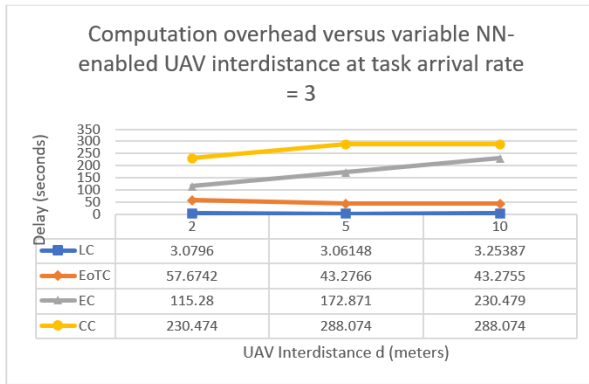


FIGURE 12. Computation time overhead of ML-based Computing Tasks at various UAV interdistance.

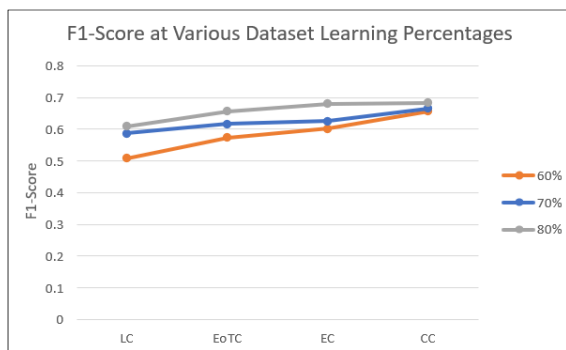


FIGURE 13. F1-Score of the Implemented ML Models at various Computing Paradigms.

We conducted simulation runs for the developed computing resource allocation algorithm. Figures 11 and 12 show the accuracy and execution time overhead of the different computing levels at various UAV interdistance values. Based on the empirical datasets and the used trained MLs, the obtained accuracy when using cloud computing (CC) was better compared with the other computing paradigms, however, the execution time was the longest value, as depicted in Fig. 12. Considering the edge of things computing with collaborative learning at the master UAV, a comparable accuracy level was achieved with shorter execution time.

Figure 13 shows the obtained F1-score of the used ML models adopting different computing paradigms when using various percentages of training data. The results show that using more data points for training will lead to an improvement in the F1-score value. In addition, better results were obtained when adopting the CC computing paradigm, however, with collaborative learning, EoTC with master UAV achieved good levels of F1-score with less execution overhead as depicted in previous figures.

In the conducted scenarios, we targeted performance metrics and related measures to highlight the impact of adopting various computing paradigms on the penalty or cost the users/administrators might pay to get the desired outputs. Running ML models on powerful engines in the cloud will lead to highly accurate results, however, for a long time. However, if there is a well-trained ML model on a

resource-constrained engine (edge of things) on UAV near to the users, a good performance level of results can be obtained within a shorter time if done on the cloud. This is valid if the ML model operates on a lightweight problem with a limited number of classes and a small size of datasets to learn.

V. CONCLUSION

We have introduced a distributed computing architecture for networked UAVs based on collaborative learning and edge-of-things computing. A system model for a networked UAV with clusters was presented. The computing architecture included a) computation at UAVs (local computing); computing at the master UAV or the cluster head UAV (edge of things computing); computing at an edge server (edge computing); and computing at a cloud server (cloud computing). Machine learning (ML) models depending on neural networks were employed at UAVs to help accomplish various data classification tasks. We introduced algorithms for resource allocation and collaborative learning to enable the distributed computing architecture. Using various sensors and ML models at UAVs and edge and cloud servers, the performance analysis for the distributed computing architecture showed good performance when having collaborative learning. We targeted the execution time overhead and ML-related performance measures, such as accuracy and mean squared error.

Future work includes extending the evaluation for the collaborative learning model using explainable AI for developing anomaly detection models and using real-world datasets to detect catastrophic regions, natural disasters, and abnormal conditions in urban areas. In addition, we target real-time delay-sensitive applications to investigate the performance of the proposed computing architecture. Additionally, lightweight ML models (TinyML models) are to be investigated to help optimize the energy consumption and the utilized storage space to provide resource-efficient operations and with acceptable performance levels.

ACKNOWLEDGMENT

This research was funded by a Start-up Research Program—College of Information Technology, United Arab Emirates University (Grant G00004585/12T050).

REFERENCES

- [1] A. Abdelmaboud, "The Internet of Drones: Requirements, taxonomy, recent advances, and challenges of research trends," *Sensors*, vol. 21, no. 17, p. 5718, Aug. 2021, doi: [10.3390/s21175718](https://doi.org/10.3390/s21175718).
- [2] P. S. Bithas, E. T. Michailidis, N. Nomikos, D. Vouyioukas, and A. G. Kanatas, "A survey on machine-learning techniques for UAV-based communications," *Sensors*, vol. 19, no. 23, p. 5170, Nov. 2019, doi: [10.3390/s19235170](https://doi.org/10.3390/s19235170).
- [3] A. Biglari and W. Tang, "A review of embedded machine learning based on hardware, application, and sensing scheme," *Sensors*, vol. 23, no. 4, p. 2131, Feb. 2023, doi: [10.3390/s23042131](https://doi.org/10.3390/s23042131).
- [4] V. M. Suresh, R. Sidhu, P. Karkare, A. Patil, Z. Lei, and A. Basu, "Powering the IoT through embedded machine learning and LoRa," in *Proc. IEEE World Forum Internet Things*, May 2018, pp. 349–354, doi: [10.1109/WF-IoT.2018.8355177](https://doi.org/10.1109/WF-IoT.2018.8355177).
- [5] E. S. Ali, M. K. Hasan, R. Hassan, R. A. Saeed, M. B. Hassan, S. Islam, N. S. Nafi, and S. Bevinakoppa, "Machine learning technologies for secure vehicular communication in Internet of Vehicles: Recent advances and applications," *Secur. Commun. Netw.*, vol. 2021, pp. 1–23, Mar. 2021, doi: [10.1155/2021/8868355](https://doi.org/10.1155/2021/8868355).

- [6] H. V. Abeywickrama, B. A. Jayawickrama, Y. He, and E. Dutkiewicz, "Comprehensive energy consumption model for unmanned aerial vehicles, based on empirical studies of battery performance," *IEEE Access*, vol. 6, pp. 58383–58394, 2018, doi: [10.1109/ACCESS.2018.2875040](https://doi.org/10.1109/ACCESS.2018.2875040).
- [7] B. Ma, Z. Liu, W. Zhao, J. Yuan, H. Long, X. Wang, and Z. Yuan, "Target tracking control of UAV through deep reinforcement learning," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 6, pp. 5983–6000, Jun. 2023, doi: [10.1109/TITS.2023.3249900](https://doi.org/10.1109/TITS.2023.3249900).
- [8] B. Ma, Z. Liu, F. Jiang, W. Zhao, Q. Dang, X. Wang, J. Zhang, and L. Wang, "Reinforcement learning based UAV formation control in GPS-denied environment," *Chin. J. Aeronaut.*, vol. 36, no. 11, pp. 281–296, Nov. 2023, doi: [10.1016/j.cja.2023.07.006](https://doi.org/10.1016/j.cja.2023.07.006).
- [9] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. 20th Int. Conf. Artif. Intell. Statist.*, 2023, pp. 1273–1282. [Online]. Available: <https://proceedings.mlr.press/v54/mcmahan17a.html>
- [10] Y. Qu, H. Dai, Y. Zhuang, J. Chen, C. Dong, F. Wu, and S. Guo, "Decentralized federated learning for UAV networks: Architecture, challenges, and opportunities," *IEEE Netw.*, vol. 35, no. 6, pp. 156–162, Nov. 2021, doi: [10.1109/MNET.001.2100253](https://doi.org/10.1109/MNET.001.2100253).
- [11] M. Cunneen, M. Mullins, and F. Murphy, "Autonomous vehicles and embedded artificial intelligence: The challenges of framing machine driving decisions," *Appl. Artif. Intell.*, vol. 33, no. 8, pp. 706–731, Jul. 2019, doi: [10.1080/08839514.2019.1600301](https://doi.org/10.1080/08839514.2019.1600301).
- [12] S. H. Alsamhi, O. Ma, and M. S. Ansari, "Survey on artificial intelligence based techniques for emerging robotic communication," *Telecommun. Syst.*, vol. 72, no. 3, pp. 483–503, Nov. 2019, doi: [10.1007/s11235-019-00561-z](https://doi.org/10.1007/s11235-019-00561-z).
- [13] R. Xie, Q. Tang, Q. Wang, X. Liu, F. R. Yu, and T. Huang, "Collaborative vehicular edge computing networks: Architecture design and research challenges," *IEEE Access*, vol. 7, pp. 178942–178952, 2019, doi: [10.1109/ACCESS.2019.2957749](https://doi.org/10.1109/ACCESS.2019.2957749).
- [14] H. El-Sayed, S. Sankar, M. Prasad, D. Puthal, A. Gupta, M. Mohanty, and C.-T. Lin, "Edge of things: The big picture on the integration of edge, IoT and the cloud in a distributed computing environment," *IEEE Access*, vol. 6, pp. 1706–1717, 2018.
- [15] C. N. Tadros, M. Gad, B. M. Mokhtar, and M. E. Shimy, "Vehicular edge of thing computing for sustainable smart IoT services," in *Proc. IEEE 7th World Forum Internet Things (WF-IoT)*, Jun. 2021, pp. 55–59, doi: [10.1109/WF-IoT51360.2021.9595826](https://doi.org/10.1109/WF-IoT51360.2021.9595826).
- [16] J.-H. Kim, N. Kim, and C. S. Won, "Deep edge computing for videos," *IEEE Access*, vol. 9, pp. 123348–123357, 2021, doi: [10.1109/ACCESS.2021.3109904](https://doi.org/10.1109/ACCESS.2021.3109904).
- [17] M. D. Hossain, T. Sultana, M. A. Hossain, M. A. Layek, M. I. Hossain, P. P. Sone, G.-W. Lee, and E.-N. Huh, "Dynamic task offloading for cloud-assisted vehicular edge computing networks: A non-cooperative game theoretic approach," *Sensors*, vol. 22, no. 10, p. 3678, May 2022, doi: [10.3390/s22103678](https://doi.org/10.3390/s22103678).
- [18] W. Ding, Z. Yang, M. Chen, J. Hou, and M. Shikh-Bahaei, "Resource allocation for UAV assisted wireless networks with QoS constraints," in *Proc. IEEE Wireless Commun. Netw. Conf.*, May 2020, pp. 1–7, doi: [10.1109/WCNC45663.2020.9120678](https://doi.org/10.1109/WCNC45663.2020.9120678).
- [19] Z. Zhang, F. Xu, Z. Qin, and Y. Xie, "Resource allocation in UAV assisted air ground intelligent inspection system," *Cognit. Robot.*, vol. 2, pp. 1–12, Jan. 2022, doi: [10.1016/j.cogr.2021.12.002](https://doi.org/10.1016/j.cogr.2021.12.002).
- [20] S. Razzaq, C. Xydeas, A. Mahmood, S. Ahmed, N. I. Ratyal, and J. Iqbal, "Efficient optimization techniques for resource allocation in UAVs mission framework," *PLoS ONE*, vol. 18, no. 4, Apr. 2023, Art. no. e0283923, doi: [10.1371/journal.pone.0283923](https://doi.org/10.1371/journal.pone.0283923).
- [21] Y. Yuan, S. Gao, Z. Zhang, W. Wang, Z. Xu, and Z. Liu, "Edge-cloud collaborative UAV object detection: Edge-embedded lightweight algorithm design and task offloading using fuzzy neural network," *IEEE Trans. Cloud Comput.*, vol. 12, no. 1, pp. 306–318, Jan. 2024, doi: [10.1109/TCC.2024.3361858](https://doi.org/10.1109/TCC.2024.3361858).
- [22] M. Ma, Z. Wang, S. Guo, and H. Lu, "Cloud-edge framework for AoI-efficient data processing in multi-UAV-assisted sensor networks," *IEEE Internet Things J.*, vol. 11, no. 14, pp. 25251–25267, Jul. 2024, doi: [10.1109/JIOT.2024.3392244](https://doi.org/10.1109/JIOT.2024.3392244).
- [23] H. Wang, H. Zhao, J. Zhang, D. Ma, J. Li, and J. Wei, "Survey on unmanned aerial vehicle networks: A cyber physical system perspective," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 2, pp. 1027–1070, 2nd Quart., 2020, doi: [10.1109/COMST.2019.2962207](https://doi.org/10.1109/COMST.2019.2962207).
- [24] P. Ravi, M. Wang, and M. J. Scott, "Journal of smart environments and green computing," *J. Smart Environments Green Comput.*, vol. 2, no. 3, pp. 126–142, Sep. 2022, doi: [10.20517/jsegc.2022.02](https://doi.org/10.20517/jsegc.2022.02).
- [25] Y. Zhang, Z. Hu, Z. Wang, X. Wen, and Z. Lu, "Survivability analysis of unmanned aerial vehicle network based on dynamic weighted clustering algorithm with dual cluster heads," *Electronics*, vol. 12, no. 7, p. 1743, Apr. 2023, doi: [10.3390/electronics12071743](https://doi.org/10.3390/electronics12071743).
- [26] O. Bouachir, A. Abrassart, F. Garcia, and N. Larriue, "A mobility model for UAV ad hoc network," in *Proc. Int. Conf. Unmanned Aircr. Syst. (ICUAS)*, May 2014, pp. 383–388, doi: [10.1109/ICUAS.2014.6842277](https://doi.org/10.1109/ICUAS.2014.6842277).
- [27] R. Zhuo, S. Song, and Y. Xu, "UAV communication network modeling and energy consumption optimization based on routing algorithm," *Comput. Math. Methods Med.*, vol. 2022, pp. 1–10, Jun. 2022, doi: [10.1155/2022/4782850](https://doi.org/10.1155/2022/4782850).
- [28] A. Abada, B. Yang, and T. Taleb, "Traffic flow modeling for UAV-enabled wireless networks," in *Proc. Int. Conf. New. Netw. Appl. (NaNA)*, Dec. 2020, pp. 59–64, doi: [10.1109/NaNA51271.2020.00018](https://doi.org/10.1109/NaNA51271.2020.00018).
- [29] A. Arvanitaki and N. Pappas, "Modeling of a UAV-based data collection system," in *Proc. IEEE 22nd Int. Workshop Comput. Aided Model. Design Commun. Links Netw. (CAMAD)*, Jun. 2017, pp. 1–6, doi: [10.1109/CAMAD.2017.8031526](https://doi.org/10.1109/CAMAD.2017.8031526).
- [30] W. A. Nelson, S. R. Yeduri, A. Jha, A. Kumar, and L. R. Cenkeramaddi, "RL-based energy-efficient data transmission over hybrid BLE/LTE/Wi-Fi/LoRa UAV-assisted wireless network," *IEEE/ACM Trans. Netw.*, vol. 32, no. 3, pp. 1951–1966, Jun. 2024, doi: [10.1109/tnet.2023.3332296](https://doi.org/10.1109/tnet.2023.3332296).
- [31] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konečný, S. Mazzocchi, B. McMahan, and T. Van Overveldt, "Towards federated learning at scale: System design," in *Proc. Mach. Learn. Syst.*, vol. 1, 2019, pp. 374–388.
- [32] F. Shang, W. Su, Q. Wang, H. Gao, and Q. Fu, "A location estimation algorithm based on RSSI vector similarity degree," *Int. J. Distrib. Sensor Netw.*, vol. 10, no. 8, Aug. 2014, Art. no. 371350, doi: [10.1155/2014/371350](https://doi.org/10.1155/2014/371350).
- [33] G. E. Athanasiadou and G. V. Tsoulos, "Path loss characteristics for UAV-to-ground wireless channels," in *Proc. 13th Eur. Conf. Antennas Propag. (EuCAP)*, Krakow, Poland, Mar. 2019, pp. 1–4.



BASSEM MOKHTAR (Senior Member, IEEE)

received the B.Sc. and M.Sc. degrees in electrical engineering from Alexandria University, Egypt, in 2004 and 2008, respectively, and the Ph.D. degree in computer engineering from Virginia Tech, Blacksburg, VA, USA. He is currently an Assistant Professor of computer engineering with the College of Information Technology, United Arab Emirates University, Al Ain, United Arab Emirates. He has experience in teaching undergraduate and graduate courses related to computer and electrical engineering and computer science. He has been involved in a set of funded multidisciplinary research projects integrating advances in fields of machine learning-based applications in sensor networks, nanotechnology, and flexible electronics. Also, he participated as a PI, a Co-PI, and a Researcher in a set of funded research projects that cross multi-disciplinary research fields, including the Internet of Things, the Internet of Vehicles, software-defined networking, and flexible electronics. His research interests include embedding intelligence in networking operations, autonomic resilient networking, network semantics reasoning, and machine learning applications in computer networks. In addition, he has been involved as a Reviewer for various conferences and journals, including *IEEE INTERNET OF THINGS JOURNAL*, *IEEE Vehicular Technology Magazine*, and *Journal of Network and Computer Applications* (Elsevier).

...