Vehicle Selection and Resource Optimization for Federated Learning in Vehicular Edge Computing

Huizi Xiao, Jun Zhao, Qingqi Pei, Jie Feng, Lei Liu, and Weisong Shi, *Fellow, IEEE*

***Abstract*—As a distributed deep learning paradigm, federated learning (FL) provides a powerful tool for the accurate and efficient processing of on-board data in vehicular edge computing (VEC). However, FL involves the training and transmission of model parameters, which consumes the vehicles' precious energy resources and takes up much time. It is a departure from many applications with severe real-time requirements in VEC. And the capabilities and data quality of each vehicle are distinct that will affect the performance of training the model. Therefore, it is crucial to select the appropriate vehicles to participate in learning tasks and optimize resource allocation under learning time and energy consumption constraints. In this paper, taking the vehicle position and velocity into consideration, we formulate a min-max optimization problem to jointly optimize the on-board computation capability, transmission power, and local model accuracy to achieve the minimum cost in the worst case of FL. Specifically, we propose a greedy algorithm to select vehicles with higher image quality dynamically, and it keeps the system's overall cost to a minimum in FL. The formulated optimization problem is a nonlinear programming problem, so we decompose it into two subproblems. For the resource allocation problem, we use the Lagrangian dual problem and the subgradient projection method to approximate the optimal value iteratively. For the local model accuracy problem, we develop an adaptive harmony algorithm for heuristic search. The simulation results show that our proposed algorithms have well convergence and effectiveness and achieve a tradeoff between cost and fairness.**

***Index Terms*—Vehicular edge computing, federated learning, vehicle selection, resource optimization, local model accuracy.**

I. Introduction

To meet the requirements of the rapid development of autonomous driving technology, sensors, computing units, algorithms, and communication mechanisms have been widely deployed to vehicles. Sensors enable the vehicle to perceive the surrounding environment correctly, which is the basis for autonomous driving [1]. Furthermore, sensors commonly used in vehicles include the camera, radar, inertial measurement unit (IMU), global position system (GPS), and sonar [2]. The global autonomous driving market may grow up to $172.15B by 2030 in [3], which means vehicles will generate massive amounts of sensor data. How to process, utilize, and mine these on-board data effectively is a tricky problem. It is necessary to obtain stable, low-latency, and high-reliability service nearby in applications with strictly real-time requirements such as simultaneous localization and mapping (SLAM), augmented reality (AR) navigation, object tracking, and high-definition (HD) map generation. These applications involve using many deep learning networks to extract accurate decisions from raw data automatically.

Deep learning network provides a robust method for approximating objective functions of various types of values. This technology has become an indispensable part of today's autonomous vehicle system [4]. It provides a powerful tool for the accurate and efficient processing of vehicular data. Nevertheless, these processes require a tremendous amount of computing power, time, and energy. Using a battery-powered vehicle with limited computing resources and strict energy consumption constraints alone is infeasible [5–7]. Moreover, the data in a vehicle is limited, so the obtained model only has local characteristics. These have led to the emergence of a fast growing area called Federated Learning (FL), a distributed deep learning paradigm [8]. It allows vehicles to use local data to train their local deep learning models individually and aggregate them into a global model. Instead of sending their local data directly, vehicles only share their local models while protecting the vehicles' privacy to some extent [9]. Furthermore, this process can integrate a network of global characteristics to realize the sharing of information between vehicles. So this flexible learning method is suitable for autonomous driving.

Vehicular edge computing (VEC) aims to exploit the computation, storage, and communication resources at the edge of vehicular networks [10]. The edge servers such as roadside units and base stations can assist autonomous driving for enabling on-board tasks to meet real-time and reliability requirements [11]. They can connect to the Internet via the backhaul link to utilize cloud computing capabilities to support more complex and comprehensive calculations or coordination for autonomous driving, but this will cause relatively higher delays. By aggregating the local model of various vehicles in the edge server, the new coming vehicle can quickly download the edge network model of the area to monitor the actual road conditions, and then realize real-time localization, lane change, collision warning, traffic light reminding, and a series of safety assisted driving functions. A global deep learning network model can be generated in the cloud by gathering the models of edge servers to provide comprehensive services in a large region, such as lane flow estimation, trajectory route planning, and so on.

FL requires vehicles to download, train, and update the models, which need many computation and communication resources. High-consumption driving not only wastes the pre-

H. Xiao, Q. Pei, J. Feng and L. Liu are with State Key Laboratory of ISN, School of Telecomm. Engineering, Xidian University, Xi'an, Shaanxi, China (email: huizi_xiao@stu.xidian.edu.cn, qqpei@mail.xidian.edu.cn, jiefengcl@163.com, leiliu@xidian.edu.cn.)

J. Zhao is with the School of Computer Science and Engineering, Nanyang Technological University, Singapore.(email: junzhao@ntu.edu.sg.)

W. Shi is with the Department of Computer Science, Wayne State University, Detroit, MI, USA. (email: weisong@wayne.edu)

cious energy of the vehicle, limits the durability of batteries, but also causes heat dissipation problems and increases hardware failures [12]. There are also real-time issues in autonomous driving scenarios. Therefore, resource and delay optimization is necessary and challenging in FL [13]. Although some works have focused on the optimization of resources and time delay for FL, few of them consider VEC scenarios. Due to the many characteristics of autonomous driving, there are still numerous challenges to be solved. Firstly, the continuous movement of vehicles on the road causes service switching between edge servers. How to optimize resources and learning time when the vehicles are still resident in the current server's coverage needs to be solved. In this way, the vehicle will complete the local model training of the current round of FL and update it to the current edge server at least. Secondly, the data obtained by the sensors may suffer from distortion, noise, and inflection in the process of driving the vehicles [14]. Especially for images from on-board cameras, slight jitter will cause severe distortion. These data potentially affect the accuracy and validity of the FL model. How to select vehicles with high image quality to join the training is a problem. Then, each vehicle has a different position and velocity. How to consider them when choosing the vehicles, optimizing resource allocation, and reducing learning time and energy consumption is also a problem. Finally, the model learning time and energy consumption of FL are strongly associated with the model accuracy. The higher the local model's accuracy, the longer the local iteration time and the greater the computational energy consumption, but the number of updates to the edge server will decrease so that the transmission energy consumption be reduced. Whether the model accuracy variables can be optimized to save time and energy.

To provide a solution to the problems mentioned above, we jointly optimize the on-board computation capability, the transmission power of updating the local model, and the model's accuracy for formulating a min-max cost problem. The cost consists of two parts: learning time and energy consumption. We achieve equilibrium optimization by minimizing the worst-case cost in the vehicle set in FL. The contributions of this paper are summarized as follows.

- We formulate a min-max optimization problem to jointly optimize the on-board computation capability, transmission power, and local model accuracy to achieve the minimum cost in the worst case of FL in VEC. Specifically, we take the vehicle position and velocity into consideration so that the vehicles can complete at least one round of FL and update it within the current edge server's coverage area.
- We propose a greedy algorithm for dynamically selecting vehicles so that the vehicles selected for the FL tasks have higher image quality data. And the vehicle with min-max cost of the selected vehicle set is added each time. The cost consists of training time and energy consumption to maintain the optimal overall system performance.
- The formulated optimization problem is a nonlinear programming problem. To solve this problem, we decompose it into two subproblems. For the resource allocation problem, we use the Lagrangian dual problem and the subgradient projection method to approximate the optimal value iteratively. For the local model accuracy problem, we develop an adaptive harmony algorithm for heuristic search.
- The simulation results show that our proposed algorithms have well convergence and effectiveness and achieve fairness to the cost and resource optimization among vehicles. At the same time, the proposed algorithms have a significant performance advantage compared with other algorithms and provide a tradeoff between learning time and energy consumption.

The rest of this paper is organized as follows. Section II survey the related works. In Section III, we introduce the system model and formalize the optimization problem. The resource allocation algorithm is presented in Section IV. We develop an adaptive harmony heuristic search algorithm to obtain local model accuracy and design a greedy vehicles selection algorithm in Section V. The proposed algorithms' convergence and performance are shown in Section VI. We conclude the paper in Section VII.

II. Related Works

There are many works on offloading and allocation of different tasks and resource optimization in VEC. The authors [15] discussed the feasibility of edge computing architectures in different offloading strategies for emerging vehicular computing applications. In [16], the authors formulated the joint optimization problem containing communication, cashing, and computational capacities to minimize the infotainment content retrieval delay for smart cars. A novel solution based on the alternative direction method of multipliers technology can operate in a distributed fashion to solve the relaxed problem. Considering the constraints on quality loss, service latency, and fog capability, Folo [17] is an event-triggered dynamic task allocation framework based on linear programming and binary particle swarm optimization. Chimera [18] is a novel hybrid edge computing architecture for future large-scale vehicular crowdsensing applications to enhance network-wide vehicle resources. In the co-operative automated driving scenario, [19] introduced an edge cloud-enabled and end-to-end vehicle-to-everything framework to support sidelink radio resource management. In [20], M. Shojafar et al. proposed and tested an energy-efficient adaptive resource scheduler for networked fog centers at the edge of links. The paper [21] proposes a multiuser noncooperative computation offloading game to adjust the offloading probability of each vehicle in vehicular multiaccess edge computing networks. It constructed a distributed best response algorithm based on the computation offloading game model to maximize the utility of each vehicle.

A detailed summary of recent FL applications in vehicular networks and current research challenges are provided in [22]. The included research topics are resource management, performance optimization, and applications based on vehicular networks. Considering the communication delays incurred by FL over wireless links, Lyapunov optimization is used to generate the joint power and resource allocation policies enabling ultra-reliable low-latency communication for each vehicular user in

[23]. M. Chen et al. [24] formulated an optimization problem containing user selection, wireless resource allocation, and joint learning to minimize the FL convergence time while optimizing the FL performance. In their other works [25, 26], the above three factors are formulated an optimization problem whose goal is to minimize the loss function that captures the performance of the FL algorithm. They solved it by deriving a closed-form expression of the expected convergence rate of FL loss function to quantify the impact of wireless elements on FL. Under long-term energy constraints, bandwidth allocation and client selection are jointly formulated to a stochastic optimization problem by J. Xu et al. [27], and it can achieve a long-term performance guarantee of FL. With a given total training time in latency constrained wireless FL, W. Shi et al. [28] maximized the model accuracy by jointing resource allocation policy and device scheduling. HFEL[29] is a novel framework in which model aggregation is partially migrated to edge servers from the cloud. Furthermore, the authors formulated a joint computation and communication resource allocation and edge association problem for device users under the hierarchical federated edge learning framework.

All of the above-proposed papers are only the optimization of VEC tasks or the resource allocation of FL. They did not consider the problem of FL in VEC and its resource management. We take the mobility and capabilities of the vehicles, the quality of the on-board data, and the accuracy of the FL model into account for FL training tasks offloading and resource optimization in VEC.

III. System Model

This section elaborated on our system model. First of all, we describe the usage scenario of autonomous driving, speed modeling, and data quality modeling of the vehicles. Then, the process of FL and its particularity used in VEC are discussed, and the FL is divided into the local computing phase and the transmission phase. Finally, we formalize an optimization problem to minimize the vehicle's maximum energy consumption and delay costs. The key notation definitions in system model are summarized in Table I.

A. Connected and Autonomous Vehicles Modeling

We consider autonomous driving in the urban scenario, and there are many edge servers distributed on the roadside, such as roadside units and cellular base stations. As shown in Fig. 1, assuming a set of vehicles on the road covered by an edge server, there are N vehicles, indexed by the set $\mathcal{N} = \{1, 2, 3, \cdots, N\}$. Each vehicle $n \in \mathcal{N}$ be represented $V_n \in \mathcal{V} = \{V_1, V_2, V_3, \cdots, V_N\}$, and they realize the communication with the wireless connected edge server through the rapidly developing C-V2X technology. The number of vehicles passing the edge server per unit of time follows a Poisson process [30]. As shown in the figure, some vehicles are selected to participate in model training. They can train and upload their local models to update the edge model in real-time while using the model. The remaining vehicles only need to download the edge model for neural network inference applications.

1) Speed modeling and residence time: The paper [31] use real traffic data to confirm that vehicle velocities follow Gaussian distribution in steady-state traffic regime (wide moving jam, and free-flow). We assume that the speeds of vehicles to be independent and identically distributed because the driver can choose an appropriate speed by themselves. Also, we assume that the speed changes on a relatively short distance road covered by an edge server are tiny, that is, remain smooth and steady. In order to limit the vehicles' speed to a certain reasonable range, each vehicle generates its velocity v_n randomly from a truncated Gaussian distribution [30, 32, 33], and keeps its assigned speed invariable when it passes an edge server. Therefore, we avoid dealing with negative speeds.

The v_n denotes the constant velocity of vehicle V_n, which is taken between the lower and upper speeds, i.e. $v^{min} \leq v_n \leq v^{max}$. The v^{min} and v^{max} represent the minimum and maximum speeds on the road, respectively. Thus, the distribution of v_n according to the following probability density function:

$$f(v_n) = \begin{cases} \frac{2e^{\frac{-(v_n-\mu)^2}{2\sigma^2}}}{\sigma\sqrt{2\pi}\left(erf\left(\frac{v^{max}-\mu}{\sigma\sqrt{2}}\right)-erf\left(\frac{v^{min}-\mu}{\sigma\sqrt{2}}\right)\right)}, \\ \qquad\qquad v^{min} \leq v_n \leq v^{max}, \\ 0, \qquad\qquad \text{otherwise.} \end{cases} \tag{1}$$

where μ is the mean speed, σ is the standard deviation of the vehicles' speeds and $erf(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-\eta^2} \mathrm{d}\eta$ is the Gauss error function. Let the coverage diameter of the edge server on the road is $\mathcal{L}$, and the distance from the vehicle V_n position to the entrance of the coverage area is l_n. Therefore, the residence time of each vehicle within the coverage of the current edge server can be obtained:

$$\tau_n = \frac{\mathcal{L} - l_n}{v_n}. \tag{2}$$

If the service requested by the vehicle V_n is an inseparable task, we need to ensure that it can be completed with the assistance of the current edge server, that is to say, the residence time τ_n of V_n is the upper time for the completion of the task. Otherwise, continuous edge servers will successively provide seamless services, but it is still necessary to ensure that an atomic subtask is completed within τ_n at least.

2) Image quality: The on-board cameras are the critical hardware for vehicles to obtain perception data, so the images are one of the most basic data types used in many vehicle intelligent applications. Due to vehicles' mobility, the images acquired by the vehicle will generally show noise, motion blur, and distortion [34]. The level of motion blur L_n is related to the instantaneous relative speed $v_n^{'}$ between the vehicle V_n and observation object [35, 36], and have

$$L_n = \frac{v_n^{'} T[f\cos(\delta) - QG\sin(\delta)]}{v_n^{'} TQ\sin(\delta) + dQ}. \tag{3}$$

where T is the exposure time interval, f is the camera focal length, Q is the charge-coupled device (CCD) pixel size in the horizontal direction, G is the starting position of the object in the image (in pixels), δ is the angle between the motion direction and the image plane, d is the perpendicular distance from the starting point of the moving object to the pinhole [35]. We consider the case where the image plane is parallel

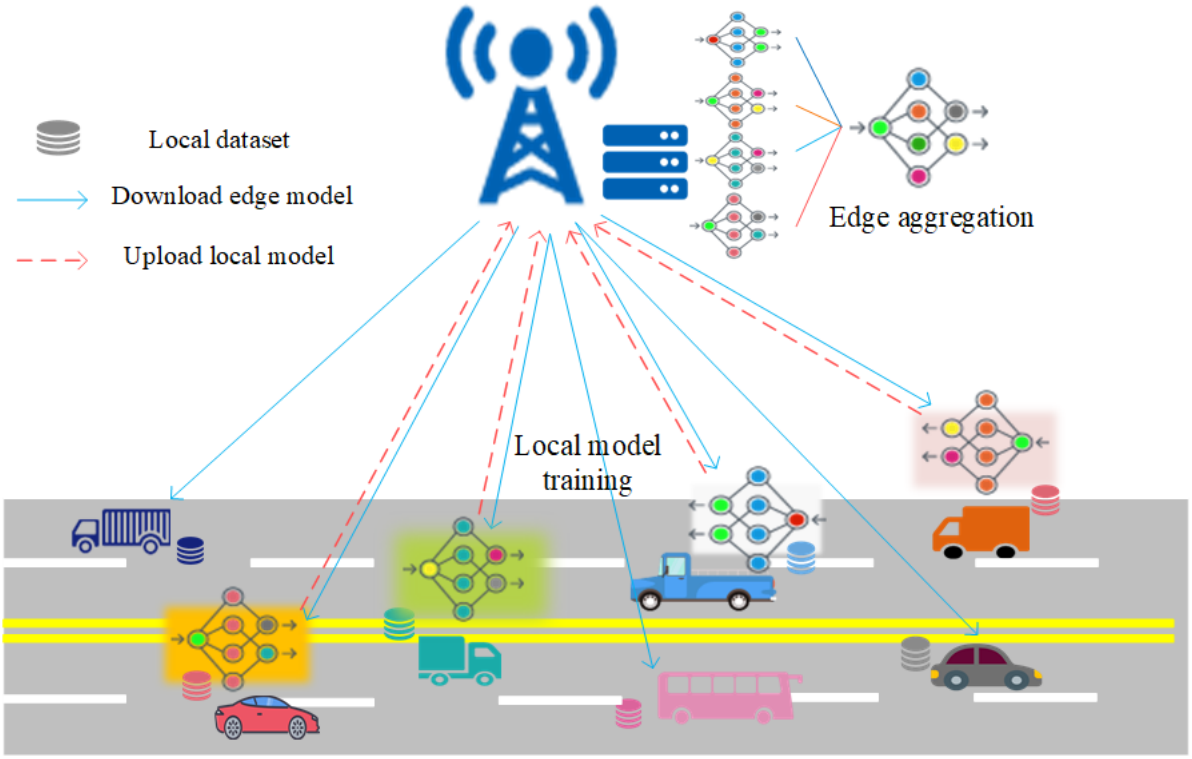

Fig. 1: The system architecture

to the moving direction and then substitute $\delta = 0$ into the equation (3) to get

$$L_n = \frac{Tf}{dQ} v_n'. \tag{4}$$

where $\frac{Tf}{dQ}$ is the parameter of the camera. The equation (4) intuitively shows that the level of motion blur of the image is proportional to the relative speed of the vehicle with the observation object. To conduct the intelligent task accurately, we need higher quality image sampling data to ensure the image motion blur is less than a certain degree.

B. Federated Learning in Vehicular Edge Computing

Each vehicle V_n collects D_n training data samples such as HD images, LiDAR data, and IMU data in our model. And each training data sample consists of an input $\boldsymbol{x}_i$ and its corresponding output $\boldsymbol{y}_i$, which can be represented $D_n = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{|D_n|}$. According to these data, each vehicle will be able to derive a local deep learning model. Nevertheless, the training data samples are too few, and the type is single, so the generalization ability of the local model is relatively weak. FL can take full advantage of such abundant multi-perspective sensor data of these distributed vehicles to derive an edge model in the edge server. Thus, the total size of data can be utilized in the road section is denoted $D = \sum_{n=1}^{N} D_n$.

Before an iteration of the edge model training begins, the vehicles selected to participate in the training download the area's current edge model from the edge. Set $a_n \in \{0, 1\}$ indicate whether the vehicle is selected for model training, if $a_n = 1$, it means V_n is chosen and $a_n = 0$ otherwise. Let $\mathcal{M} \subseteq \mathcal{V}$ be the selected set of vehicles. Since vehicles are frequently moving in the road, the selection of vehicles participating in the training task needs to consider many factors, such as the residence time τ_n in the current edge server coverage, the quality of vehicle collected data L and the vehicle's energy consumption restriction. This allows the selected vehicle $V_n \in \mathcal{M}$ to complete the training of the high-precision model in the road section with lower power consumption.

The whole training process is divided into two sections, local model training and edge aggregation, as shown in Fig. 1. We are taking the k^{th} iteration of the edge server as an example.

1) Local model training: Each vehicle in $\mathcal{M}$ which $a_n = 1$ uses local data set D_n to conduct deep learning training. The goal of V_n is to obtain model parameters $\hat{\boldsymbol{w}}_n^k$ that can fit the local data set very well in the k^{th} edge iteration. This purpose is achieved by minimizing the loss function:

$$\begin{aligned} \hat{\boldsymbol{w}}_n^k &= \arg\min_{\boldsymbol{w}} L_n(\boldsymbol{w}_n^k, \boldsymbol{x}_i, \boldsymbol{y}_i) \\ &= \arg\min_{\boldsymbol{w}} \frac{1}{|D_n|} \sum_{i=1}^{|D_n|} l_n(\boldsymbol{w}_n^k, \boldsymbol{x}_i, \boldsymbol{y}_i). \end{aligned} \tag{5}$$

where $l_n(\cdot)$ is the each vehicle's loss function that evaluates how different the predicted value $\tilde{\boldsymbol{y}}_i = f(\boldsymbol{w}, \boldsymbol{x}_i)$ of the model is from the true one $\boldsymbol{y}_i$. Different deep learning algorithms can use different loss function. For example, the square loss is $l(\tilde{\boldsymbol{y}}_i, \boldsymbol{y}_i) = (\boldsymbol{y}_i - \tilde{\boldsymbol{y}}_i)^2$, the exponential loss is $l(\tilde{\boldsymbol{y}}_i, \boldsymbol{y}_i) = \exp(-\tilde{\boldsymbol{y}}_i \boldsymbol{y}_i)$, the Hinge loss is $l(\tilde{\boldsymbol{y}}_i, \boldsymbol{y}_i) = \max(0, 1 - \tilde{\boldsymbol{y}}_i \boldsymbol{y}_i)$, $\boldsymbol{y}_i \in \{-1, 1\}$. Each vehicle performs the

TABLE I: Key Notation Definitions in System Model

Symbol	Definitions	Symbol	Definitions
$\mathcal{N}$	the number of vehicles covered by an edge server	$\mathcal{V}$	a set of vehicles covered by an edge server
v_n	the speed of the vehicle V_n	v^{min} / v^{max}	the minimum / maximum speed on the road
μ / σ	the mean / standard deviation speed of vehicles	$\mathcal{L}$	the coverage diameter of the edge server on the road
l_n	the distance from the vehicle V_n position to the entrance of the coverage area	τ_n	the residence time of the vehicle V_n within the coverage of the edge server
L_n	the motion blur level of the vehicle V_n	$v_n^{'}$	the instantaneous relative speed between the vehicle V_n and observation object
T	the exposure time interval of the on-board cameras	f	the camera focal length
Q	the charge-coupled device pixel size in the horizontal direction	G	the starting pixel position of the object in the image
δ	the angle between the motion direction and the image plane	d	the perpendicular distance from the starting point of the moving object to the pinhole
D_n	training data samples of the vehicle V_n	$\boldsymbol{x}_i$ / $\boldsymbol{y}_i$	the input / output of the training data sample i
$a_n \in \{0,1\}$	whether the vehicle V_n is selected for model training	$\mathcal{M}$	the selected vehicles set for training task
$\boldsymbol{w}_n^k$	local model weight parameters of the vehicle V_n for the k^{th} edge iteration	$L_n(\boldsymbol{w}_n^k, \boldsymbol{x}_i, \boldsymbol{y}_i)$	the model loss function of the vehicle V_n
λ^k	the learning rate of the k^{th} edge iteration	θ / ε	the accuracy of the local / edge model
$\boldsymbol{w}^k$	edge model weight parameters for the k^{th} edge iteration	$\mathcal{I}(\theta)$	the number of edge model iterations
q_n	the average CPU cycles to process a data sample of the vehicle V_n	f_n	on-board CPU calculated frequency of the vehicle V_n
T_n^{com} / E_n^{com}	local computing time / energy consumption of the vehicle V_n	T_n^{tran} / E_n^{tran}	the time / energy consumption to transmit model parameters of the vehicle V_n
k_n	effective switched capacitance of the vehicle V_n	p_n	the transmit power of the vehicle V_n
r_n	transmitting rate of the vehicle V_n	α_n	the state of the V_n vehicular connection
K_0	the performance of the error-recovery system using Forward Error Correction technology	Z_n	the mobility function of V_n in the TCP/IP mobile connection protocol service
C_n	the data size of the local model parameters in V_n	$\mathcal{S}_n$	the total cost of the vehicle V_n
$\lambda^t, \lambda^e \in [0,1]$	the importance weighting indicators of the delay / energy consumption	T_n / E_n	the whole time / energy consumption of the vehicle V_n for one edge iteration

local parameter update in the k^{th} edge iteration according to the following algorithm to approximate $\hat{\boldsymbol{w}}_n^k$:

$$\boldsymbol{w}_n^k \leftarrow \boldsymbol{w}_n^k - \lambda^k \nabla L_n(\boldsymbol{w}_n^k, \boldsymbol{x}_i, \boldsymbol{y}_i). \tag{6}$$

where λ^k is the learning rate of the k^{th} edge iteration.

When the inequality $||\nabla L_n(\boldsymbol{w}_n^k, \boldsymbol{x}_i, \boldsymbol{y}_i)|| \leq \theta ||\nabla L_n(\boldsymbol{w}_n^{k-1}, \boldsymbol{x}_i, \boldsymbol{y}_i)||$ is met during the iteration process, the iteration stops and the vehicle local model coverage to a local accuracy θ $(0 \leq \theta \leq 1)$. When $\theta = 0$, the model will get a exact solution of the local problem, and $\theta = 1$ means that the local problem has no progress at all. For strongly convex objective $L_n(\boldsymbol{w}_n^k, \boldsymbol{x}_i, \boldsymbol{y}_i)$, the number of local iterations of a wide range of iterative algorithms to solve (5) is upper-bounded by $\mathcal{O}(\log(1/\theta))$ [37], and we can normalize $\mathcal{O}(\log(1/\theta))$ to $\log(1/\theta)$ as the upper bound of local computation iterations [38]. After the vehicle local model converges to the specified accuracy θ, the model parameters $\boldsymbol{w}_n^k$ and $\nabla L_n(\boldsymbol{w}_n^k, \boldsymbol{x}_i, \boldsymbol{y}_i)$ are transmitted to the current edge server through the wireless channel.

2) Edge aggregation: The edge server to generate the new edge model $\boldsymbol{w}^{k+1}$ by computing the weighted average of received local models $\boldsymbol{w}_n^k$ from all connected vehicles set $\mathcal{M}$, so that an edge model can be derived according to the following formula:

$$\boldsymbol{w}^{k+1} \leftarrow \frac{1}{|\mathcal{M}|} \sum_{n=1}^{|\mathcal{M}|} \boldsymbol{w}_n^k. \tag{7}$$

The gradient is aggregated as

$$\nabla L(\boldsymbol{w}^{k+1}, \boldsymbol{x}_i, \boldsymbol{y}_i) \leftarrow \frac{1}{|\mathcal{M}|} \sum_{n=1}^{|\mathcal{M}|} \nabla L_n(\boldsymbol{w}_n^k, \boldsymbol{x}_i, \boldsymbol{y}_i). \tag{8}$$

There are many other update strategies, such as Federated Averaging (FedAVG) and Alternating Direction of Multipliers Method (ADMM). When the inequality $||\nabla L(\boldsymbol{w}^{k+1}, \boldsymbol{x}_i, \boldsymbol{y}_i)|| \leq \varepsilon ||\nabla L(\boldsymbol{w}^k, \boldsymbol{x}_i, \boldsymbol{y}_i)||$ is met during the edge iteration process, the iteration stops and the edge model coverage to an accuracy ε $(0 \leq \varepsilon \leq 1)$. Otherwise, these two parameters are fed back to the newly selected vehicles, the $(k+1)^{th}$ edge iteration starts. Due to strongly convex problems, the number of edge iterations is upper-bounded by $\mathcal{I}(\varepsilon, \theta) = \mathcal{O}(\log(1/\varepsilon))/(1-\theta)$ [39]. For a fixed edge model accuracy ε, we can normalize $\mathcal{O}(\log(1/\varepsilon))$ to 1, so the upper bound of the number of edge iterations $\mathcal{I}(\theta) = 1/(1-\theta)$ can be obtained [38].

C. Local Computation Phase

Let q_n be the average CPU cycles required to process a data sample when the vehicle V_n trains the local model. A total of $|D_n|q_n$ CPU cycles are required to process the training data samples D_n in one local iteration showed in (6). We denote f_n be the local computation capability, which is cycles per second of on-board CPU. Thus, we can derive the time taken for one local iteration is $\frac{|D_n|q_n}{f_n}$. Based on the analysis about the number of iterations in local convergence to ensure the

specified accuracy θ in Section III-B1, the whole time of local computation can be obtained:

$$T_n^{com} = \log(\frac{1}{\theta})\frac{|D_n|q_n}{f_n}. \quad (9)$$

According to [40–42], we model the computation power of on-board CPU as $p_n^{com} = k_n f_n^3$, where k_n represents effective switched capacitance depending on the chip architecture [43]. The energy consumption of V_n for local computation is

$$E_n^{com} = p_n^{com} T_n^{com} = \log(\frac{1}{\theta}) k_n |D_n| q_n f_n^2. \quad (10)$$

We can optimally schedule the supply voltage and clock frequency f_n based on Dynamic Voltage and Frequency Scaling (DVFS) technology to minimize computation energy consumption.

D. Transmission Phase

The communication cost of downloading the model from the edge server in the downlink can be neglected compared to the uplink transmission for vehicle clients. Let p_n be the transmit power of V_n, so the transmitting rate r_n that the wireless TCP/IP can provide in the steady-state [20] is:

$$r_n = \alpha_n \sqrt{p_n}. \quad (11)$$

α_n as the state of the vehicular connection can be defined:

$$\alpha_n \triangleq \frac{K_0 \sqrt{Z_n}}{RRT}. \quad (12)$$

where the positive constant K_0 describe the performance of the error-recovery system using Forward Error Correction (FEC) technology, Z_n is the mobility function of V_n in the TCP/IP mobile connection protocol service, which can be modeled by a time-correlated log-distributed sequence [44]. RRT is the average round-trip-time of the wireless connection, which can be calculated iteratively using Jacobson's formula [45].

We denote the data size of the local model parameters in V_n to be updated as C_n, so the time to transmit them to the current edge server can be derived:

$$T_n^{tran} = \frac{C_n}{r_n} = \frac{C_n}{\alpha_n \sqrt{p_n}}. \quad (13)$$

$p_n = (\frac{r_n}{\alpha_n})^2$ can be obtained by equation (11), so the energy consumption for V_n to transmit local model parameters is:

$$E_n^{tran} = p_n T_n^{tran} = \frac{C_n \sqrt{p_n}}{\alpha_n}. \quad (14)$$

Hereto, we conclude that the whole time and the energy consumption for the vehicle V_n to execute one edge iteration are represented by respectively:

$$T_n = T_n^{com} + T_n^{tran}. \quad (15)$$

$$E_n = E_n^{com} + E_n^{tran}. \quad (16)$$

E. Problem Formulation

We now formulate an optimization problem to minimize the vehicle's maximum energy consumption and delay costs in FL. Specifically, we consider the time delay, transmission energy consumption, vehicle mobility, and the acquisition image quality as constraints. And jointly optimize the vehicle selected set $\mathcal{M}$, local training accuracy θ, on-board CPU calculated frequency $\boldsymbol{f} = (f_1, f_2, \cdots, f_N)$ and transmission power $\boldsymbol{p} = (p_1, p_2, \cdots, p_N)$ to minimize the energy consumption and learning time of the vehicle V_n in the FL. According to the analysis about the number of iterations in edge model convergence to ensure the specified accuracy in Section III-B2, the total cost of V_n can be modeled as the weighted sum of time and energy consumption to converge the edge model, i.e., $\mathcal{S}_n = \mathcal{I}(\theta)(\lambda^t T_n + \lambda^e E_n)$, where $\lambda^t, \lambda^e \in [0,1]$ represent the importance weighting indicators of the delay and energy consumption, respectively. Our optimization problem is organized as follows:

$$\begin{aligned} &\min_{a_n, \theta, \boldsymbol{f}, \boldsymbol{p}} \max_n \mathcal{S}_n \\ &\text{s.t. } (\mathrm{C}_1): a_n \in \{0,1\}, \ \forall n \in \mathcal{N}, \\ &\quad (\mathrm{C}_2): a_n \left(T_n^{com} + T_n^{tran}\right) \le \tau_n, \ \forall n \in \mathcal{N}, \\ &\quad (\mathrm{C}_3): E_n^{tran} \le E_{max}^{tran}, \ \forall n \in \mathcal{N}, \\ &\quad (\mathrm{C}_4): a_n L_n \le L_{max}, \ \forall n \in \mathcal{N}, \\ &\quad (\mathrm{C}_5): 0 \le f_n \le f_n^{max}, \ \forall n \in \mathcal{N}, \\ &\quad (\mathrm{C}_6): 0 \le \theta \le 1. \end{aligned} \quad (17)$$

where E_{max}^{tran}, L_{max} and f_n^{max} are the maximum transmission energy consumption, the level of maximum acceptable image blur for federated learning tasks, and the maximum on-board CPU frequency of the vehicle V_n, respectively. (C_1) indicates whether the vehicle V_n is selected to participate in the federated learning model's training and become an element of the set $\mathcal{M}$. (C_2) restricts the residence time of the selected vehicle within the edge server's coverage area to ensure that at least one local training is completed and the parameters are updated. (C_3), (C_4) and (C_5) confine the transmission energy consumption, the level of image blur and on-board CPU frequency of V_n, respectively. (C_6) is the value range of the local model accuracy.

IV. Design of Resource Allocation Algorithm

In this section, under given the set $\mathcal{M}$ of the selected vehicles V_n with $a_n = 1$ and the local model accuracy θ, we design an algorithm for solving min-max problems to obtain the optimal resource allocation. Thus, we can get on-board CPU frequency $\boldsymbol{f}$ and transmission power $\boldsymbol{p}$ by solving the following problem converted from (17):

$$\begin{aligned} &\min_{\boldsymbol{f}, \boldsymbol{p}} \max_n \mathcal{I}(\theta)(\lambda^t T_n + \lambda^e E_n) \\ &\text{s.t. } (\mathrm{C}_2'): \log(\frac{1}{\theta})\frac{|D_n|q_n}{f_n} + \frac{C_n}{\alpha_n \sqrt{p_n}} \le \tau_n, \ \forall V_n \in \mathcal{M}, \\ &\quad (\mathrm{C}_3'): p_n \le \left(\frac{E_{max}^{tran} \alpha_n}{C_n}\right)^2, \ \forall V_n \in \mathcal{M}, \\ &\quad (\mathrm{C}_5): 0 \le f_n \le f_n^{max}, \ \forall V_n \in \mathcal{M}. \end{aligned} \quad (18)$$

To solve the problem effectively, we first convert it into the epigraph form [46]. A new variable ζ is introduced to transform problem (18) to:

$$\begin{aligned}&\min_{\zeta,\boldsymbol{f},\boldsymbol{p}}\ \zeta\\&\text{s.t. } (\mathrm{C}_2^{'}),(\mathrm{C}_3^{'}),(\mathrm{C}_5),\\&\quad(\mathrm{C}_7):\mathcal{I}(\theta)\left[\lambda^t\left(\log(\frac{1}{\theta})\frac{|D_n|q_n}{f_n}+\frac{C_n}{\alpha_n\sqrt{p_n}}\right)\right.\\&\quad\left.+\lambda^e\left(\log(\frac{1}{\theta})k_n|D_n|q_nf_n^2+\frac{C_n\sqrt{p_n}}{\alpha_n}\right)\right]\le\zeta,\forall V_n\in\mathcal{M}.\end{aligned}\tag{19}$$

The problem is the minimum linear objective function with the convex constraints, which satisfies Slater's condition and the strong duality. The optimal solution of the original problem can be obtained by solving the Lagrangian dual problem with zero dual gap.

A. Resource Allocation Solution

Let $\boldsymbol{\lambda}=\{\lambda_1,\lambda_2,\cdots,\lambda_N\}\succeq 0$, $\boldsymbol{\beta}=\{\beta_1,\beta_2,\cdots,\beta_N\}\succeq 0$, $\boldsymbol{\mu}=\{\mu_1,\mu_2,\cdots,\mu_N\}\succeq 0$, and $\boldsymbol{\varphi}=\{\varphi_1,\varphi_2,\cdots,\varphi_N\}\succeq 0$ be dual variables corresponding to the constraints $(\mathrm{C}_2^{'})$, $(\mathrm{C}_3^{'})$, (C_5) and (C_7) in (19), respectively, so the Lagrangian dual function of problem (19) can be expressed as

$$G(\boldsymbol{\lambda},\boldsymbol{\beta},\boldsymbol{\mu},\boldsymbol{\varphi})=\min_{\zeta,0\preceq\boldsymbol{f},\boldsymbol{p}}\mathscr{L}(\zeta,\boldsymbol{f},\boldsymbol{p},\boldsymbol{\lambda},\boldsymbol{\beta},\boldsymbol{\mu},\boldsymbol{\varphi}).\tag{20}$$

where $\mathscr{L}(\zeta,\boldsymbol{f},\boldsymbol{p},\boldsymbol{\lambda},\boldsymbol{\beta},\boldsymbol{\mu},\boldsymbol{\varphi})$ is the Lagrangian function represented in (21).

Due to the convexity of the problem, we can acquire the optimal solution structures directly by using the Karush-Kuhn-Tucker (KKT) conditions. Regarding other variables as the fixed values, we take the partial derivatives of $\mathscr{L}(\zeta,\boldsymbol{f},\boldsymbol{p},\boldsymbol{\lambda},\boldsymbol{\beta},\boldsymbol{\mu},\boldsymbol{\varphi})$ in (21) with respect to variables f_n and p_n and make them equal to 0. Then the following equations can be obtained respectively:

$$\begin{aligned}\frac{\partial\mathscr{L}}{\partial f_n}=&\mu_n+2\varphi_n\mathcal{I}(\theta)\lambda^e\log(\frac{1}{\theta})k_n|D_n|q_nf_n\\&-\log(\frac{1}{\theta})\frac{|D_n|q_n}{f_n^2}\left(\lambda_n+\lambda^t\varphi_n\mathcal{I}(\theta)\right)=0.\end{aligned}\tag{22}$$

$$\begin{aligned}\frac{\partial\mathscr{L}}{\partial p_n}=&\beta_n+\frac{1}{2}\varphi_n\mathcal{I}(\theta)\lambda^e\frac{C_n}{\alpha_n\sqrt{p_n}}\\&-\frac{1}{2}\frac{C_n}{\alpha_n\sqrt{p_n^3}}\left(\lambda_n+\lambda^t\varphi_n\mathcal{I}(\theta)\right)=0.\end{aligned}\tag{23}$$

1) The optimal solution structure of on-board CPU frequency

Let $A=\mu_n^2$, $B=18\mathcal{I}(\theta)\lambda^e\varphi_nk_n\log^2(\frac{1}{\theta})|D_n|^2q_n^2(\lambda_n+\lambda^t\mathcal{I}(\theta)\varphi_n)$ and $C=3\mu_n\log(\frac{1}{\theta})|D_n|q_n(\lambda_n+\lambda^t\mathcal{I}(\theta)\varphi_n)$. Solving the equation (22), we can get:

$$f_n^*=\begin{cases}\frac{-\mu_n-(\sqrt[3]{Y_1}+\sqrt[3]{Y_2})}{6\mathcal{I}(\theta)\lambda^e\varphi_nk_n\log(\frac{1}{\theta})|D_n|q_n}, & \mu_n\in(0,-(\sqrt[3]{Y_1}+\sqrt[3]{Y_2})]\\ & \cap(0,\Delta),\\ \frac{B}{4A}, & \mu_n=\Delta,\\ \frac{\mu_n(\cos\frac{\arccos G}{3}+\sqrt{3}\sin\frac{\arccos G}{3}-1)}{6\mathcal{I}(\theta)\lambda^e\varphi_nk_n\log(\frac{1}{\theta})|D_n|q_n}, & \mu_n>\Delta,\\ \text{no solution}, & \text{otherwise.}\end{cases}\tag{24}$$

where $Y_1=\mu_n^3+6\mathcal{I}(\theta)\lambda^e\varphi_nk_n\log(\frac{1}{\theta})|D_n|q_n\frac{-B+\sqrt{B^2-4AC}}{2}$, $Y_2=\mu_n^3+6\mathcal{I}(\theta)\lambda^e\varphi_nk_n\log(\frac{1}{\theta})|D_n|q_n\frac{-B-\sqrt{B^2-4AC}}{2}$, $\Delta=3\log(\frac{1}{\theta})|D_n|q_n\sqrt[3]{\mathcal{I}^2(\theta)(\lambda^e)^2\varphi_n^2k_n^2(\lambda_n+\lambda^t\mathcal{I}(\theta)\varphi_n)}$ and $G=\frac{\mu_nA-3\mathcal{I}(\theta)\lambda^e\varphi_nk_n\log(\frac{1}{\theta})|D_n|q_nB}{\mu_n^3}$.

2) The optimal solution structure of transmission power

Let $x_n=\sqrt{p_n}$, $\bar{A}=\frac{1}{4}\mathcal{I}^2(\theta)(\lambda^e)^2\varphi_n^2\frac{C_n^2}{\alpha_n^2}$, $\bar{B}=\frac{9}{2}\beta_n\frac{C_n}{\alpha_n}(\lambda_n+\lambda^t\mathcal{I}(\theta)\varphi_n)$ and $\bar{C}=\frac{3}{4}\mathcal{I}(\theta)\lambda^e\varphi_n\frac{C_n^2}{\alpha_n^2}(\lambda_n+\lambda^t\mathcal{I}(\theta)\varphi_n)$. Solving the equation (23), we can get:

$$x_n^*=\begin{cases}\frac{-\sqrt{\bar{A}}-(\sqrt[3]{\bar{Y}_1}+\sqrt[3]{\bar{Y}_2})}{3\beta_n}, & \Delta_1\in(0,-\frac{2\alpha_n}{C_n}(\sqrt[3]{\bar{Y}_1}+\sqrt[3]{\bar{Y}_2})]\\ & \cap(0,\Delta_2),\\ \frac{\Delta_1(\frac{4}{3}\alpha_n^4-C_n^4)}{2\beta_nC_n^3\alpha_n}, & \Delta_1=\Delta_2,\\ \frac{\sqrt{\bar{A}}(\cos\frac{\arccos\bar{G}}{3}+\sqrt{3}\sin\frac{\arccos\bar{G}}{3}-1)}{3\beta_n}, & \Delta_1>\Delta_2,\\ \text{no solution}, & \text{otherwise.}\end{cases}\tag{25}$$

where $\bar{Y}_1=\sqrt{\bar{A}^3}+3\beta_n\frac{-\bar{B}+\sqrt{\bar{B}^2-4\bar{A}\bar{C}}}{2}$, $\bar{Y}_2=\sqrt{\bar{A}^3}+3\beta_n\frac{-\bar{B}-\sqrt{\bar{B}^2-4\bar{A}\bar{C}}}{2}$, $\Delta_1=\mathcal{I}(\theta)\lambda^e\varphi_n$, $\Delta_2=3\sqrt[3]{\beta_n^2\frac{\alpha_n^2}{c_n^2}(\lambda_n+\lambda^t\mathcal{I}(\theta)\varphi_n)}$ and $\bar{G}=\frac{2\sqrt{\bar{A}^3}-3\beta_n\bar{B}}{2\sqrt{\bar{A}^3}}$. Then, we will get the optimal solution of transmission power $p_n^*=(x_n^*)^2$.

3) The optimal solution structure of ζ

The optimal solution structure of f_n^* and p_n^* have been obtained under given the set $\mathcal{M}$ of the selected vehicles and the local model accuracy θ. According to the objective formula and constriction formula (C_7) in (19), the optimal solution of ζ can be given by:

$$\begin{aligned}\zeta^*=\max_n\ \mathcal{I}(\theta)&\left[\lambda^t\left(\log(\frac{1}{\theta})\frac{|D_n|q_n}{f_n^*}+\frac{C_n}{\alpha_n\sqrt{p_n^*}}\right)\right.\\&\left.+\lambda^e\left(\log(\frac{1}{\theta})k_n|D_n|q_n(f_n^*)^2+\frac{C_n\sqrt{p_n^*}}{\alpha_n}\right)\right].\end{aligned}\tag{26}$$

B. Lagrange Multipliers Update

If the Lagrange dual variables $\boldsymbol{\lambda}$, $\boldsymbol{\beta}$, $\boldsymbol{\mu}$ and $\boldsymbol{\varphi}$ are known in advance, we can obtain the optimal solutions of f_n^* and p_n^* from (24) and (25). The problem in (27) is the Lagrangian dual problem of original problem, the Lagrange dual variables can be obtained by solving it.

$$\begin{aligned}&\max_{\boldsymbol{\lambda},\boldsymbol{\beta},\boldsymbol{\mu},\boldsymbol{\varphi}}\ G(\boldsymbol{\lambda},\boldsymbol{\beta},\boldsymbol{\mu},\boldsymbol{\varphi})\\&\quad\text{s.t. }\boldsymbol{\lambda}\succeq 0,\boldsymbol{\beta}\succeq 0,\boldsymbol{\mu}\succeq 0,\boldsymbol{\varphi}\succeq 0.\end{aligned}\tag{27}$$

$G(\boldsymbol{\lambda},\boldsymbol{\beta},\boldsymbol{\mu},\boldsymbol{\varphi})$ is the linear relationship with respect to $\boldsymbol{\lambda}$, $\boldsymbol{\beta}$, $\boldsymbol{\mu}$ and $\boldsymbol{\varphi}$ by observing (21). Due to the convexity of the problem, the subgradient projection method can be applied to update the values to approximate the optimal one. The subgradient of (27) can be calculated by the following theorem.

Theorem 1 : The problem in (27) is the Lagrangian dual problem of original problem (19). A set of subgradients of

$$\mathscr{L}(\zeta, \boldsymbol{f}, \boldsymbol{p}, \boldsymbol{\lambda}, \boldsymbol{\beta}, \boldsymbol{\mu}, \boldsymbol{\varphi}) = \zeta + \sum_{V_n \in \mathcal{M}} \lambda_n \left[\log(\frac{1}{\theta}) \frac{|D_n| q_n}{f_n} + \frac{C_n}{\alpha_n \sqrt{p_n}} - \tau_n \right] + \sum_{V_n \in \mathcal{M}} \beta_n \left[p_n - \left(\frac{E_{max}^{tran} \alpha_n}{C_n} \right)^2 \right]$$
$$+ \sum_{V_n \in \mathcal{M}} \mu_n (f_n - f_n^{max}) + \sum_{V_n \in \mathcal{M}} \varphi_n \left[\mathcal{I}(\theta) \left(\lambda^t \log(\frac{1}{\theta}) \frac{|D_n| q_n}{f_n} + \lambda^t \frac{C_n}{\alpha_n \sqrt{p_n}} + \lambda^e \log(\frac{1}{\theta}) k_n |D_n| q_n f_n^2 + \lambda^e \frac{C_n \sqrt{p_n}}{\alpha_n} \right) - \zeta \right] \tag{21}$$

$G(\boldsymbol{\lambda}, \boldsymbol{\beta}, \boldsymbol{\mu}, \boldsymbol{\varphi})$ are given by the following equation:

$$\begin{aligned}
\bigtriangledown \lambda_n &= \log(\frac{1}{\theta}) \frac{|D_n| q_n}{f_n^*} + \frac{C_n}{\alpha_n \sqrt{p_n^*}} - \tau_n, \ \forall V_n \in \mathcal{M} \\
\bigtriangledown \beta_n &= p_n^* - \left(\frac{E_{max}^{tran} \alpha_n}{C_n} \right)^2, \ \forall V_n \in \mathcal{M} \\
\bigtriangledown \mu_n &= f_n^* - f_n^{max}, \ \forall V_n \in \mathcal{M} \\
\bigtriangledown \varphi_n &= \mathcal{I}(\theta) \left(\lambda^t \log(\frac{1}{\theta}) \frac{|D_n| q_n}{f_n^*} + \lambda^t \frac{C_n}{\alpha_n \sqrt{p_n^*}} \right. \\
&\quad \left. + \lambda^e \log(\frac{1}{\theta}) k_n |D_n| q_n (f_n^*)^2 + \lambda^e \frac{C_n \sqrt{p_n^*}}{\alpha_n} \right) - \zeta, \forall V_n \in \mathcal{M}
\end{aligned} \tag{28}$$

where f_n^* and p_n^* are the optimal solutions from (24) and (25) under given Lagrange multipliers λ_n, β_n, μ_n and φ_n.

proof: See Appendix A.

From (28), the subgradients of $G(\boldsymbol{\lambda}, \boldsymbol{\beta}, \boldsymbol{\mu}, \boldsymbol{\varphi})$ can be obtained. The subgradient projection method for acquiring Lagrange multipliers to solve (27) is iterated in the following way:

$$\begin{aligned}
\lambda_n(t+1) &= [\lambda_n(t) - i(t) \bigtriangledown \lambda_n(t)]^+, \\
\beta_n(t+1) &= [\beta_n(t) - j(t) \bigtriangledown \beta_n(t)]^+, \\
\mu_n(t+1) &= [\mu_n(t) - k(t) \bigtriangledown \mu_n(t)]^+, \\
\varphi_n(t+1) &= [\varphi_n(t) - o(t) \bigtriangledown \varphi_n(t)]^+.
\end{aligned} \tag{29}$$

where $[y]^+ \triangleq \max\{0, y\}$, t is the subscript of the number of iterations, $i(t)$, $j(t)$, $k(t)$ and $o(t)$ are small positive step size. The step size setting method is the mean square summation bounded, but the direct summation is unbounded [47]. Specifically, we set $i(t) = j(t) = k(t) = o(t) = \frac{0.1}{t}$ [48]. We update the Lagrangian dual variables λ_n, β_n, μ_n and φ_n according to the iterative method in (29), and then substitute the known dual variables into (24) and (25) to obtain the optimal solutions of f_n and p_n. The optimal solutions can be brought to (28) and (29) to obtain the new dual variables. This is a cyclic iterative approximation process, which is explained in Algorithm 1.

Algorithm 1 On-Board CPU Frequency and Transmission Power Optimization Algorithm

Initialization:

- Set the initial value of dual variables to $\boldsymbol{\lambda}(0)$, $\boldsymbol{\beta}(0)$, $\boldsymbol{\mu}(0)$, $\boldsymbol{\varphi}(0)$, maximum number of iterations t_{max} and the specified precision ϵ.
- Let $t = 0$.

Iteration:

while $t \leq t_{max}$ **do**
  Substitute the dual variables $\boldsymbol{\lambda}(t)$, $\boldsymbol{\beta}(t)$, $\boldsymbol{\mu}(t)$ and $\boldsymbol{\varphi}(t)$ into (24) and (25) to obtain $f_n(t)$ and $p_n(t)$ respectively.
  Update new dual variables $\boldsymbol{\lambda}(t+1)$, $\boldsymbol{\beta}(t+1)$, $\boldsymbol{\mu}(t+1)$ and $\boldsymbol{\varphi}(t+1)$ using (29), according to the new $f_n(t)$ and $p_n(t)$.
  if $||\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}(t)|| < \epsilon$, $||\boldsymbol{\beta}(t+1) - \boldsymbol{\beta}(t)|| < \epsilon$, $||\boldsymbol{\mu}(t+1) - \boldsymbol{\mu}(t)|| < \epsilon$ and $||\boldsymbol{\varphi}(t+1) - \boldsymbol{\varphi}(t)|| < \epsilon$ **then**
    $f_n^* = f_n(t)$ and $p_n^* = p_n(t)$.
    break.
  else
    $t = t + 1$.
  end if
end while

Output: $\boldsymbol{f}^* = (f_1^*, f_2^*, \cdots, f_N^*)$ and $\boldsymbol{p}^* = (p_1^*, p_2^*, \cdots, p_N^*)$.

V. Local Model Accuracy and Selection of Vehicles

In this section, the on-board CPU frequency $\boldsymbol{f}^*$ and transmission power $\boldsymbol{p}^*$ obtained from the above section are regarded as fixed values, and we design the heuristic search algorithms to solve the min-max problem. Thus, we can get the set $\mathcal{M}$ of the selected vehicles V_n with $a_n = 1$ and the local model accuracy θ.

A. Local Model Accuracy Optimization Algorithm

Under given the set $\mathcal{M}$ of the selected vehicles and having $\boldsymbol{f}$ and $\boldsymbol{p}$, we transform problem (17) into the following form:

$$\begin{aligned}
&\min_{\theta} \ \max_{n} \ \frac{\log(\frac{1}{\theta}) H_n + G_n}{1 - \theta} \\
&\text{s.t. } (\bar{\mathrm{C}}_2) : \log(\frac{1}{\theta}) \frac{|D_n| q_n}{f_n} \leq \tau_n^{\theta}, \ \forall V_n \in \mathcal{M}, \\
&\quad (\mathrm{C}_6) : 0 \leq \theta \leq 1.
\end{aligned} \tag{30}$$

where $H_n = \lambda^t \frac{|D_n| q_n}{f_n} + \lambda^e k_n |D_n| q_n f_n^2$, $G_n = \lambda^t \frac{C_n}{\alpha_n \sqrt{p_n}} + \lambda^e \frac{C_n \sqrt{p_n}}{\alpha_n}$ and $\tau_n^{\theta} = \tau_n - \frac{C_n}{\alpha_n \sqrt{p_n}}$ are calculated from known values, which can be regarded as constants.

Constructing penalty function from constraint $(\bar{\mathrm{C}}_2)$ which can be transformed into unconstrained, and the following problem can be obtained by rearranging:

$$\begin{aligned}
&\min_{\theta} \ F(\theta) \\
&\text{s.t. } (\mathrm{C}_6) : 0 \leq \theta \leq 1.
\end{aligned} \tag{31}$$

when penalty factor $\gamma \to 0$, we set the function $F(\theta) = \max_n \frac{\log(\frac{1}{\theta}) H_n + G_n}{1 - \theta} + \frac{1}{\gamma} \sum_{V_n \in \mathcal{M}} \max\{0, \log(\frac{1}{\theta}) \frac{|D_n| q_n}{f_n} - \tau_n^{\theta}\}$ [49].

Algorithm 2 The Computational Procedure of the Self-adaptive Global Best Harmony Search (SGHS) algorithm

Initialization:

- Set the parameters HMS, NI, UP.
- Initialize the update parameters of the main parameters BW_{min}, BW_{max}, μ_{HMCR}, σ^2_{HMCR}, μ_{PAR} and σ^2_{PAR}.
- Initialize the HM and evaluate it using $F(\theta)$.
- Set $t_{NI} = 1$, $t_{UP} = 1$, and $r_1, r_2, r_3 \in (0, 1)$.

Iteration:

while $t_{NI} \leq NI$ **do**
  Generate $HMCR \leftarrow \mathcal{N}(\mu_{HMCR}, \sigma^2_{HMCR})$, $PAR \leftarrow \mathcal{N}(\mu_{PAR}, \sigma^2_{PAR})$ and compute $BW(t_{NI})$ according to (32).
  if $r_1 < HMCR$ **then**
    $\theta^{new} = \theta_i \pm r_3 \times BW$, where $i \in \{1, 2, \cdots, HMS\}$.
    if $r_2 < PAR$ **then**
      $\theta^{new} = \theta_{best}$, where θ_{best} is the best harmony in the HM as evaluated by $F(\theta)$.
    end if
  else
    $\theta^{new} = \theta_{min} + r_3 \times (\theta_{max} - \theta_{min})$.
  end if
  if $F(\theta^{new}) < F(\theta_{bad})$ **then**
    Substitute θ_{bad} in HM to θ^{new} and preserve the values of $HMCR$ and PAR, where θ_{bad} is the worst harmony in the HM as evaluated by $F(\theta)$.
  end if
  if $t_{UP} = UP$ **then**
    Update the μ_{HMCR} and μ_{PAR} by averaging the recorded values $HMCR$ and PAR.
    Reset $t_{UP} = 1$.
  else
    $t_{UP} = t_{UP} + 1$.
  end if
  $t_{NI} = t_{NI} + 1$.
end while

Output: the optimal local model accuracy θ^* in the HM as evaluated by $F(\theta)$.

By solving (31), we can get the optimal solution of problem (30). We develop a self-adaptive global best harmony search (SGHS) algorithm [50] to solve this continuous optimization problem. Then, we elaborate on the logic of the SGHS algorithm and determine its parameters.

We regard the unknown variable θ as a harmony played, and the optimized function $F(\theta)$ is the evaluation of this harmony. Based on the evaluation $F(\theta)$, the harmony θ be continuously adjusted, i.e., the search process, until the evaluation meets the requirements. The main parameters of the algorithm are shown below:

- Harmony Memory Size (HMS): It is a set of harmony, there are a total of HMS rows, and each row is a harmony which is a solution to the problem (31);
- Harmony Memory Consideration Rate (HMCR): It is the probability of choosing a harmony from the historic values stored in the HM;
- Pitch Adjustment Rate (PAR): It is the probability of fine-tuning the harmony chosen;
- distance BandWidth (BW): It is the amplitude of fine-tuning the harmony chosen;
- the Number of Improvisations (NI): It is the number of times that the harmony is played, and it is the same as the total number of function evaluations.

The control parameters mentioned above are closely related to the problem being solved and the exploration phase in the search process. The larger $HMCR$, the greater the probability that the newly played harmony be selected from HM, which increases the local search capability and convergence speed of the algorithm but reduces the diversity of the HM. A large BW value means that the algorithm performs a larger search scope. Therefore, as the number of iterations t_{NI} increases, the value of BW should become smaller. These selection of parameters is highly empirical, we set $HMCR$ and PAR to obey normal distribution, $\mathcal{N}(\mu_{HMCR}, \sigma^2_{HMCR})$ and $\mathcal{N}(\mu_{PAR}, \sigma^2_{PAR})$ respectively, and dynamically update on the basis of historic values. Let BW be updated with the following formula [50]:

$$BW(t_{NI}) = \begin{cases} BW_{max} - \frac{BW_{max} - BW_{min}}{NI} \times 2t_{NI}, & \\ & t_{NI} < \frac{NI}{2}, \\ BW_{min}, & \\ & t_{NI} \geq \frac{NI}{2}. \end{cases} \quad (32)$$

where BW_{min} and BW_{max} are the minimum and maximum distance bandwidths, respectively, which are the parameters we set in advance. In this way, the parameters do not need to be accurately set in advance, and the algorithm can automatically learn and dynamically iterate adaptively. Therefore, the SGHS algorithm has better performance on the problems with different characteristics and complexity.

The θ is a one-dimensional variable. From (C_6) in (31), we set $\theta_{min} = 0$ and $\theta_{max} = 1$. Let UP be the update period of $HMCR$ and PAR according to the recording parameters. The implementation of the algorithm details have been presented in Algorithm 2.

Combining the on-board CPU frequency $\boldsymbol{f}$, the transmission power $\boldsymbol{p}$ and the local model accuracy θ that have been obtained in Algorithm 1 and Algorithm 2, we designed Algorithm 3 to solve (17) under given $\mathcal{M}$ with $a_n = 1$. This applies the idea of block coordinate descent algorithm.

B. Vehicle Selection Algorithm

We then consider how to solve the problem of selecting vehicles participating in the FL workload on the road segment covered by an edge server. Under given $\mathcal{M}$, we can obtain $\boldsymbol{f}$, $\boldsymbol{p}$ and θ. It is hoped that selecting a group of vehicles can ensure that training task is carried out accurately and efficiently with minimum energy consumption and time delay. From the perspective of minimizing the objective function, not selecting vehicles to participate in the task is the most time-saving and energy-saving for each vehicle. Therefore, $\mathcal{M}$ cannot be obtained directly. We design a greedy algorithm shown in Algorithm 4 to iteratively add vehicles to $\mathcal{M}$ so that it maintains the overall performance.

Algorithm 3 A Joint Algorithm for Resource Allocation and Local Model Accuracy

Initialization:

- Initialize local model accuracy $\theta(0) \leftarrow [0, 1]$.
- Set the maximum number of iterations l_{max} and the specified precision ε.
- Let $l = 1$.

Allocate on-board CPU frequency $\boldsymbol{f}(0)$ and the transmission power $\boldsymbol{p}(0)$ by calling Algorithm 1 based on $\theta(0)$.
Substitute $\theta(0)$, $\boldsymbol{f}(0)$ and $\boldsymbol{p}(0)$ into (26) to obtain $\zeta(0)$ among $|\mathcal{M}|$ vehicles.

Iteration:

while $l \leq l_{max}$ **do**
  Search $\theta(l)$ by calling Algorithm 2 based on $\boldsymbol{f}(l-1)$ and $\boldsymbol{p}(l-1)$.
  Compute $\boldsymbol{f}(l)$ and $\boldsymbol{p}(l)$ by calling Algorithm 1 based on $\theta(l)$.
  Substitute $\theta(l)$, $\boldsymbol{f}(l)$ and $\boldsymbol{p}(l)$ into (26) to obtain $\zeta(l)$.
  if $|\zeta(l) - \zeta(l-1)| \leq \varepsilon$ **then**
    break.
  end if
  $l = l + 1$.
end while

Output: the optimal resource allocation $\hat{\boldsymbol{f}}^*$, $\hat{\boldsymbol{p}}^*$ and local model accuracy $\hat{\theta}^*$ in vehicle set $\mathcal{M}$.

Algorithm 4 Greedy vehicles selection algorithm

Initialization:

- Initialize $\mathcal{M} \leftarrow \emptyset$.

Iteration:

while $\mathcal{V} \neq \emptyset$ **do**
  Solve $\Psi_n = \min\limits_{\theta, \boldsymbol{f}, \boldsymbol{p}} \max\limits_{m} \mathcal{S}_m$ with $V_m \in \{\mathcal{M} \cup V_n\}$ by calling Algorithm 3 for each vehicle V_n which meet the condition $L_n \leq L_{max}$ in the vehicle set $\mathcal{V}$.
  Set $\Psi_{\hat{n}}^{min} = \min\{\Psi_n\}$.
  Update $\mathcal{M} \leftarrow \mathcal{M} \cup V_{\hat{n}}$ and $\mathcal{V} \leftarrow \mathcal{V} \setminus V_{\hat{n}}$.
end while

Output: the set $\mathcal{M}^*$ of the selected vehicles.

Algorithm 4 is a dynamic addition process. When a new vehicle enters the current edge server's coverage area, it will turn to the vehicle set $\mathcal{V}$ automatically that can be selected. Combining with the vehicle set $\mathcal{M}$ currently participating in the training, the vehicle that minimizes the energy and delay cost of the overall task is added among the available vehicles each time.

VI. Simulation Results and Analysis

In this section, we present the convergence and performance of the proposed algorithms. First of all, we introduce the parameter settings of the simulation. Then, the simulation results be used to verify the convergence of Algorithm 1, Algorithm 2 and Algorithm 3, and we confirm the effectiveness of Algorithm 4. Finally, a series of comparative experiments are done to demonstrate the performance of the proposed scheme.

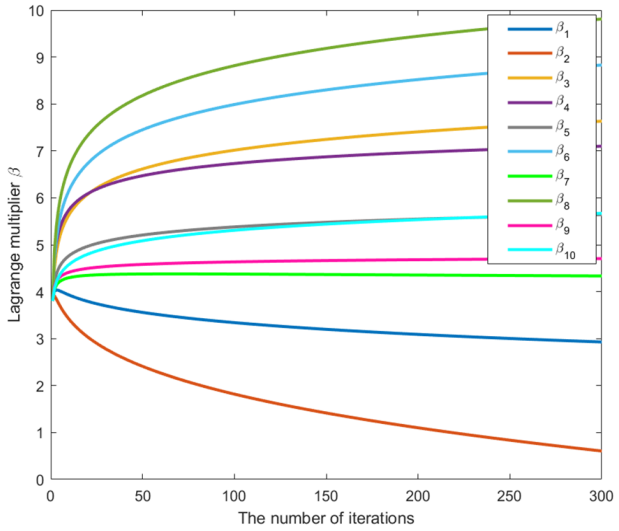

Fig. 2: Convergence of Algorithm 1.

TABLE II: Parameter Settings in the Simulation

Parameter	Meaning	Value
v^{min}/v^{max}	Minimum / Maximum Speed	[2, 24] m/s
μ	Mean Speed	13 m/s
σ	Standard Deviation of Speed	5
$\mathcal{L}$	Coverage Diameter of Edge Server	10 m
l_n	The Position of Vehicle V_n	[1, 10] m
T	Exposure Time Interval	$\frac{1}{200}s$
f	Camera Focal Length	10 mm
d	Perpendicular Distance	5 m
Q	CCD Pixel Size	0.011 mm
D_n	Training Data Samples	[60, 150] KB
q_n	Processing Density	[900,1600] cycles/bit
k_n	Capacitance Coefficient	10^{-27}
α_n	State of Vehicular connection	[3, 5]$\times 10^6$
HMS	Harmony Memory Size	5
NI	Number of Improvisations	6000
UP	The Period of Parameter Update	100
BW_{min}	Minimum Distance Bandwidth	0.0005
BW_{max}	Maximum Distance Bandwidth	0.05
μ_{HMCR}/μ_{PAR}	Mean HMCR/PAR	0.95 / 0.3
$\sigma_{HMCR}/\sigma_{PAR}$	Standard Deviation of HMCR/PAR	0.01 / 0.05
L_{max}	Maximum Acceptable Image Blur	16

A. Simulation Settings

We simulated the experiments based on the Monte Carlo simulation in a Matlab-based laptop with 16GB memory. The CPU is Intel Core i7-10510U with 4 cores. We assumed that there are $N = 10$ vehicles traveling on a road segment covered by an edge server in the simulation. The lower and upper-speed constraints of the vehicles on different traffic lanes are distinct in urban streets, suburban roads, and highways [51]. In order to be suitable for a more comprehensive scene application, we set the vehicle speed range to be larger so that it can get the general speed of the ordinary road. The parameters used in the simulation are listed in Table II.

In the case of optimizing resource allocation and local model accuracy, the experiments are conducted to verify the effectiveness of Algorithm 4 to select vehicle set $\mathcal{M}$. We set up

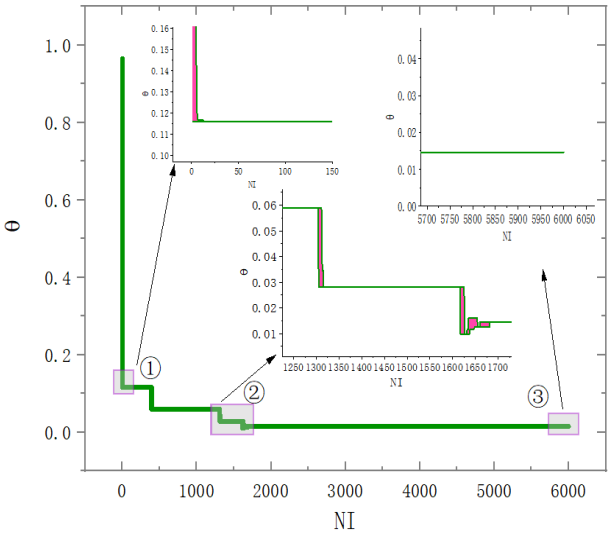

Fig. 3: Convergence of Algorithm 2.

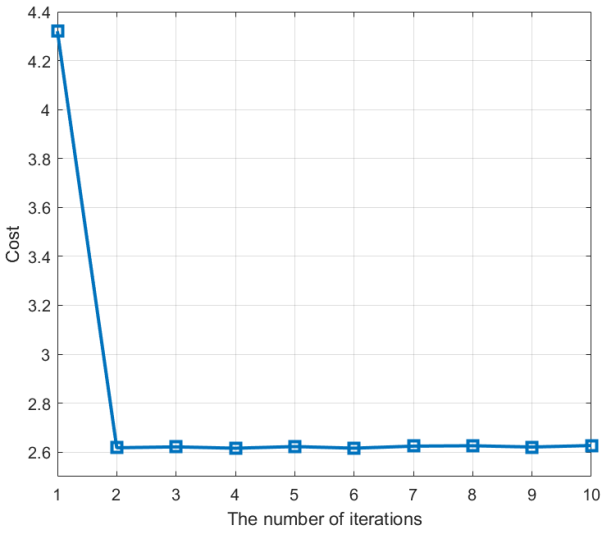

Fig. 4: Convergence of Algorithm 3.

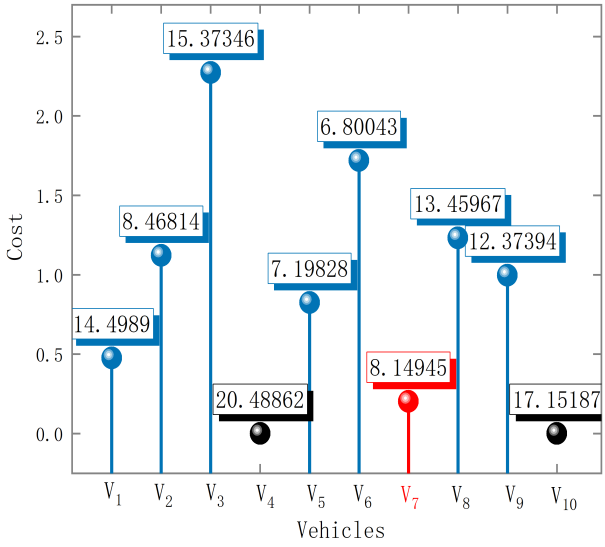

Fig. 5: Choose a vehicle V_7 in Algorithm 4.

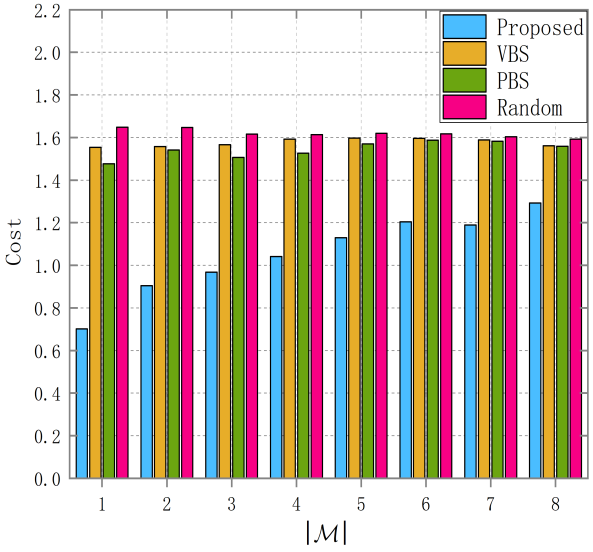

Fig. 6: The effectiveness of Algorithm 4.

the velocity-based selection scheme (VBS) to add the vehicles with the smallest driving velocity, the position-based selection scheme (PBS) to add the vehicles with the least distance from the coverage area's entrance of the edge server, and the random selection scheme for comparison. We optimized some variables separately and randomly selected the remaining variables to highlight the proposed scheme's advantages. The experiments set the following schemes:

- the accuracy-optimized scheme (AOS) to optimize the local model accuracy and randomly select the on-board CPU frequency and transmission power.
- the resource-optimized scheme (ROS) to optimize the on-board CPU frequency, transmission power and randomly select the local model accuracy.
- the non-optimized scheme (NOS) to randomly select the local model accuracy, the on-board CPU frequency, and transmission power.

B. *Convergence of the Proposed Algorithm*

Fig. 2 plots the Lagrange multiplier β versus the number of iterations to show the convergence of the Algorithm 1. We output the iterative curve of all N vehicles, which shows the effectiveness of the Algorithm 1. It can be observed that the subgradient projection method of Lagrange multipliers is gradually approaching the optimal values, and a convergent solution is obtained. Fig. 3 shows the value range and convergence of the harmony memory with the iterative process of $NI = 6000$. We used two green lines to draw the maximum and minimum values of the five θ, and filled the middle gap with red to indicate the value area of θ in HM. We randomly

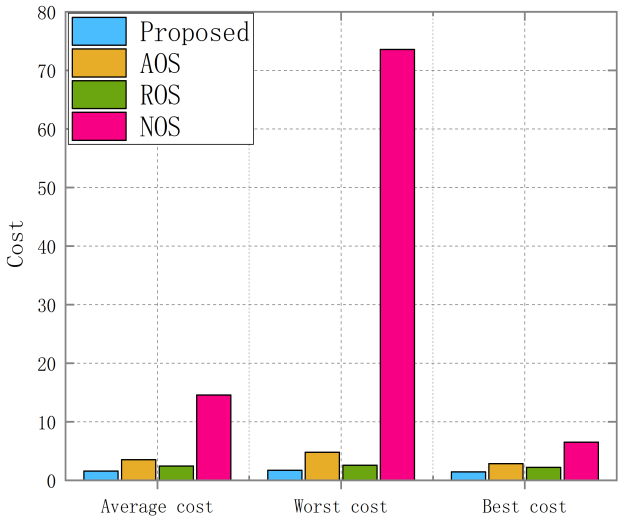

Fig. 7: Comparisons of the average cost, the worst cost, and the best cost.

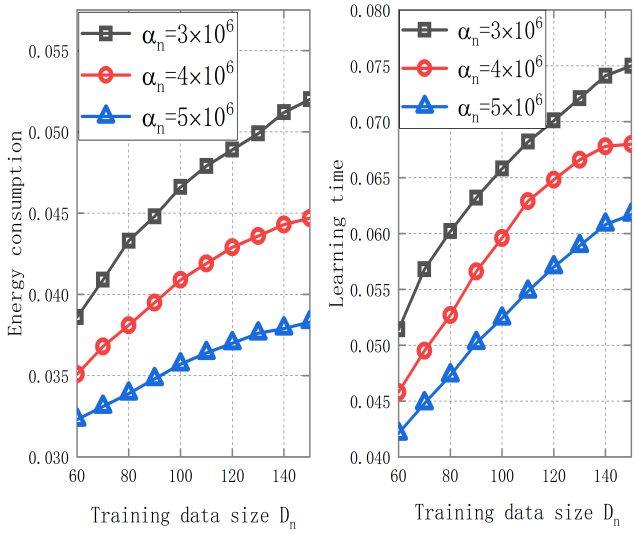

Fig. 8: Energy consumption / training time under different training data size D_n.

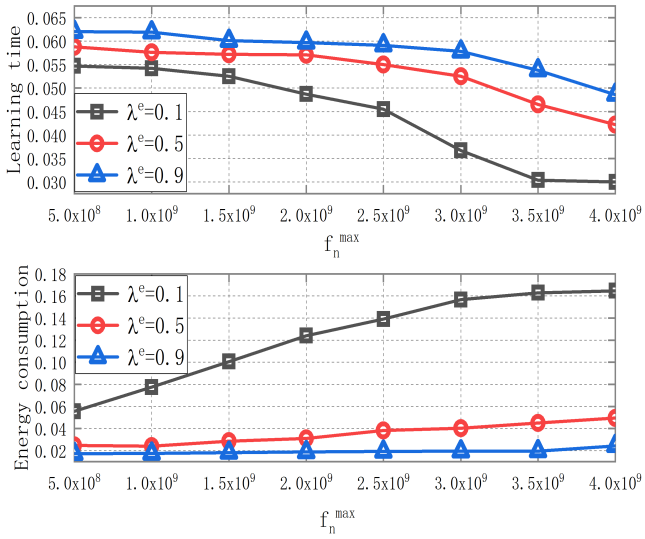

Fig. 9: Training time / energy consumption under different maximum on-board CPU frequency f_n^{max}.

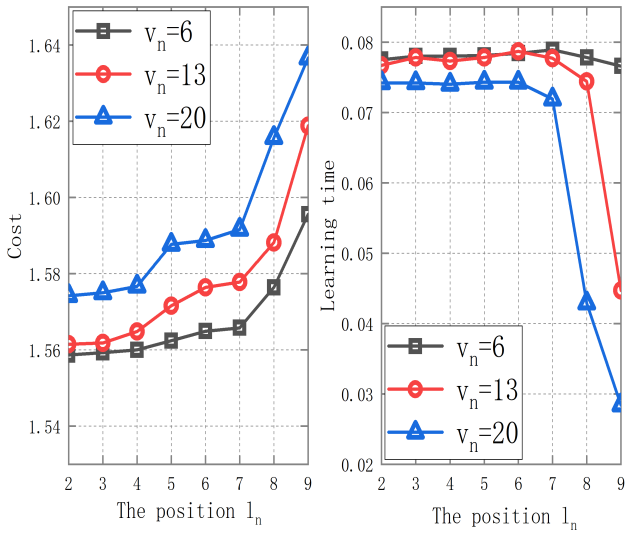

Fig. 10: Cost / learning time under different position l_n.

initialized θ in [0,1] and enlarged the position ①. At the beginning of the iterations, it can be seen that there are long-span values in HM, and it converge quickly after about 20 iterations. As the iteration progressing, inflection points and gaps between two lines like the position ② appears, which means that HM is updating and adding the new θ in search. We enlarged the position ③ at the end of the iterations, showing that the five θ in the HM converge to $\theta^* = 0.0146$. Fig. 4 plots the cost versus the number of iterations to show the convergence of the Algorithm 3. And it represents that the mutual iterations between Algorithm 1 and Algorithm 2. It can quickly achieve good convergence results, which is based on the benign convergence of the inner loop shown in Fig. 2 and Fig. 3 certainly.

In a greedy search, adding the vehicle $V_{\hat{n}}$ that makes the min-max cost in set $\mathcal{M}$ is minimum. Moreover, the vehicle meets the requirements of the FL tasks for image quality. Fig. 5 verifies the effectiveness of this process in Algorithm 4. The blur degree of images in V_4 and V_{10} exceed the threshold L_{max}, so they are not involved in the resource scheduling optimization of training tasks. The degree of blur is marked in the box next to each point in Fig. 5. According to each vehicle's cost shown in the figure, the algorithm selects the least min-max cost V_7 into the set $\mathcal{M}$.

C. Performance of the Proposed Algorithm

In Fig. 6, 1 to 8 vehicles are chosen from the N vehicles according to the proposed scheme, the VBS, the PBS, and the random scheme, respectively, and they are added to the vehicle set $\mathcal{M}$ one by one. We compared the cost of the four schemes. The proposed scheme added the vehicle that keeps the overall cost of $\mathcal{M}$ to a minimum each time. As the number of vehicles in $\mathcal{M}$ increases, the overall system cost rises gradually. The VBS and the PBS are better than the random selection scheme slightly. Since each scheme optimizes resources and local model accuracy, the more vehicles added, the smaller the cost difference. However, Algorithm 4 is a dynamic process. After a new vehicle enters the edge server's coverage area, the vehicle that minimizes the overall cost is always added at first. This experiment shows the superiority of the proposed vehicle selection algorithm compared with other alternatives under optimizing resource allocation and local model accuracy.

In Fig. 7, we compare the average cost, the worst cost, and the best cost among the proposed scheme, the AOS, the ROS, and the NOS. The cost of the AOS exceeds the ROS, so the optimization of the resource allocation is more helpful in reducing system cost than local model accuracy. The NOS will cause colossal energy waste and time delay and make a vast difference in the cost of vehicles participating in training tasks, causing unfairness among them. We found that the proposed optimization scheme not only minimizes the cost but also has a small cost difference between each vehicle, which is well-balanced and fair.

Fig. 8 shows the energy consumption and training time under different training data size D_n. For a given α_n, the larger the training data size, the longer the learning time and the higher the energy consumption. For a given D_n, the better the state of the vehicular connection α_n, the shorter the learning time and the lower the energy consumption. Due to the proposed scheme can achieve a tradeoff between energy consumption and time delay, the growth of energy consumption and learning time become slow gradually as D_n increases. The α_n can also affect the iterative process by influencing the maximum transmission power indirectly.

In Fig. 9, we investigate the training time and energy consumption under different maximum on-board CPU frequency f_n^{max}, and we can observe that the learning time is decreasing and energy consumption increasing as the f_n^{max} of vehicles increase. For a given f_n^{max}, the larger the λ^e, the longer the learning time and the lower the consumption. This is because when the weight of the energy consumption in the $\mathcal{S}_n = \mathcal{I}(\theta)(\lambda^t T_n + \lambda^e E_n)$ becomes larger, the learning time will become longer, and the energy consumption will be reduced to balance the value of the total cost.

In Fig. 10, we present the cost and learning time under the different positions of vehicles. The farther the vehicle is from the edge server coverage area entrance, it means that the vehicle is about to leave this communication area. This causes the residence time τ_n to be small. So the learning time must be sharply reduced to complete the learning task within the requirement, and the cost will increase accordingly. For a given position l_n, if the faster the velocity v_n of vehicles, it means that the shorter the residence time, the learning time will be reduced, and the cost will also increase.

Fig. 8, Fig. 9 and Fig. 10 show the impact of vehicles with different training data sizes D_n, connection status α_n, maximum on-board CPU frequency f_n^{max}, optimized parameters λ^e, speed v_n and position l_n on FL training time and energy consumption. These experiments further highlight the performance optimization of the proposed scheme among various vehicle states and achieve the preset goals.

VII. Conclusions

In this paper, we have considered the problems and characteristics of FL deployed in VEC. Mainly include image quality, velocity and position of the vehicles, learning time, energy consumption, and vehicle selection. We formulated a joint optimization with on-board computation capability, transmission power, and local model accuracy to achieve the minimum cost in the worst case of FL. Due to the min-max optimization problem formed as a nonlinear programming problem, we decomposed it into two subproblems. For the resource allocation problem, we used the Lagrangian dual problem and the sub-gradient projection method to approximate the optimal value iteratively. For the local model accuracy problem, we developed an adaptive harmony algorithm for heuristic search. Based on the on-board image quality and joint optimization process, we developed a greedy vehicle selection algorithm to minimize the overall system cost. Finally, the simulation results have shown that our proposed algorithms have well convergence and effectiveness. It is utility to achieve fairness to the cost and resource optimization among vehicles, and it provides a tradeoff between learning time and energy consumption.

Appendix A
Proof of the Theorem 1

From the definition of $G(\boldsymbol{\lambda}, \boldsymbol{\beta}, \boldsymbol{\mu}, \boldsymbol{\varphi})$ in (20), we can obtain the optimal solution of on-board CPU frequency f_n^* and transmission power p_n^* under given Lagrange multipliers λ_n, β_n, μ_n and φ_n. So the following inequality can be established:

$$
\begin{aligned}
&G(\boldsymbol{\lambda}^{'}, \boldsymbol{\beta}^{'}, \boldsymbol{\mu}^{'}, \boldsymbol{\varphi}^{'}) \\
&\quad \leq \zeta + \sum_{V_n \in \mathcal{M}} \lambda_n^{'} \left[\log(\frac{1}{\theta}) \frac{|D_n| q_n}{f_n^*} + \frac{C_n}{\alpha_n \sqrt{p_n^*}} - \tau_n \right] \\
&\quad + \sum_{V_n \in \mathcal{M}} \beta_n^{'} \left[p_n^* - \left(\frac{E_{max}^{tran} \alpha_n}{C_n} \right)^2 \right] + \sum_{V_n \in \mathcal{M}} \mu_n^{'} (f_n^* - f_n^{max}) \\
&\quad + \sum_{V_n \in \mathcal{M}} \varphi_n^{'} \left[\mathcal{I}(\theta) \left(\lambda^t \log(\frac{1}{\theta}) \frac{|D_n| q_n}{f_n^*} + \lambda^t \frac{C_n}{\alpha_n \sqrt{p_n^*}} \right.\right. \\
&\quad \left.\left. + \lambda^e \log(\frac{1}{\theta}) k_n |D_n| q_n (f_n^*)^2 + \lambda^e \frac{C_n \sqrt{p_n^*}}{\alpha_n} \right) - \zeta \right]
\end{aligned}
$$

Adding $G(\boldsymbol{\lambda}, \boldsymbol{\beta}, \boldsymbol{\mu}, \boldsymbol{\varphi})$ to the right of the inequality and transpose the remainder to the left side. Then, we can get

following inequality after rearranging them:

$$
\begin{aligned}
&G(\boldsymbol{\lambda}^{'},\boldsymbol{\beta}^{'},\boldsymbol{\mu}^{'},\boldsymbol{\varphi}^{'}) \\
&\quad+\sum_{V_n\in\mathcal{M}}(\lambda_n-\lambda_n^{'})\left[\log(\frac{1}{\theta})\frac{|D_n|q_n}{f_n^*}+\frac{C_n}{\alpha_n\sqrt{p_n^*}}-\tau_n\right] \\
&\quad+\sum_{V_n\in\mathcal{M}}(\beta_n-\beta_n^{'})\left[p_n^*-\left(\frac{E_{max}^{tran}\alpha_n}{C_n}\right)^2\right] \\
&\quad+\sum_{V_n\in\mathcal{M}}(\mu_n-\mu_n^{'})(f_n^*-f_n^{max}) \\
&\quad+\sum_{V_n\in\mathcal{M}}(\varphi_n-\varphi_n^{'})\left[\mathcal{I}(\theta)\left(\lambda^t\log(\frac{1}{\theta})\frac{|D_n|q_n}{f_n^*}+\lambda^t\frac{C_n}{\alpha_n\sqrt{p_n^*}}\right.\right. \\
&\qquad\left.\left.+\lambda^e\log(\frac{1}{\theta})k_n|D_n|q_n(f_n^*)^2+\lambda^e\frac{C_n\sqrt{p_n^*}}{\alpha_n}\right)-\zeta\right] \\
&\qquad\leq G(\boldsymbol{\lambda},\boldsymbol{\beta},\boldsymbol{\mu},\boldsymbol{\varphi})
\end{aligned}
$$

The subgradient z of a convex function $g(\cdot)$ at the point x_0 is defined as: $g(x_0)+z^T(x-x_0)\leq g(x)$, it holds for all x in the domain. Thus, the *Theorem 1* holds.

REFERENCES

[1] C. Chen, A. Xl, C. Tq, and D. Aks, "A short-term traffic prediction model in the vehicular cyberphysical systems," *Future Generation Computer Systems*, vol. 105, pp. 894–903, 2020.

[2] C. Chen, H. Jinna, T. Qiu, M. Atiquzzaman, and Z. Ren, "Cvcg: Cooperative v2v-aided transmission scheme based on coalitional game for popular content distribution in vehicular ad-hoc networks," *IEEE Transactions on Mobile Computing*, pp. 1–1, 2018.

[3] "Global autonomous driving market outlook, 2018." [Online]. Available: https://www.prnewswire.com/news-releases/globalautonomous-driving-market-outlook-2018-300624588.html

[4] Z. Dong, W. Shi, G. Tong, and K. Yang, "Collaborative autonomous driving: Vision and challenges," in *MetroCAD 2020*, 2020.

[5] 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 of Things Journal*, vol. 7, no. 7, pp. 6214–6228, 2020.

[6] L. Liu, J. Feng, Q. Pei, C. Chen, Y. Ming, B. Shang, and M. Dong, "Blockchain-enabled secure data sharing scheme in mobile-edge computing: An asynchronous advantage actorcritic learning approach," *IEEE Internet of Things Journal*, vol. 8, no. 4, pp. 2342–2353, 2021.

[7] J. Du, F. R. Yu, G. Lu, J. Wang, J. Jiang, and X. Chu, "Mec-assisted immersive vr video streaming over terahertz wireless networks: A deep reinforcement learning approach," *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 9517–9529, 2020.

[8] Z. Yang, M. Chen, K. K. Wong, H. V. Poor, and S. Cui, "Federated learning for 6g: Applications, challenges, and opportunities," 2021.

[9] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Transactions on Intelligent Systems and Technology*, vol. 10, no. 2, pp. 1–19, 2019.

[10] S. Liu, L. Liu, J. Tang, B. Yu, and W. Shi, "Edge computing for autonomous driving: Opportunities and challenges," *Proceedings of the IEEE*, vol. PP, no. 99, pp. 1–20, 2019.

[11] A. Ndikumana, N. H. Tran, T. M. Ho, Z. Han, W. Saad, D. Niyato, and C. S. Hong, "Joint communication, computation, caching, and control in big data multi-access edge computing," *IEEE Transactions on Mobile Computing*, vol. 19, no. 6, pp. 1359–1374, 2020.

[12] L. Liu, S. Lu, R. Zhong, B. Wu, Y. Yao, Q. Zhang, and W. Shi, "Computing systems for autonomous driving: State-of-the-art and challenges," *arXiv preprint arXiv:2009.14349*, 2020.

[13] L. U. Khan, S. R. Pandey, N. H. Tran, W. Saad, Z. Han, M. N. H. Nguyen, and C. S. Hong, "Federated learning for edge networks: Resource optimization and incentive mechanism," 2019.

[14] Haigen, Min, Xia, Chaoyi, Cheng, Xiangmo, and Zhao, "Kinematic and dynamic vehicle model-assisted global positioning method for autonomous vehicles with low-cost gps/camera/in-vehicle sensors." *Sensors (Basel, Switzerland)*, vol. 19, no. 24, 2019.

[15] S. Baidya, Y.-J. Ku, H. Zhao, J. Zhao, and S. Dey, "Vehicular and edge computing for emerging connected and autonomous vehicle applications," in *2020 57th ACM/IEEE Design Automation Conference (DAC)*, 2020, pp. 1–6.

[16] S. M. A. Kazmi, T. N. Dang, I. Yaqoob, A. Ndikumana, E. Ahmed, R. Hussain, and C. S. Hong, "Infotainment enabled smart cars: A joint communication, caching, and computation approach," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 9, pp. 8408–8420, 2019.

[17] C. Zhu, J. Tao, G. Pastor, Y. Xiao, Y. Ji, Q. Zhou, Y. Li, and A. Yl-Jski, "Folo: Latency and quality optimized task allocation in vehicular fog computing," *IEEE Internet of Things Journal*, 2018.

[18] L. Pu, X. Chen, G. Mao, Q. Xie, and J. Xu, "Chimera: An energy-efficient and deadline-aware hybrid edge computing framework for vehicular crowdsensing applications," *IEEE Internet of Things Journal*, vol. PP, pp. 1–1, 2018.

[19] P. Keshavamurthy, E. Pateromichelakis, D. Dahlhaus, and C. Zhou, "Edge cloud-enabled radio resource management for co-operative automated driving," *IEEE Journal on Selected Areas in Communications*, vol. PP, no. 99, pp. 1–1, 2020.

[20] M. Shojafar, N. Cordeschi, D. Amendola, and E. Baccarelli, "Energy-saving adaptive computing and traffic engineering for real-time-service data centers," in *2015 IEEE International Conference on Communication Workshop (ICCW)*, 2015, pp. 1800–1806.

[21] Y. Wang, P. Lang, D. Tian, J. Zhou, and D. Zhao, "A game-based computation offloading method in vehicular multiaccess edge computing networks," *IEEE Internet of Things Journal*, vol. PP, no. 99, pp. 1–1, 2020.

[22] K. Tan, D. Bremner, J. L. Kernec, and M. Imran, "Federated machine learning in vehicular networks: A summary of recent applications," in *2020 International Conference on UK-China Emerging Technologies (UCET)*, 2020, pp. 1–4.

[23] S. Samarakoon, M. Bennis, W. Saad, and M. Debbah, "Distributed federated learning for ultra-reliable low-latency vehicular communications," *IEEE Transactions on Communications*, vol. PP, no. 99, pp. 1–1, 2019.

[24] M. Chen, H. V. Poor, W. Saad, and S. Cui, "Convergence time optimization for federated learning over wireless networks," *IEEE Transactions on Wireless Communications*, vol. 20, no. 4, pp. 2457–2471, 2021.

[25] M. Chen, Z. Yang, W. Saad, C. Yin, and S. Cui, "Performance optimization of federated learning over wireless networks," in *GLOBECOM 2019 - 2019 IEEE Global Communications Conference*, 2019.

[26] M. Chen, Z. Yang, W. Saad, C. Yin, H. V. Poor, and S. Cui, "A joint learning and communications framework for federated learning over wireless networks," *IEEE Transactions on Wireless Communications*, vol. 20, no. 1, pp. 269–283, 2021.

[27] J. Xu and H. Wang, "Client selection and bandwidth allocation in wireless federated learning networks: A long-term perspective," 2020.

[28] W. Shi, S. Zhou, Z. Niu, M. Jiang, and L. Geng, "Joint device scheduling and resource allocation for latency constrained wireless federated learning," 2020.

[29] S. Luo, X. Chen, Q. Wu, Z. Zhou, and S. Yu, "Hfel: Joint edge association and resource allocation for cost-efficient hierarchical federated edge learning," *IEEE Transactions on Wireless Communications*, vol. PP, no. 99, pp. 1–1, 2020.

[30] S. Yousefi, E. Altman, R. El-Azouzi, and M. Fathy, "Analytical model for connectivity in vehicular ad hoc networks," *IEEE Transactions on Vehicular Technology*, vol. 57, no. 6, pp. 3341–3356, 2008.

[31] S. M. Abuelenin and A. Y. Abul-Magd, "Empirical study of traffic velocity distribution and its effect on vanets connectivity," in *2014 International Conference on Connected Vehicles and Expo (ICCVE)*. IEEE, 2014, pp. 391–395.

[32] Y. AlNagar, S. Hosny, and A. A. El-Sherif, "Towards mobility-aware proactive caching for vehicular ad hoc networks," in *2019 IEEE Wireless Communications and Networking Conference Workshop (WCNCW)*. IEEE, 2019, pp. 1–6.

[33] Z. Yu, J. Hu, G. Min, Z. Zhao, W. Miao, and M. S. Hossain, "Mobility-aware proactive edge caching for connected vehicles using federated learning," *IEEE Transactions on Intelligent Transportation Systems*, 2020.

[34] S. Dodge and L. Karam, "Understanding how image quality affects deep neural networks," in *2016 Eighth International Conference on Quality of Multimedia Experience (QoMEX)*, 2016, pp. 1–6.

[35] J. A. Corts-Osorio, J. B. Gmez-Mendoza, and J. C. Riao-Rojas, "Velocity estimation from a single linear motion blurred image using discrete cosine transform," *IEEE Transactions on Instrumentation and Measurement*, vol. 68, no. 10, pp. 4038–4050, 2019.

[36] D. Ye, R. Yu, M. Pan, and Z. Han, "Federated learning in vehicular edge computing: A selective model aggregation approach," *IEEE Access*,

vol. 8, pp. 23 920–23 935, 2020.
[37] J. Konecny, P. Richtarik, and Z. Qu, "Semi-stochastic coordinate descent," *Mathematics*, 2014.
[38] 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 *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*, 2019, pp. 1387–1395.
[39] C. Ma, J. Konečnỳ, M. Jaggi, V. Smith, M. I. Jordan, P. Richtárik, and M. Takáč, "Distributed optimization with arbitrary local solvers," *Optimization Methods and Software*, vol. 32, no. 4, pp. 813–848, 2017.
[40] T. D. Burd and R. W. Brodersen, "Processor design for portable systems," *Journal of VLSI signal processing systems for signal, image and video technology*, vol. 13, no. 2-3, pp. 203–221, 1996.
[41] Y. Wang, M. Sheng, X. Wang, L. Wang, and J. Li, "Mobile-edge computing: Partial computation offloading using dynamic voltage scaling," *IEEE Transactions on Communications*, vol. 64, no. 10, pp. 4268–4282, 2016.
[42] J. Feng, F. R. Yu, Q. Pei, J. Du, and L. Zhu, "Joint optimization of radio and computational resources allocation in blockchain-enabled mobile edge computing systems," *IEEE Transactions on Wireless Communications*, 2020.
[43] W. Zhang, Y. Wen, K. Guan, D. Kilper, H. Luo, and D. O. Wu, "Energy-optimal mobile cloud computing under stochastic wireless channel," *IEEE Transactions on Wireless Communications*, vol. 12, no. 9, pp. 4569–4581, 2013.
[44] M. Gudmundson, "Correlation model for shadow fading in mobile radio systems," *Electronics Letters*, vol. 27, no. 23, pp. 2145–2146, 1991.
[45] M. Shojafar, N. Cordeschi, and E. Baccarelli, "Energy-efficient adaptive resource management for real-time vehicular cloud services," *IEEE Transactions on Cloud Computing*, vol. 7, no. 1, pp. 196–209, 2019.
[46] J. Liu, K. Xiong, D. W. K. Ng, P. Fan, Z. Zhong, and K. B. Letaief, "Max-min energy balance in wireless-powered hierarchical fog-cloud computing networks," *IEEE Transactions on Wireless Communications*, pp. 1–1, 2020.
[47] S. Boyd, "Subgradient methods." [Online]. Available: http://stanford.edu/class/ee364b/lectures/subgrad_method_notes.pdf/
[48] L. V. Stephen Boyd, "Covex optimization." [Online]. Available: https://web.stanford.edu/~boyd/cvxbook/bv_cvxbook.pdf
[49] A. Smith, D. Coit, T. Bck, D. Fogel, and Z. Michalewicz, "Penalty functions," 07 1998.
[50] Q. K. Pan, P. N. Suganthan, M. F. Tasgetiren, and J. J. Liang, "A self-adaptive global best harmony search algorithm for continuous optimization problems," *Applied Mathematics and Computation*, vol. 216, no. 3, pp. 830–848, 2010.
[51] A. F. Molisch, F. Tufvesson, J. Karedal, and C. F. Mecklenbrauker, "A survey on vehicle-to-vehicle propagation channels," *IEEE Wireless Communications*, vol. 16, no. 6, pp. 12–22, 2009.