IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 22 April 2024, accepted 22 June 2024, date of publication 1 July 2024, date of current version 9 July 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3421652

RESEARCH ARTICLE

An Energy-Efficient Intelligence Sharing Scheme in Intelligence Networking-Empowered Edge Computing

JUNFENG XIE[1], (Member, IEEE), QINGMIN JIA[2], AND FENGLIANG LU[1]
[1]School of Information and Communication Engineering, North University of China, Taiyuan 030051, China
[2]Purple Mountain Laboratories, Nanjing 211111, China

Corresponding author: Junfeng Xie (xiejunfeng@nuc.edu.cn)

This work was supported in part by the Research Project supported by Shanxi Scholarship Council of China under Grant 2022-147, in part by the Research Project supported by the Fundamental Research Program of Shanxi Province under Grant 202203021212151 and Grant 202203021221117, in part by the Research Project supported by the National Natural Science Foundation of China under Grant 92367104 and Grant 92267301, and in part by the Research Project supported by the National Key Research and Development Program of China under Grant 2023YFB2704200.

ABSTRACT Advanced artificial intelligence (AI) and multi-access edge computing (MEC) technologies facilitate the development of edge intelligence, which enables the intelligence learned from remote cloud to network edge. To realize automatic decision-making, the accuracy of AI models and training efficiency are crucial for edge intelligence. However, the data volume collected by each network edge node from various sensors is limited, which may cause the over-fitting of AI models. To improve the accuracy of AI models and training efficiency for edge intelligence, intelligence networking-empowered edge computing (INEEC) is emerging as a promising solution, which enables each network edge node to improve its AI models quickly and economically with the help of other network edge nodes' sharing of their learned intelligence. Sharing intelligence among network edge nodes efficiently is essential for INEEC. Thus in this paper, we study the intelligence sharing scheme, which aims to maximize the system energy efficiency while ensuring the latency tolerance via jointly optimizing intelligence requesting strategy, transmission power control and computation resource allocation. The system energy efficiency is defined as the ratio of model performance to energy consumption. To solve this complex optimization problem, a hybrid algorithm that integrates GA and PSO is proposed to make the optimal intelligence sharing decision. Finally, the convergence and superiority of the proposed scheme in terms of intelligence sharing efficiency are evaluated through extensive simulation experiments.

INDEX TERMS Intelligence sharing, intelligence networking, edge computing, GA-PSO.

I. INTRODUCTION

Multi-access edge computing (MEC) and artificial intelligence (AI) are two promising technologies to facilitate the development of smart cities. On the one hand, the advancement of internet of things (IoT) and mobile Internet has triggered the evolution of delay-sensitive computation-intensive mobile applications, such as gesture recognition and augmented reality (AR) [1]. However, the mobile terminal devices (e.g., user equipments (UEs)) generally have limited computing capacity and energy, making it difficult to handle all the computation-intensive tasks locally. Handling these tasks on the cloud can alleviate the pressure on the mobile terminal devices, but lead to high latency due to data transmission from the mobile terminal devices to the remote cloud center. In this case, MEC [2], [3], [4] as a network paradigm has been proposed to deploy IT services and computing capabilities at the network edge (e.g., base stations), which is in close proximity to the mobile terminal devices. On the other hand, to build modern smart cities, more sensors and IoT devices are deployed to sense their environments. The collected massive amounts

The associate editor coordinating the review of this manuscript and approving it for publication was Rajesh Kumar.

of data can be utilized by AI technology to adapt to the dynamic environment and make decisions automatically [5], [6]. Therefore, due to the advantages of MEC and AI, edge intelligence (EI) which integrates the two technologies, has undoubtedly attracted both academia and industry [7], [8], [9]. EI enables the intelligence learned from remote cloud to network edge, leading to perceive the environment quickly and train AI models efficiently.

To realize automatic decision-making, the accuracy of AI models and training efficiency are crucial for edge intelligence. First, applications in mobile terminal devices generally generate similar computation tasks, which may require training the same AI models. Training these AI models independently at the network edge nodes will cause a lot of repetitive training and waste their limited communication, computation and storage resources. Second, the data volume collected by each network edge node from various sensors is limited, which may cause the over-fitting of AI models [10], [11]. The redundant training, limited collected data volume and limited computing capability of network edge nodes make it challenging to guarantee the accuracy of AI models and training efficiency [12].

To solve the above challenges, intelligence networking [13], [14], [15], [16] has been proposed as an emerging technology. Intelligence is an abstraction of AI models. With intelligence networking-empowered edge computing (INEEC), the intelligence learned by each network edge node can be transmitted and shared with other network edge nodes, making it easy to acquire intelligence. This way not only ensures the training efficiency by reducing the number of redundant training, but also improves the accuracy of AI models with the help of other network edge nodes' intelligence.

Intelligence sharing is a key point in INEEC, which includes an intelligence request procedure, a local intelligence training procedure and an intelligence transmission procedure. Regarding intelligence sharing in INEEC, the request decision making of the intelligence request procedure, the computation resource allocation of the local intelligence training procedure, as well as the wireless resource allocation of the intelligence transmission procedure should be well studied.

Therefore, this paper's intent is to investigate an intelligence sharing scheme for INEEC via jointly optimizing intelligence requesting strategy, computation resource allocation and transmission power control. Our objective is to maximize the system energy efficiency related to the AI model performance and energy consumption due to intelligence sharing. Specifically, our main contributions are listed as follows.

- We present the system model for INEEC and formulate the intelligence sharing problem as an optimization problem with constraints, the objective of which is to maximize the system energy efficiency which is defined as the ratio of model performance to energy consumption while ensuring the latency tolerance.
- To solve this problem, considering the advantage of genetic algorithm (GA) in global search capability and the advantage of particle swarm optimization (PSO) in convergence speed, a hybrid algorithm that integrates GA and PSO is proposed to find the feasible intelligence sharing solution, including intelligence requesting strategy, computation resource allocation and transmission power control mechanisms.
- Finally, in the simulation, the convergence of the proposed intelligence sharing scheme is studied, and the performance and superiority of it in terms of intelligence sharing efficiency are also evaluated by comparing with other benchmark schemes.

The rest of this paper is organized as follows. The related works are given in Section II. Section III presents the system model of INEEC and formulates the intelligence sharing problem. To solve the problem, a hybrid GA-PSO algorithm is proposed in Section IV. Section V discusses the numerical simulation results. Finally, this study is concluded in Section VI.

II. RELATED WORKS

Edge intelligence enables the intelligence learned from remote cloud to network edge, leading to perceive the environment quickly and train AI models efficiently. The authors in [17], [18], and [19] survey edge intelligence from different perspectives. In [20], a novel online method is proposed to minimize the energy consumption in processing AIoT tasks. The authors in [21] focus on the integration of AI and edge computing in IoT and present an ICE computing architecture. Another notable work is [22], which designs a framework called CEIF to provide containerized EI inference services and applies a deep Q-learning algorithm to process the requests of EI inference without overloading the edge computing devices. Taking into account the model training and task inference processes, execution latency and energy consumption are minimized in [23] by optimizing the task splitting ratio, the bandwidth allocation and the local CPU frequency. Literature [24] designs a framework called Edgent to accelerate DNN inference through edge computing and device-edge cooperation. In [25], the authors improve the inference accuracy by optimizing the inference jobs offloading scheme.

The redundant training, limited collected data volume and limited computing capability of network edge nodes make it challenging for edge intelligence to guarantee the accuracy of AI models and training efficiency. To cope with these challenges, distributed learning [26] which utilizes the interaction of AI and wireless communications has been proposed. Federated learning (FL) as a popular distributed learning technique has been investigated by many efforts. Literature [27] focuses on data-centric client selection and presents an approach called DICE to improve the accuracy of FL. In [28], the authors consider the client selection and resource allocation optimization to achieve the purpose of

energy minimization. Another notable work is [29], which focuses on the time consumption minimization problem in FL over NOMA networks and proposes an AoU-based client selection scheme. In [30], REINFORCE algorithm is applied to enhance the training efficiency in FL by optimizing client selection and bandwidth allocation. The authors in [31] optimize client selection and bandwidth allocation from a long-term perspective. Literature [32] designs a framework called AUCTION to automatically optimize client selection by applying reinforcement learning. In [33], the hierarchical FL is considered and an online policy called COCS is presented to select clients. The authors in [34] aim to minimize convergence time of FL by optimizing client selection.

Intelligence networking is another approach to cope with the challenges of edge intelligence. Literature [13] first proposes the concept of intelligence networking. In [14], intelligence networking is applied to support the ubiquitous deployments of wireless virtual reality (VR). The authors in [35] design a novel collective DRL algorithm to optimize training iteration selection, intelligence requesting and spectrum resource allocation. In [36], quantum collective learning and many-to-many matching game are applied to optimize the vehicles selection and spectrum resource allocation respectively. The authors in [16] and [37] propose a NFT-based green intelligence networking architecture to support connected and automated vehicles (CAVs) in smart cities. Another notable work is [38], which focuses on the model trading between CAVs in Web3. Driven by intelligence networking, different from these works, in this article, we consider the intelligence sharing problem and aim to maximize the system energy efficiency which is related to both model performance and energy consumption.

III. SYSTEM DESCRIPTION

In this section, we first present the assumptions and definitions of the INEEC system model. Based on the system model, the intelligence sharing problem is formulated as an optimization problem, which jointly optimizes intelligence requesting strategy, transmission power control and computation resource allocation.

A. SYSTEM MODEL

As shown in Figure 1, we consider an INEEC system comprising a set of edge nodes, which connect with each other in wireless manner. Due to the heterogeneous nature of edge nodes, their intelligence sharing willingness is different. The heterogeneities among edge nodes generally include data heterogeneity and resource heterogeneity [30]. Data heterogeneity means that the quantity and quality of data samples collected by edge nodes are different, which influences the performance of AI models [39]. Resource heterogeneity means that the storage, communication (e.g., transmission power and bandwidth) and computation resources (e.g., CPU capacity and performance) of edge nodes are different,

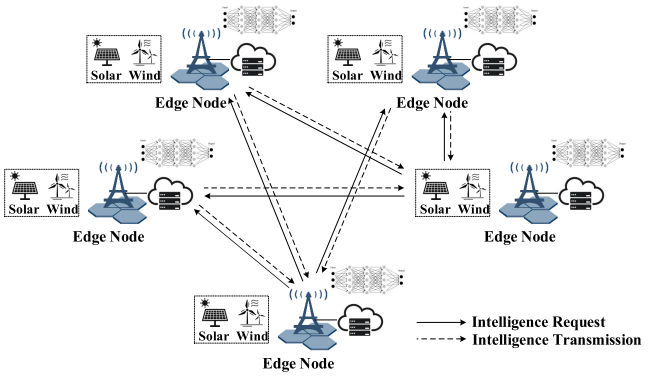


FIGURE 1. System model of intelligence sharing in INEEC.

which influences the training efficiency. In general, when an edge node with sufficient storage, communication and computation resources collects abundant and high-quality data samples, the performance of AI models and the training efficiency can be guaranteed. In this case, the edge node has the ability to share its trained intelligence. On the contrary, when an edge node with very limited resources collects small and poor data samples, the training of AI models may be relatively inefficient. In this case, the edge node tends to request for other edge nodes' AI models to improve its own AI model.

Considering the heterogeneous nature and different intelligence sharing willingness of edge nodes, we assume that each edge node can be either an intelligence requester or an intelligence provider. The intelligence providers share their trained intelligence to the intelligence requesters. Suppose there are M intelligence requesters and N intelligence providers, the set of which are denoted as $\mathcal{M} = \{1, 2, \cdots, m, \cdots, M\}$ and $\mathcal{N} = \{1, 2, \cdots, n, \cdots, N\}$ respectively.

In INEEC, the intelligence providers may share false, irrelevant or useless AI models. Taking into account the potential selfish and malicious behaviours of intelligence providers, trust model [40], [41], [42] is introduced to quantify the reputation of intelligence providers and prevent the unreliable and dishonest intelligence sharing operations. The trust value of intelligence requester m to intelligence provider n is denoted as $R^{trust}_{m\to n}$, which ranges from 0 to 1. When requesting AI models, the intelligence requesters should consider the trust values of the intelligence providers. The intelligence requesters tend to request AI models from the intelligence providers with high trust values. In general, the higher the trust value of an intelligence provider, the higher the probability that an intelligence requester will request AI model from the intelligence provider.

Intelligence sharing in INEEC consists of an intelligence request procedure, a local intelligence training procedure, an intelligence transmission procedure and an intelligence aggregation procedure. In the intelligence request procedure, according to the intelligence providers' status information, such as the trust value, the quantity and quality of their

collected data, as well as wireless channel conditions, the intelligence requesters send requests to the intelligence providers. In the local intelligence training procedure, each intelligence provider allocates computation resources to train local AI model iteratively using its own collected data samples. In the intelligence transmission procedure, each intelligence provider transmits its trained AI model weights to the intelligence requesters over wireless links. In the intelligence aggregation procedure, after receiving AI model weights from corresponding intelligence providers, each intelligence requester performs weighted aggregation based on the received intelligence.

For intelligence providers, the local intelligence training and transmission both consume energy. Taking into account the green communication, energy harvesting (EH) [43], [44], [45] has been proposed to reduce the energy consumption of traditional power grid. In this paper, we consider the EH-based INEEC system. Specifically, in addition to be powered by the traditional power grid, each intelligence provider is equipped with EH devices, which can convert renewable resources (e.g., solar and wind) to electrical energy. A rechargeable battery is used by each intelligence provider to store the renewable electrical energy. Let $b = [b_1, b_2, \cdots, b_n, \cdots, b_N]$ denote the energy level of rechargeable batteries, where b_n is the energy level of intelligence provider n's rechargeable battery, which satisfies $b_n \leq b_n^{\max}$. $b_n^{\max}$ is the maximum capacity of intelligence provider n's rechargeable battery. The intelligence providers can use the renewable green energy for the local intelligence training and transmission, thereby reducing the energy consumption of traditional power grid. To extend the lifetime of rechargeable batteries and avoid the battery exhausted, we assume that when the energy level of rechargeable batteries is lower than a threshold value b^{th} (i.e., $b_n < b^{th}$), the intelligence providers will be powered by the traditional power grid and the rechargeable batteries can not be used until they are charged enough renewable green energy [46], [47].

1) INTELLIGENCE REQUEST PROCEDURE MODEL

In the intelligence request procedure, according to the intelligence providers' status information, such as the trust value, the quantity and quality of their collected data, as well as wireless channel conditions, the intelligence requesters send requests to the intelligence providers. In this paper, we consider the random requesting strategy, also known as probabilistic requesting strategy, where the intelligence requesters send requests to the intelligence providers randomly under certain probability distributions. Let $P = [P_1, P_2, \cdots, P_m, \cdots, P_M]$ denote the random requesting strategy, where P_m is the requesting probability distribution of intelligence requester m. $P_m = \left[P_{m,1}, P_{m,2}, \cdots, P_{m,n}, \cdots, P_{m,N}\right]$, where $P_{m,n}$ is the probability that intelligence requester m sends request to intelligence provider n, which satisfies $0 \leq P_{m,n} \leq 1$. Larger $P_{m,n}$ means that intelligence requester m is more likely to request AI models from intelligence provider n, and vice versa. Therefore, the random requesting strategy $P \in \mathbb{R}^{M \times N}$ has to be determined.

2) LOCAL INTELLIGENCE TRAINING PROCEDURE MODEL

In the local intelligence training procedure, each intelligence provider allocates computation resources to train local AI model iteratively using its own collected data samples. Let $\mathcal{D}_n = \{X_i, Y_i\}_{i=1}^{D_n}$ denote the training data samples of intelligence provider n, where X_i is the input feature vector of ith data sample, Y_i is the corresponding output, D_n represents the number of data samples. Suppose that intelligence provider n trains its local model τ_n times iteratively. The latency of intelligence provider n for the local intelligence training can be calculated by

$$t_n^{train} = \frac{\tau_n \zeta D_n}{F_n} \tag{1}$$

where F_n is the computation resources (i.e., CPU cycle frequency) allocated by intelligence provider n to train its local model, ζ represents the number of CPU cycles needed to process one data sample. Therefore, the energy consumption of intelligence provider n for the local intelligence training can be expressed as

$$E_n^{train} = p_n^{comp} t_n^{train} = e_n \tau_n \zeta D_n F_n^2 \tag{2}$$

where p_n^{comp} is the computing power consumption of intelligence provider n to train its local model. Similar to [48] and [49], we model p_n^{comp} as $p_n^{comp} = e_n F_n^3$, where e_n is the effective capacitance parameter related to the CPU chip of intelligence provider n.

3) INTELLIGENCE TRANSMISSION PROCEDURE MODEL

In the intelligence transmission procedure, each intelligence provider transmits its trained AI model weights to the intelligence requesters over wireless links. Suppose that the bandwidth capacity of intelligence provider n is W_n, which is equally allocated to all intelligence requesters. The transmission rate from intelligence provider n to intelligence requester m can be expressed as

$$R_{n,m} = \frac{W_n}{M} \log_2 \left(1 + \frac{p_n h_{n,m}^2}{\sigma^2}\right) \tag{3}$$

where p_n is the transmission power of intelligence provider n, $h_{n,m}$ is the channel gain between intelligence provider n and intelligence requester m, σ^2 is the noise power. Thus, the latency of intelligence provider n for transmitting its trained AI model weights to intelligence requester m can be calculated by

$$t_{n,m}^{comm} = \frac{s_n}{R_{n,m}} \tag{4}$$

where s_n represents intelligence provider n's local model size. We assume that the sizes of all intelligence providers' local models are same [50], i.e., $s = s_n, \forall n \in \mathcal{N}$. Given the transmission latency, the energy consumption of intelligence

provider n for transmitting its trained AI model weights to intelligence requester m can be represented as

$$E_{n,m}^{comm} = p_n t_{n,m}^{comm} = \frac{p_n s_n}{R_{n,m}} \tag{5}$$

Hence, the total transmission energy consumption of intelligence provider n can be calculated by

$$E_n^{comm} = \sum_{m\in\mathcal{M}} P_{m,n} E_{n,m}^{comm} = \sum_{m\in\mathcal{M}} \frac{P_{m,n} p_n s_n}{R_{n,m}} \tag{6}$$

Note that both traditional power grid and rechargeable batteries can supply energy for the intelligence providers to train and transmit AI models. Because rechargeable batteries store the renewable green energy, in this paper we just focus on the energy consumed from the traditional power grid. Thus, the total energy consumption of intelligence provider n can be expressed as

$$E_n = E_n^{train} + E_n^{comm} - \max\left\{b_n - b^{th}, 0\right\} \tag{7}$$

Given the energy consumption of all intelligence providers, the average energy consumption per intelligence provider can be calculated by

$$\overline{E} = \frac{\sum_{n\in\mathcal{N}} E_n}{N} \tag{8}$$

In addition to energy consumption, the total latency for intelligence provider n to train and transmit AI models can be expressed as

$$t_n = t_n^{train} + \max\left\{t_{n,m}^{comm}, \forall m \in \mathcal{M}\right\} \tag{9}$$

4) INTELLIGENCE AGGREGATION PROCEDURE MODEL

In the intelligence aggregation procedure, after receiving AI model weights from corresponding intelligence providers, each intelligence requester performs weighted aggregation based on the received intelligence. Suppose that ω_n is intelligence provider n's local model weights. Then the aggregated model weights of intelligence requester m can be calculated by

$$\omega_m = \sum_{n\in\mathcal{N}} \frac{\varepsilon_n}{\widehat{\varepsilon_m}} P_{m,n} d_{m,n} R_{m\to n}^{trust} \omega_n \tag{10}$$

where $\widehat{\varepsilon_m} = \sum_{n\in\mathcal{N}} P_{m,n} d_{m,n} R_{m\to n}^{trust} \varepsilon_n$, $d_{m,n}$ is the data correlation between intelligence provider n and intelligence requester m, ε_n is the accuracy of intelligence provider n's local model. According to [51] and [52], ε_n can be calculated by

$$\varepsilon_n = 1 - \exp\left(-\varpi^{lc} F_n \left(Q_n D_n \tau_n{}^{\alpha}\right)^{\nu}\right) \tag{11}$$

where ϖ^{lc} and ν are weight factors, Q_n is the quality level of intelligence provider n's training data samples $\mathcal{D}_n$, α is a constant related to the marginal revenue of iterations. Then similar to [53], the performance of intelligence requester m's aggregated model is expressed as

$$\varphi_m = 1 - \exp\left(-\varpi^{ag} f_m \widetilde{\varepsilon_m}\right) \tag{12}$$

where ϖ^{ag} is a weight factor, f_m is the available computation resources of intelligence requester m. $\widetilde{\varepsilon_m}$ is the average accuracy of the received models, which can be calculated by

$$\widetilde{\varepsilon_m} = \frac{\sum_{n\in\mathcal{N}} P_{m,n} d_{m,n} R_{m\to n}^{trust} \varepsilon_n}{\sum_{n\in\mathcal{N}} P_{m,n} d_{m,n} R_{m\to n}^{trust}} \tag{13}$$

Given the model performance of all intelligence requesters, the average model performance per intelligence requester can be calculated by

$$\overline{\varphi} = \frac{\sum_{m\in\mathcal{M}} \varphi_m}{M} \tag{14}$$

B. PROBLEM FORMULATION

Based on the system model, taking into account the green communication, in this paper, we aim to reduce the intelligence sharing energy consumption while guaranteeing the aggregated model performance of all intelligence requesters. Thus we introduce energy efficiency as a criterion to evaluate the utility of intelligence sharing, which is defined as the ratio of the average model performance to the average energy consumption and can be expressed as

$$U = \Xi \overline{\varphi} / \overline{E} \tag{15}$$

where a positive coefficient Ξ is applied to amplify the effect of $\overline{\varphi}$. Then we formulate the intelligence sharing optimization problem as follows:

$$\max_{P,p,F} U \tag{16}$$

$$\text{s.t. } P_{m,n} \in [0,1], \forall m \in \mathcal{M}, n \in \mathcal{N} \tag{16a}$$

$$p_n \in \left(0, p_n^{\max}\right], \forall n \in \mathcal{N} \tag{16b}$$

$$F_n \in \left(0, F_n^{\max}\right], \forall n \in \mathcal{N} \tag{16c}$$

$$\max\left\{t_n, \forall n \in \mathcal{N}\right\} \leqslant t^{\max} \tag{16d}$$

where (16) is the optimization objective that is to maximize the energy efficiency of the INEEC system by jointly optimizing intelligence requesting strategy denoted as $P \in \mathbb{R}^{M\times N}$, transmission power control denoted as $p = [p_1, p_2, \cdots, p_n, \cdots, p_N]$ and computation resource allocation denoted as $F = [F_1, F_2, \cdots, F_n, \cdots, F_N]$. Constraint (16a) indicates the feasible solution region of variable P. Constraint (16b) indicates that each intelligence provider's transmission power cannot exceed its maximum allowable transmission power $p_n^{\max}$. Constraint (16c) indicates that the computation resources allocated by each intelligence provider cannot exceed its maximum available computation resources $F_n^{\max}$. Constraint (16d) ensures the intelligence sharing latency, which cannot exceed the maximum latency tolerance $t^{\max}$.

IV. PROPOSED ALGORITHM

As mentioned in Section III, in the problem (16), the intelligence requesting strategy, transmission power control and computation resource allocation need to be optimized simultaneously. To solve such a large-scale problem, we adopt meta-heuristic algorithms. GA [54], [55] and PSO [56], [57]

are two widely used meta-heuristic algorithms. But both of them have strengths and weaknesses. The strength of GA is its ability to search global optimal solution. The weakness of GA is its poor convergence speed especially when the solution space is large. The strength of PSO is easy implement and fast convergence speed. The weakness of PSO is premature convergence, which means that it is easy for PSO to fall into a local optimal solution. In order to use strengths of the two algorithms and simultaneously overcome their weaknesses, we combine GA and PSO, and propose a hierarchical GA-PSO based intelligence sharing algorithm. In this section, we first introduce GA and PSO, then the hierarchical GA-PSO based intelligence sharing algorithm is presented.

A. GA

As a popular random searching meta-heuristic algorithm, GA mimics biological evolutionary process and is widely used to find the optimal or near optimal solutions for complex optimization problems. By adopting the principle of survival of the fittest, GA initially generates a set of feasible solutions randomly, and then performs genetic operations (i.e., selection, crossover and mutation) iteratively to search better solutions until the algorithm converges. The crossover and mutation operations can greatly improve population diversity and make GA powerful in global search capability. In the following, the details of GA is described.

Algorithm 1 GA

Initialization:
Initialize the number of individuals in a population, which is denoted as I.
Initialize the maximum number of generations G_1.
Set the generation counter $g_1 = 0$.
for $g_1 = 0, 1, 2, \cdots, G_1$ **do**
  if $g_1 = 0$ **then**
    Initialize the I individuals as the positions of I particles after G_2 generations using Algorithm 2.
  end if
  Evaluate the I individuals by calculating the fitness values of them.
  Find the best individual $B_{best}^{g_1}$ at this generation.
  if $g_1 = 0$ **then**
    Set the historical best individual $B_{best} = B_{best}^{0}$.
  else
    if The fitness value of $B_{best}^{g_1}$ is higher than the fitness value of B_{best} **then**
      Update the historical best individual $B_{best} = B_{best}^{g_1}$.
    end if
  end if
  Perform the selection operation.
  Perform the crossover operation.
  Perform the mutation operation.
  Set $g_1 = g_1 + 1$.
end for

Intelligence requesting strategy P
Transmission power control p
Computation resource allocation F

FIGURE 2. The chromosome of an individual.

1) CHROMOSOME ENCODING

In this paper, we consider real coded GA. In GA, a population consists of several individuals, each of which is defined by a chromosome and represents a feasible solution of the optimization problem (16). Thus as shown in Figure 2, the chromosome of an individual is composed of intelligence requesting strategy P, transmission power control p and computation resource allocation F. The chromosome structure of individual i at generation g is expressed as (17), where $P_{m,n}^{g,i}$ represents the probability that intelligence requester m sends request to intelligence provider n in the individual i at generation g, $p_n^{g,i}$ and $F_n^{g,i}$ respectively represent the transmission power of intelligence provider n and its allocated computation resources in the individual i at generation g.

$$\begin{bmatrix} P_{1,1}^{g,i} & \cdots & P_{1,n}^{g,i} & \cdots & P_{1,N}^{g,i} \\ \vdots & \cdots & \vdots & \vdots & \vdots \\ P_{M,1}^{g,i} & \cdots & P_{M,n}^{g,i} & \cdots & P_{M,N}^{g,i} \\ p_1^{g,i} & \cdots & p_n^{g,i} & \cdots & p_N^{g,i} \\ F_1^{g,i} & \cdots & F_n^{g,i} & \cdots & F_N^{g,i} \end{bmatrix} \tag{17}$$

2) FITNESS FUNCTION

Fitness function is mainly applied to judge the performance of an individual and guide the GA to search better solutions in a right direction. Considering the objective of the optimization problem (16) and its constraint (16d), we define the fitness function as

$$Fitness = \Xi\overline{\varphi}/\overline{E} - \beta\left(\max\left\{\max\left\{t_n, \forall n \in \mathcal{N}\right\} - t^{\max}, 0\right\}\right) \tag{18}$$

where β is the penalty factor.

3) POPULATION INITIALIZATION

In GA, the individuals of the initial population (i.e., the 0th generation) are generated randomly. To make the generated individuals satisfy constraints (16a)-(16c), the genes of initial population's each individual (e.g., the ith individual) are generated based on the following rules.

$$\begin{cases} P_{m,n}^{0,i} = rand\,(1) \\ p_n^{0,i} = rand\left(p_n^{\max}\right) \\ F_n^{0,i} = rand\left(F_n^{\max}\right) \end{cases} \tag{19}$$

where $rand\,(x)$ is a generator function, which outputs a random number between 0 and x (i.e., the interval $[0, x]$).

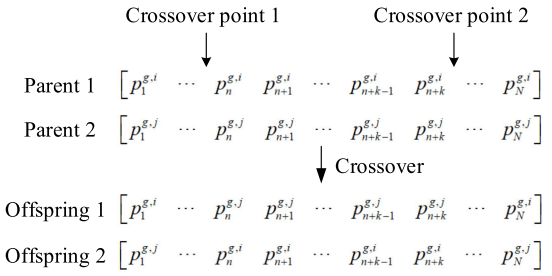


FIGURE 3. An example of the crossover operation.

4) SELECTION OPERATION

The selection operation is done according to the chromosomes' fitness values. In this paper, the tournament method is applied to select several good chromosomes which are kept in the population of the next generation. Compared with the way that only selects the best chromosome, the tournament method maintains the population diversity and is good for the population evolution.

5) CROSSOVER OPERATION

The crossover operation is done by randomly selecting two chromosomes (i.e., parents) and exchanging their corresponding gene segments in each row with the crossover probability p_{cross}. Then two new chromosomes (i.e., offsprings) can be obtained. In this paper, we conduct the standard two-point crossover operation to generate offsprings. An example of the crossover operation for the transmission power control variables is shown in Figure 3. First, two parents (e.g., the ith chromosome and the jth chromosome) and two crossover points are randomly selected. Next, the corresponding gene segments of the ith chromosome and the jth chromosome between the two crossover points are exchanged, which produces two offsprings (i.e., offspring 1 and offspring 2).

6) MUTATION OPERATION

The mutation operation is done by randomly selecting a chromosome (i.e., parent) and mutating its gene values with the mutation probability p_{mutat}. Then a new chromosome (i.e., offspring) can be obtained. Considering the constraints (16a)-(16c), the gene values of the selected chromosome must be mutated within a specific range. The mutation principles are expressed as (20).

$$\begin{cases} P_{m,n}^{g+1,i} = \begin{cases} (1-a_1)\,P_{m,n}^{g,i} + a_1, a_2 > 0.5 \\ (1-a_1)\,P_{m,n}^{g,i}, a_2 \leqslant 0.5 \end{cases} \\ p_n^{g+1,i} = \begin{cases} (1-a_1)\,p_n^{g,i} + a_1 p_n^{\max}, a_2 > 0.5 \\ (1-a_1)\,p_n^{g,i}, a_2 \leqslant 0.5 \end{cases} \\ F_n^{g+1,i} = \begin{cases} (1-a_1)\,F_n^{g,i} + a_1 F_n^{\max}, a_2 > 0.5 \\ (1-a_1)\,F_n^{g,i}, a_2 \leqslant 0.5 \end{cases} \end{cases} \tag{20}$$

where a_1 and a_2 are two random numbers in the interval [0, 1], which make sure that the variables are within the feasible range. The higher the value of a_1, the higher the mutation magnitude. Furthermore, a_2 is applied to control the searching direction. Specifically, when $a_2 > 0.5$, the variable mutates towards its corresponding maximum value, and when $a_2 \leqslant 0.5$, the variable mutates towards its corresponding minimum value.

B. PSO

As another meta-heuristic algorithm, PSO mimics the social behavior of bird migration process. In PSO, a swarm (i.e., a population in GA) consists of several particles, each of which has two features, position and velocity. The position of a particle represents a feasible solution of the optimization problem (16), which is also expressed as (17). The velocity of a particle represents how its solution is evolved. Similar to GA, the PSO starts by generating a random initial position and a random initial velocity for each particle in the swarm. Then the position and velocity of each particle are updated iteratively. Let $\xi_{k,l}$ denote the element of a matrix in the kth row and lth column. The element $\xi_{k,l}$'s values of particle i's position and velocity at generation g are denoted as $Z_{\xi_{k,l}}^{g,i}$ and $V_{\xi_{k,l}}^{g,i}$ respectively. $Z_{\xi_{k,l}}^{g,i}$ is updated by

$$Z_{\xi_{k,l}}^{g+1,i} = Z_{\xi_{k,l}}^{g,i} + V_{\xi_{k,l}}^{g+1,i} \tag{21}$$

$V_{\xi_{k,l}}^{g,i}$ is updated by

$$\begin{aligned} V_{\xi_{k,l}}^{g+1,i} = \theta V_{\xi_{k,l}}^{g,i} &+ r_1 c_1 \left(pbest_{\xi_{k,l}} - Z_{\xi_{k,l}}^{g,i}\right) \\ &+ r_2 c_2 \left(gbest_{\xi_{k,l}} - Z_{\xi_{k,l}}^{g,i}\right) \end{aligned} \tag{22}$$

where θ is an inertia weight, r_1 and r_2 are random factors in the interval [0, 1], c_1 and c_2 are learning coefficients that control the learning ability of particles. *pbest* represents the local best position found by a single particle, and *gbest* represents the global best position found by all particles in the swarm.

C. THE HIERARCHICAL GA-PSO BASED INTELLIGENCE SHARING ALGORITHM

Based on the above description, the GA and PSO algorithms are shown in Algorithm 1 and Algorithm 2 respectively. In order to improve the convergence speed and enhance the global search capability, we combine GA and PSO, and propose a hierarchical GA-PSO based intelligence sharing algorithm, which is shown in Algorithm 3. GA is applied to perform the coarse-grained search, while PSO is applied to perform the fine-grained search. Specifically, Algorithm 3 starts by generating the individuals of the initial population according to (19), and then conducts GA (i.e., Algorithm 1) and PSO (i.e., Algorithm 2) iteratively until convergence. The number of individuals in GA is equal to the number of particles in PSO. When conducting Algorithm 1, the individuals of GA is initialized by the solutions obtained by Algorithm 2. When conducting Algorithm 2, the particles of PSO is initialized by the solutions obtained by Algorithm 1.

Algorithm 2 PSO

Initialization:
Initialize the number of particles in a swarm, which is denoted as I.
Initialize the maximum number of generations G_2.
Set the generation counter $g_2 = 0$.
for $g_2 = 0, 1, 2, \cdots, G_2$ **do**
  if $g_2 = 0$ **then**
    Initialize the positions of I particles as the I individuals after G_1 generations using Algorithm 1.
    Initialize the velocities of I particles.
  end if
  Evaluate the I particles by calculating the fitness values of them.
  for $i = 1, 2, \cdots, I$ **do**
    if $g_2 = 0$ **then**
      Set the local best position of ith particle $pbest^i = Z^{0,i}$.
    else
      if The fitness value of $Z^{g_2,i}$ is higher than the fitness value of $pbest^i$ **then**
        Update the local best position of ith particle $pbest^i = Z^{g_2,i}$.
      end if
    end if
  end for
  Update the global best position $gbest$ according to $pbest$ of all particles.
  Update the positions of I particles according to (21).
  Update the velocities of I particles according to (22).
  Set $g_2 = g_2 + 1$.
end for

V. SIMULATION RESULTS AND DISCUSSIONS

In this section, we use a Python-based simulator to evaluate our proposed hierarchical GA-PSO based intelligence sharing algorithm which is called "GAPSO-ISA". For comparison, the other three benchmark algorithms are considered, i.e., GA based intelligence sharing algorithm which is called "GA-ISA", PSO based intelligence sharing algorithm which is called "PSO-ISA", and random intelligence sharing algorithm which is called "Random-ISA". "GA-ISA" and "PSO-ISA" tackle the problem (16) by adopting GA (i.e., Algorithm 1) and PSO (i.e., Algorithm 2) respectively. In "Random-ISA", the intelligence requesting strategy, transmission power control and computation resource allocation decisions are made randomly. In the simulation, the INEEC system consists of 10 intelligence requesters and 10 intelligence providers. The other important parameters are set as Table 1. In the following, based on the parameter settings, the simulation results are presented.

Figure 4 shows the impacts of three parameters (i.e., the population size I, the crossover probability p_{cross} and the mutation probability p_{mutat}) on the convergence of

Algorithm 3 The Hierarchical GA-PSO Based Intelligence Sharing Algorithm

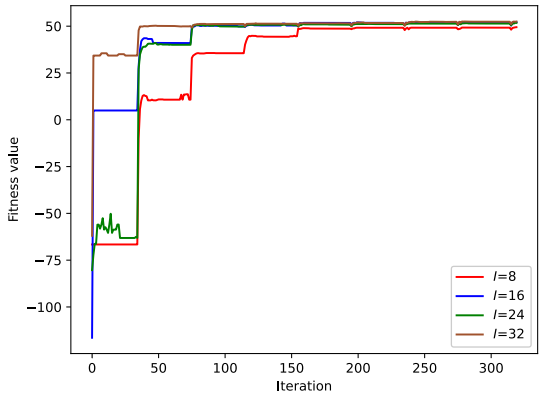
Initialization:
Initialize the number of individuals in a population, which is denoted as I.
Initialize the maximum number of iterations G.
Set the iteration counter $g = 0$.
for $g = 0, 1, 2, \cdots, G$ **do**
  if $g = 0$ **then**
    Generate the I individuals of the initial population according to (19).
  end if
  Perform the coarse-grained search by conducting Algorithm 1.
  Perform the fine-grained search by conducting Algorithm 2.
  Set $g = g + 1$.
end for

TABLE 1. Simulation parameters.

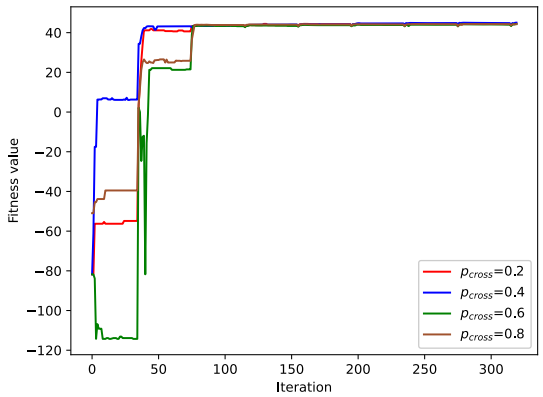
Parameter	Value	Parameter	Value
M	10	D_n	$[100, 300]$
N	10	s_n	$300KB$
$R_{m\to n}^{trust}$	$[0.45, 0.95]$	$p_n^{\max}$	$1W$
b_n	$[0, 400mJ]$ [46]	$F_n^{\max}$	$2GHz$ [58]
b^{th}	$20mJ$ [46]	$d_{m,n}$	$[0.75, 0.95]$
W_n	$20MHz$ [30]	Q_n	$[0.65, 0.95]$
σ^2	$-100dBm$ [27]	θ	0.4 [59]
e_n	10^{-27} [60]	c_1	1.5 [59]
τ_n	5 [30]	c_2	1.5 [59]
ζ	$0.25Gcycles$	I	20

"GAPSO-ISA". As shown in Figure 4(a), "GAPSO-ISA" converges faster with I increasing. This is because the larger the value of I, the higher the population diversity, resulting that less iteration is needed to converge. As shown in Figure 4(b) and 4(c), the convergence performance of "GAPSO-ISA" with a larger p_{cross} and p_{mutat} is not always better than that with a smaller p_{cross} and p_{mutat}. The reason is that the intention of crossover and mutation operations is to improve population diversity but they are done in a random manner. Thus, a larger p_{cross} and p_{mutat} cannot guarantee that a better offspring is always obtained with a larger probability. As a result, a proper value of p_{cross} and p_{mutat} is important. Therefore, we set $I = 20$, $p_{cross} = 0.4$ and $p_{mutat} = 0.08$ for "GAPSO-ISA" in the following simulations.

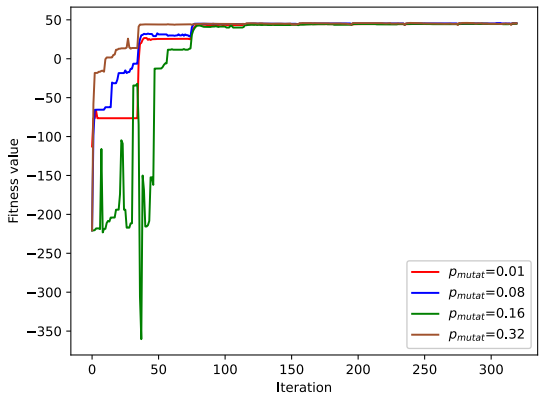
Then the convergence performance of "GAPSO-ISA", "GA-ISA" and "PSO-ISA" is compared. As illustrated in Figure 5, the fitness values of "GAPSO-ISA" and "GA-ISA" gradually increase as the iteration number increases. After about 350 and 500 iterations, the fitness value of "GAPSO-ISA" and "GA-ISA" converges to an approximately stable value respectively, while the fitness values of "PSO-ISA" are all around a local optimum. "PSO-ISA" has the fastest convergence speed, while "GA-ISA" has the slowest convergence speed. Meanwhile, the converged fitness value of



(a) Fitness value versus Iteration with different I



(b) Fitness value versus Iteration with different p_{cross}



(c) Fitness value versus Iteration with different p_{mutat}

FIGURE 4. Convergence of "GAPSO-ISA" with different parameters.

"GAPSO-ISA" is largest, while the converged fitness value of "PSO-ISA" is smallest. This is because the advantage of PSO is fast convergence speed, the advantage of GA is global search capability, both of which are combined in "GAPSO-ISA". Thus, the convergence performance of "GAPSO-ISA" is better than "GA-ISA" and "PSO-ISA".

Figure 6 illustrates the fitness values of "GAPSO-ISA", "GA-ISA" and "PSO-ISA" versus the data quantity D_n from 200 to 400. We observe that the fitness values decrease with D_n increasing. This is because the change of D_n affects

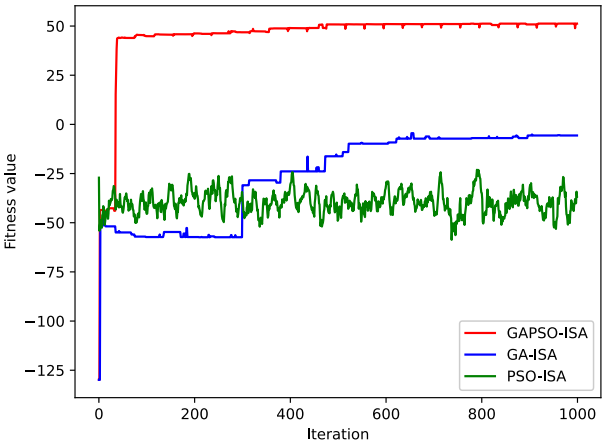


FIGURE 5. Convergence performance of three algorithms.

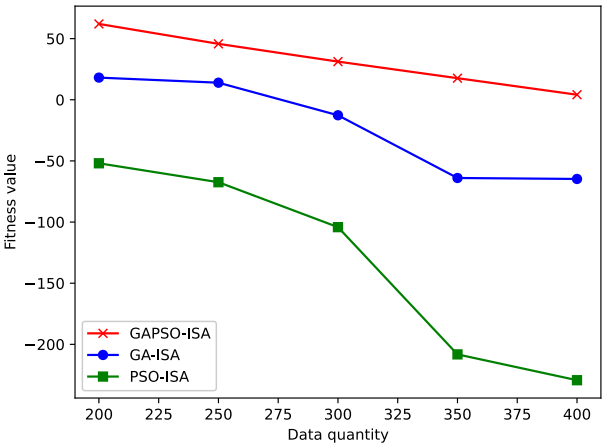


FIGURE 6. Fitness value versus data quantity.

both the average model performance $\overline{\varphi}$ and the average energy consumption $\overline{E}$. Obviously, as the value of D_n increases, the accuracy of an intelligence provider's trained model becomes higher. Meanwhile, the intelligence provider needs to consume more energy to process its data samples. According to Eq. (11), the accuracy of an intelligence provider's trained model increases slower gradually when D_n increases, resulting that the increase of $\overline{\varphi}$ is not capable of covering the increase of $\overline{E}$. Thus, the energy efficiency decreases, which leads to the decrease of fitness values.

In Figure 7, we illustrate the relationship between the fitness value and the data quality Q_n which varies from 0.1 to 0.9. As shown in the figure, the fitness values of "GAPSO-ISA", "GA-ISA" and "PSO-ISA" increase with the value of Q_n increasing. The reason is that according to Eq. (11), the accuracy of an intelligence provider's trained model directly depends on its Q_n. The higher the value of Q_n, the higher the accuracy of the intelligence provider's trained model. When Q_n is high, each intelligence provider can train a better model, leading to the increase of average model performance $\overline{\varphi}$. As a result, the fitness value is improved.

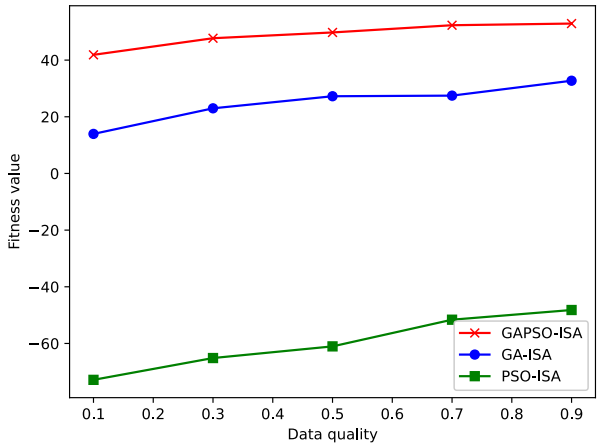


FIGURE 7. Fitness value versus data quality.

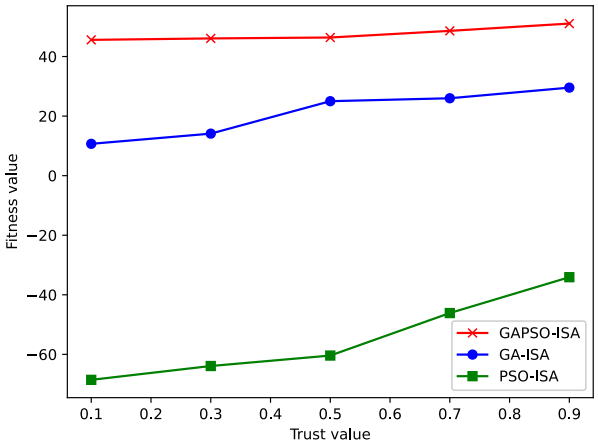


FIGURE 9. Fitness value versus trust value.

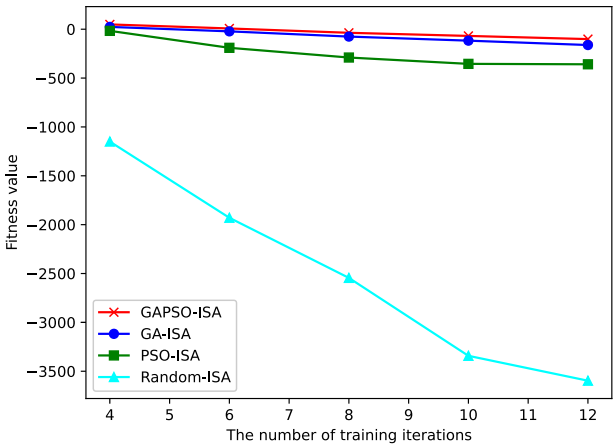


FIGURE 8. Fitness value versus the number of training iterations.

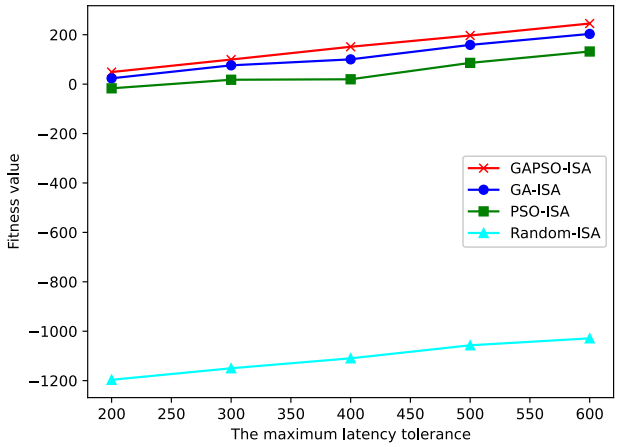


FIGURE 10. Fitness value versus the maximum latency tolerance.

In Figure 8, we show the influence of the number of training iterations τ_n on the fitness values of "GAPSO-ISA", "GA-ISA", "PSO-ISA" and "Random-ISA". τ_n varies from 4 to 12. We observe that with the increase of τ_n, the fitness values decline. This can be explained that higher τ_n can help an intelligence provider to train a model with higher accuracy. Meanwhile, the model training process consumes more energy. As the value of τ_n increases, the increasing speed of the energy consumption is larger than the increasing speed of the model accuracy, which leads to the decrease in energy efficiency. In this case, the fitness values decrease.

Figure 9 illustrates the fitness values of "GAPSO-ISA", "GA-ISA" and "PSO-ISA" versus the trust value $R_{m\to n}^{trust}$ from 0.1 to 0.9. As indicated in the figure, when the value of $R_{m\to n}^{trust}$ increases, the fitness values increase. This is because the performance of an intelligence requester's aggregated model increases with $R_{m\to n}^{trust}$ increasing. Thus, the average model performance $\overline{\varphi}$ is increased, which leads to the increase of fitness values.

In Figure 10, we show the impact of the maximum latency tolerance $t^{\max}$ on the fitness values of "GAPSO-ISA", "GA-ISA", "PSO-ISA" and "Random-ISA". As observed from the figure, the fitness values increase with $t^{\max}$ increasing. This is because when $t^{\max}$ is larger, decreasing the average energy consumption $\overline{E}$ can also satisfy the constraint (16d). As a result, the system tends to improve the energy efficiency by decreasing the average energy consumption $\overline{E}$, resulting in the increase of fitness values.

VI. CONCLUSION

In this paper, we have investigated the energy-efficient intelligence sharing scheme in INEEC system, which considers intelligence requesting strategy, transmission power control and computation resource allocation. The system model is first presented, and the intelligence sharing problem is formulated as a nonlinear optimization problem, the objective of which is to maximize the system energy efficiency while satisfying the latency tolerance. To tackle the optimization problem, a hybrid GA-PSO algorithm is designed to balance the coarse-grained search and the fine-grained search. Finally, compared with other benchmark algorithms, the convergence and superiorities of the designed algorithm in terms of intelligence sharing efficiency are validated through extensive simulation.

REFERENCES

[1] S. Abolfazli, Z. Sanaei, E. Ahmed, A. Gani, and R. Buyya, "Cloud-based augmentation for mobile devices: Motivation, taxonomies, and open challenges," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 1, pp. 337–368, 1st Quart., 2014.

[2] P. Porambage, J. Okwuibe, M. Liyanage, M. Ylianttila, and T. Taleb, "Survey on multi-access edge computing for Internet of Things realization," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 4, pp. 2961–2991, 4th Quart., 2018.

[3] T. Taleb, K. Samdanis, B. Mada, H. Flinck, S. Dutta, and D. Sabella, "On multi-access edge computing: A survey of the emerging 5G network edge cloud architecture and orchestration," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 3, pp. 1657–1681, 3rd Quart., 2017.

[4] X. Jiang, F. R. Yu, T. Song, and V. C. M. Leung, "A survey on multi-access edge computing applied to video streaming: Some research issues and challenges," *IEEE Commun. Surveys Tuts.*, vol. 23, no. 2, pp. 871–903, 2nd Quart., 2021.

[5] Y. Zuo, J. Guo, N. Gao, Y. Zhu, S. Jin, and X. Li, "A survey of blockchain and artificial intelligence for 6G wireless communications," *IEEE Commun. Surveys Tuts.*, vol. 25, no. 4, pp. 2494–2528, 4th Quart., 2023.

[6] E. Baccour, N. Mhaisen, A. A. Abdellatif, A. Erbad, A. Mohamed, M. Hamdi, and M. Guizani, "Pervasive AI for IoT applications: A survey on resource-efficient distributed artificial intelligence," *IEEE Commun. Surveys Tuts.*, vol. 24, no. 4, pp. 2366–2418, 4th Quart., 2022.

[7] X. Wang, X. Ren, C. Qiu, Z. Xiong, H. Yao, and V. C. M. Leung, "Integrating edge intelligence and blockchain: What, why, and how," *IEEE Commun. Surveys Tuts.*, vol. 24, no. 4, pp. 2193–2229, 4th Quart., 2022.

[8] Z. Zhou, X. Chen, E. Li, L. Zeng, K. Luo, and J. Zhang, "Edge intelligence: Paving the last mile of artificial intelligence with edge computing," *Proc. IEEE*, vol. 107, no. 8, pp. 1738–1762, Aug. 2019.

[9] T. Gong, L. Zhu, F. R. Yu, and T. Tang, "Edge intelligence in intelligent transportation systems: A survey," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 9, pp. 8919–8944, Sep. 2023.

[10] M. I. Jordan and T. M. Mitchell, "Machine learning: Trends, perspectives, and prospects," *Science*, vol. 349, no. 6245, pp. 255–260, Jul. 2015.

[11] R. Xie, Q. Tang, S. Qiao, H. Zhu, F. R. Yu, and T. Huang, "When serverless computing meets edge computing: Architecture, challenges, and open issues," *IEEE Wireless Commun.*, vol. 28, no. 5, pp. 126–133, Oct. 2021.

[12] S. Liao, J. Wu, J. Li, A. K. Bashir, S. Mumtaz, A. Jolfaei, and N. Kvedaraite, "Cognitive popularity based AI service sharing for software-defined information-centric networks," *IEEE Trans. Netw. Sci. Eng.*, vol. 7, no. 4, pp. 2126–2136, Oct. 2020.

[13] F. R. Yu, "From information networking to intelligence networking: Motivations, scenarios, and challenges," *IEEE Netw.*, vol. 35, no. 6, pp. 209–216, Nov. 2021.

[14] P. Lin, Q. Song, F. R. Yu, D. Wang, A. Jamalipour, and L. Guo, "Wireless virtual reality in beyond 5G systems with the Internet of Intelligence," *IEEE Wireless Commun.*, vol. 28, no. 2, pp. 70–77, Apr. 2021.

[15] Q. Tang, F. R. Yu, R. Xie, A. Boukerche, T. Huang, and Y. Liu, "Internet of Intelligence: A survey on the enabling technologies, applications, and challenges," *IEEE Commun. Surveys Tuts.*, vol. 24, no. 3, pp. 1394–1434, 3rd Quart., 2022.

[16] Y. Ren, R. Xie, F. R. Yu, T. Huang, and Y. Liu, "NFT-based intelligence networking for connected and autonomous vehicles: A quantum reinforcement learning approach," *IEEE Netw.*, vol. 36, no. 6, pp. 116–124, Nov. 2022.

[17] J. Zhang and K. B. Letaief, "Mobile edge intelligence and computing for the Internet of vehicles," *Proc. IEEE*, vol. 108, no. 2, pp. 246–261, Feb. 2020.

[18] S. Deng, H. Zhao, W. Fang, J. Yin, S. Dustdar, and A. Y. Zomaya, "Edge intelligence: The confluence of edge computing and artificial intelligence," *IEEE Internet Things J.*, vol. 7, no. 8, pp. 7457–7469, Aug. 2020.

[19] D. Xu, T. Li, Y. Li, X. Su, S. Tarkoma, T. Jiang, J. Crowcroft, and P. Hui, "Edge intelligence: Empowering intelligence to the edge of network," *Proc. IEEE*, vol. 109, no. 11, pp. 1778–1837, Nov. 2021.

[20] S. Zhu, K. Ota, and M. Dong, "Energy-efficient artificial intelligence of things with intelligent edge," *IEEE Internet Things J.*, vol. 9, no. 10, pp. 7525–7532, May 2022.

[21] C. Gong, F. Lin, X. Gong, and Y. Lu, "Intelligent cooperative edge computing in Internet of Things," *IEEE Internet Things J.*, vol. 7, no. 10, pp. 9372–9382, Oct. 2020.

[22] L. Nkenyereye, K.-J. Baeg, and W.-Y. Chung, "Deep reinforcement learning for containerized edge intelligence inference request processing in IoT edge computing," *IEEE Trans. Services Comput.*, vol. 16, no. 6, pp. 4328–4344, Dec. 2023.

[23] X. Li, S. Bi, and H. Wang, "Optimizing resource allocation for joint AI model training and task inference in edge intelligence systems," *IEEE Wireless Commun. Lett.*, vol. 10, no. 3, pp. 532–536, Mar. 2021.

[24] E. Li, L. Zeng, Z. Zhou, and X. Chen, "Edge AI: On-demand accelerating deep neural network inference via edge computing," *IEEE Trans. Wireless Commun.*, vol. 19, no. 1, pp. 447–457, Jan. 2020.

[25] A. Fresa and J. P. Champati, "Offloading algorithms for maximizing inference accuracy on edge device in an edge intelligence system," *IEEE Trans. Parallel Distrib. Syst.*, vol. 34, no. 7, pp. 2025–2039, Jul. 2023.

[26] X. Liu, J. Yu, Y. Liu, Y. Gao, T. Mahmoodi, S. Lambotharan, and D. H. Tsang, "Distributed intelligence in wireless networks," *IEEE Open J. Commun. Soc.*, vol. 4, pp. 1001–1039, 2023.

[27] R. Saha, S. Misra, A. Chakraborty, C. Chatterjee, and P. K. Deb, "Data-centric client selection for federated learning over distributed edge networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 34, no. 2, pp. 675–686, Feb. 2023.

[28] M. H. Mahmoud, A. Albaseer, M. Abdallah, and N. Al-Dhahir, "Federated learning resource optimization and client selection for total energy minimization under outage, latency, and bandwidth constraints with partial or no CSI," *IEEE Open J. Commun. Soc.*, vol. 4, pp. 936–953, 2023.

[29] B. Wu, F. Fang, and X. Wang, "Joint age-based client selection and resource allocation for communication-efficient federated learning over NOMA networks," *IEEE Trans. Commun.*, vol. 72, no. 1, pp. 179–192, Jan. 2024.

[30] W. Mao, X. Lu, Y. Jiang, and H. Zheng, "Joint client selection and bandwidth allocation of wireless federated learning by deep reinforcement learning," *IEEE Trans. Services Comput.*, vol. 17, no. 1, pp. 336–348, Feb. 2024.

[31] J. Xu and H. Wang, "Client selection and bandwidth allocation in wireless federated learning networks: A long-term perspective," *IEEE Trans. Wireless Commun.*, vol. 20, no. 2, pp. 1188–1200, Feb. 2021.

[32] Y. Deng, F. Lyu, J. Ren, H. Wu, Y. Zhou, Y. Zhang, and X. Shen, "AUCTION: Automated and quality-aware client selection framework for efficient federated learning," *IEEE Trans. Parallel Distrib. Syst.*, vol. 33, no. 8, pp. 1996–2009, Aug. 2022.

[33] Z. Qu, R. Duan, L. Chen, J. Xu, Z. Lu, and Y. Liu, "Context-aware online client selection for hierarchical federated learning," *IEEE Trans. Parallel Distrib. Syst.*, vol. 33, no. 12, pp. 4353–4367, Dec. 2022.

[34] J. Lee, H. Ko, S. Seo, and S. Pack, "Data distribution-aware online client selection algorithm for federated learning in heterogeneous networks," *IEEE Trans. Veh. Technol.*, vol. 72, no. 1, pp. 1127–1136, Jan. 2023.

[35] Q. Tang, R. Xie, F. R. Yu, T. Chen, R. Zhang, T. Huang, and Y. Liu, "Collective deep reinforcement learning for intelligence sharing in the Internet of Intelligence-empowered edge computing," *IEEE Trans. Mobile Comput.*, vol. 22, no. 11, pp. 6327–634, Nov. 2023.

[36] Y. Ren, R. Xie, F. R. Yu, T. Huang, and Y. Liu, "Quantum collective learning and many-to-many matching game in the metaverse for connected and autonomous vehicles," *IEEE Trans. Veh. Technol.*, vol. 71, no. 11, pp. 12128–12139, Nov. 2022.

[37] Y. Ren, R. Xie, F. R. Yu, T. Huang, and Y. Liu, "Green intelligence networking for connected and autonomous vehicles in smart cities," *IEEE Trans. Green Commun. Netw.*, vol. 6, no. 3, pp. 1591–1603, Sep. 2022.

[38] Y. Ren, R. Xie, F. R. Yu, R. Zhang, Y. Wang, Y. He, and T. Huang, "Connected and autonomous vehicles in Web3: An intelligence-based reinforcement learning approach," *IEEE Trans. Intell. Transp. Syst.*, early access, Jan. 31, 2024, doi: 10.1109/TITS.2024.3355179.

[39] Z. Zhou, S. Yang, L. Pu, and S. Yu, "CEFL: Online admission control, data scheduling, and accuracy tuning for cost-efficient federated learning across edge nodes," *IEEE Internet Things J.*, vol. 7, no. 10, pp. 9341–9356, Oct. 2020.

[40] G. Han, J. Jiang, L. Shu, and M. Guizani, "An attack-resistant trust model based on multidimensional trust metrics in underwater acoustic sensor network," *IEEE Trans. Mobile Comput.*, vol. 14, no. 12, pp. 2447–2459, Dec. 2015.

[41] J. Feng, F. Richard Yu, Q. Pei, X. Chu, J. Du, and L. Zhu, "Cooperative computation offloading and resource allocation for blockchain-enabled mobile-edge computing: A deep reinforcement learning approach," *IEEE Internet Things J.*, vol. 7, no. 7, pp. 6214–6228, Jul. 2020.

[42] J. Kang, Z. Xiong, D. Niyato, D. Ye, D. I. Kim, and J. Zhao, "Toward secure blockchain-enabled Internet of Vehicles: Optimizing consensus management using reputation and contract theory," *IEEE Trans. Veh. Technol.*, vol. 68, no. 3, pp. 2906–2920, Mar. 2019.
[43] K. Suto, H. Nishiyama, and N. Kato, "Postdisaster user location maneuvering method for improving the QoE guaranteed service time in energy harvesting small cell networks," *IEEE Trans. Veh. Technol.*, vol. 66, no. 10, pp. 9410–9420, Oct. 2017.
[44] Y. Wei, F. R. Yu, M. Song, and Z. Han, "User scheduling and resource allocation in HetNets with hybrid energy supply: An actor-critic reinforcement learning approach," *IEEE Trans. Wireless Commun.*, vol. 17, no. 1, pp. 680–692, Jan. 2018.
[45] H. H. Yang, J. Lee, and T. Q. S. Quek, "Heterogeneous cellular network with energy harvesting-based D2D communication," *IEEE Trans. Wireless Commun.*, vol. 15, no. 2, pp. 1406–1419, Feb. 2016.
[46] Z. Zhang, F. R. Yu, F. Fu, Q. Yan, and Z. Wang, "Joint offloading and resource allocation in mobile edge computing systems: An actor-critic approach," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Abu Dhabi, United Arab Emirates, Dec. 2018, pp. 1–6.
[47] Q. Tang, R. Xie, F. R. Yu, T. Huang, and Y. Liu, "Decentralized computation offloading in IoT fog computing system with energy harvesting: A dec-POMDP approach," *IEEE Internet Things J.*, vol. 7, no. 6, pp. 4898–4911, Jun. 2020.
[48] J. Du, W. Liu, G. Lu, J. Jiang, D. Zhai, F. R. Yu, and Z. Ding, "When mobile-edge computing (MEC) meets nonorthogonal multiple access (NOMA) for the Internet of Things (IoT): System design and optimization," *IEEE Internet Things J.*, vol. 8, no. 10, pp. 7849–7862, May 2021.
[49] Y. Mao, J. Zhang, S. H. Song, and K. B. Letaief, "Stochastic joint radio and computational resource management for multi-user mobile-edge computing systems," *IEEE Trans. Wireless Commun.*, vol. 16, no. 9, pp. 5994–6009, Sep. 2017.
[50] M. Chen, H. V. Poor, W. Saad, and S. Cui, "Convergence time optimization for federated learning over wireless networks," *IEEE Trans. Wireless Commun.*, vol. 20, no. 4, pp. 2457–2471, Apr. 2021.
[51] W. Y. B. Lim, Z. Xiong, C. Miao, D. Niyato, Q. Yang, C. Leung, and H. V. Poor, "Hierarchical incentive mechanism design for federated machine learning in mobile networks," *IEEE Internet Things J.*, vol. 7, no. 10, pp. 9575–9588, Oct. 2020.
[52] M. Xu, J. Peng, B. B. Gupta, J. Kang, Z. Xiong, Z. Li, and A. A. A. El-Latif, "Multiagent federated reinforcement learning for secure incentive mechanism in intelligent cyber–physical systems," *IEEE Internet Things J.*, vol. 9, no. 22, pp. 22095–22108, Nov. 2022.
[53] S. R. Pandey, N. H. Tran, M. Bennis, Y. K. Tun, A. Manzoor, and C. S. Hong, "A crowdsourcing framework for on-device federated learning," *IEEE Trans. Wireless Commun.*, vol. 19, no. 5, pp. 3241–3256, May 2020.
[54] H. Jalota and M. Thakur, "Genetic algorithm designed for solving linear or nonlinear mixed-integer constrained optimization problems," in *Proc. Int. Adv. Soft Comput., Intell. Syst. Appl.* Singapore: Springer, 2018, pp. 277–290.
[55] E. S. H. Hou, N. Ansari, and H. Ren, "A genetic algorithm for multiprocessor scheduling," *IEEE Trans. Parallel Distrib. Syst.*, vol. 5, no. 2, pp. 113–120, Feb. 1994.
[56] L. Yiqing, Y. Xigang, and L. Yongjian, "An improved PSO algorithm for solving non-convex NLP/MINLP problems with equality constraints," *Comput. Chem. Eng.*, vol. 31, no. 3, pp. 153–162, Jan. 2007.
[57] D. Wang, D. Tan, and L. Liu, "Particle swarm optimization algorithm: An overview," *Soft Comput.*, vol. 22, no. 2, pp. 387–408, Jan. 2018.
[58] N. H. Tran, W. Bao, A. Zomaya, M. N. H. Nguyen, and C. S. Hong, "Federated learning over wireless networks: Optimization model design and analysis," in *Proc. IEEE INFOCOM*, Paris, France, Apr. 2019, pp. 1387–1395.
[59] F. Guo, H. Zhang, H. Ji, X. Li, and V. C. M. Leung, "An efficient computation offloading management scheme in the densely deployed small cell networks with mobile edge computing," *IEEE/ACM Trans. Netw.*, vol. 26, no. 6, pp. 2651–2664, Dec. 2018.
[60] J. Xie, "Deep Q-learning aided energy-efficient caching and transmission for adaptive bitrate video streaming over dynamic cellular networks," *IEEE Access*, vol. 12, pp. 24232–24242, 2024.

JUNFENG XIE (Member, IEEE) received the B.S. degree in communication engineering from the University of Science and Technology Beijing, in 2013, and the Ph.D. degree from the School of Information and Communication Engineering, Beijing University of Posts and Telecommunications, in 2019. From September 2017 to September 2018, he visited Carleton University, Ottawa, ON, Canada, as a Visiting Ph.D. Student. He is currently an Assistant Professor with the North University of China. His research interests include machine learning, content delivery networks, resource management, and wireless networks.

QINGMIN JIA received the B.S. degree in communication engineering from Qingdao University of Technology, in 2014, and the Ph.D. degree in information and communication engineering from Beijing University of Posts and Telecommunications, in 2019. From July 2019 to May 2020, he was with China Mobile Hangzhou Research and Development Center. He is currently a Researcher with the Future Network Research Center, Purple Mountain Laboratories. His current research interests include edge intelligence, computing and network convergence, and the Industrial Internet of Things.

FENGLIANG LU received the B.S. degree in communication engineering from the North University of China, in 2023, where he is currently pursuing the M.S. degree with the School of Information and Communication Engineering. His research interests include machine learning, information processing and reconstruction, resource management, and wireless networks.