2025 IEEE International Conference on Edge Computing and Communications (EDGE) | 979-8-3315-5559-7/25/$31.00 ©2025 IEEE | DOI: 10.1109/EDGE67623.2025.00024

Energy Efficient Client Selection in Federated Learning for Orbital Edge Computing

Bara'ah Al-Blewi
Western Sydney University
Sydney, Australia
b.al-blewi@westernsydney.edu.au

Bahman Javadi
Western Sydney University
Sydney, Australia
b.javadi@westernsydney.edu.au

Rodrigo N. Calheiros
Western Sydney University
Sydney, Australia
r.calheiros@westernsydney.edu.au

***Abstract*—Low Earth Orbit (LEO) satellite constellations provide a wide range of services such as communications, earth observation, signal monitoring, and scientific missions. While these constellations generate valuable data, transferring it to ground stations (GS) for machine learning-based analysis presents significant challenges due to downlink bandwidth and energy constraints. Federated Learning (FL) integrated with Orbital Edge Computing (OEC) has been explored as a solution to these challenges. This paper presents FedSCS (Satellite Client Selection), a novel energy-efficient and decentralised FL framework designed to optimise communication with GSs and minimise energy consumption. FedSCS selects satellites (clients) based on their available resources and utilises reinforcement learning for cluster formation. The performance evaluation conducted under the Walker Delta-based LEO constellation across various datasets reveals that FedSCS can sustain high accuracy while considerably reducing training time and energy consumption. FedSCS achieves a notable reduction in energy consumption of 6.67%, 10.34%, and 9.09% compared to the recently developed FedOrbit on the MNIST, CIFAR-10, and EuroSat datasets, while also achieving a slight improvement in accuracy.**

***Index Terms*—Client Selection, Federated Learning, Orbital Edge Computing, Energy Consumption.**

I. Introduction

Satellites in Low Earth Orbit (LEO) have gained significant attention in recent years due to technological improvements and their applications in various fields [24]. Since LEO satellites produce a large amount of data, artificial intelligence (AI) and machine learning (ML) have been incorporated into the Ground Station (GS) to analyse the data produced. Orbital Edge Computing (OEC) leverages satellite-based systems to process data closer to its source, reducing latency and enhancing the efficiency of global communications [5].

Federated Learning (FL) [22] is applied in OEC for decentralised machine learning among satellites (clients). It allows satellites to train ML models onboard without transmitting raw data, reducing communication overhead. The process depends on a global server, which in this scenario is the GS, to aggregate model updates from participating satellites. A challenge in this context is the high energy consumption and inefficiency of FL training in space-based systems. Running complex ML models onboard LEO satellites is computationally intensive and energy-demanding, especially with limited power availability from solar panels. Additionally, traditional FL relies on GS-based model aggregation, which leads to long training times and high power consumption [3], [14].

The advances of Inter-Satellite Links (ISLs) [2] offer a promising approach to collaboration among satellites by enabling direct satellite-to-satellite communication [9] [33], potentially reducing dependence on GSs for model aggregation. However, ISL-based FL introduces new challenges, including extended training durations, energy constraints, and accuracy degradation. Previous efforts to implement decentralised FL aggregation via ISLs have encountered problems such as model aging and reduced accuracy, primarily due to high data similarity within the same orbital regions [8]. Additionally, client selection strategy is another critical factor affecting FL performance [11], as LEO satellites are heterogeneous, differing in battery capacity, computational capabilities, memory, and communication availability [19].

In response to these challenges, this paper introduces a new FL approach (FedSCS) for OEC. It optimises client selection, ensuring that satellites are chosen based on computational resources, battery capacity, and communication availability, thereby reducing inefficient participation and accelerating model convergence. The proposed approach enhances energy efficiency while maintaining model accuracy. The contributions of this paper are as follows:

- It proposes FedSCS, a decentralised federated learning approach that selects clients based on their resources and predicts the future available resources for the selection.
- It proposes a new strategy based on a resource prediction model and communication strategy to cluster satellites to improve the energy consumption and training performance.
- It conducts comprehensive experiments across diverse datasets to demonstrate the performance of FedSCS with remarkable improvements in training time, energy consumption and accuracy.

The rest of this paper is organised as follows. Related work is presented in Section II. The system model and problem formulation are presented in Section III. In Section IV, we present the proposed FedSCS algorithm. Performance evaluation and experimental results are presented in Section V. Conclusion and future work are discussed in Section VI.

2767-9918/25/$31.00 ©2025 IEEE
DOI 10.1109/EDGE67623.2025.00024

II. Related Work

This section provides recent advancements in the application of FL and client selection in OEC.

A. Federated Learning in Orbital Edge Computing

Early FL approaches in LEO Satellite Networks [4] applied FedAvg to the constellations to validate the effectiveness of FL in satellite networks. However, the synchronous nature of FedAvg requires a constant connection to GS. leading to long convergence times, potentially taking several days to complete a model update. FedSat [25] is an asynchronous FL approach where the global server does not need to wait for all client updates for aggregation. Although this accelerates convergence, it leads to model aging, since satellites with low computational power or poor connectivity fall significantly behind in the global rounds.

FedSpace [30] is an asynchronous Federated Learning algorithm that buffers client models and prioritises newer updates for aggregation. It adjusts global model aggregation schedules based on satellite orbits and Earth's rotation, ensuring satellites with better connectivity and fresher models contribute effectively. FedISL [28] [26] utilises inter-satellite links (ISLs) to facilitate direct satellite-to-satellite communication, reducing reliance on GS. Building on this, FedSatSchedule [27] was proposed as a scheduling algorithm that determines whether a satellite can complete local training during the communication window. This scheduling mechanism is essential due to the brief visibility period of satellites with GS. FedGSM [34] was introduced as an advanced asynchronous FL algorithm that incorporates a gradient staleness compensation mechanism to mitigate the impact of outdated model updates.

B. Client Selection in Orbital Edge Computing

Client selection plays a crucial role in FL, which identifies which clients participate in each training round to improve training efficiency and reduce resource consumption. There is limited work related to the client selection for FL in OEC. FedLEO [9] is a synchronous FL algorithm that selects an optimal sink satellite in each orbit to generate a partial global model. It optimises communication between sink satellites and the GS by using predictable satellite orbits. FELLO [3] is an FL algorithm that aims to minimise reliance on GSs by enabling satellites to perform onboard training and decentralised model aggregation while maintaining efficient communication. FedOrbit [15] proposed a novel decentralised FL approach in a cluster formation based on visiting patterns, ISLs plane and reinforcement learning. One satellite is chosen as the master to aggregate local models before transmitting updates to the next level. It utilises Block Minifloat (BM) to accelerate FL training on custom hardware (i.e., FPGA). Wu et al. [32] introduced a client affinity-based approach, selecting satellites based on measuring the contribution of the client to the global model. While many existing works have been proposed to tackle FL in OEC, there is a significant gap in the literature when addressing client selection in OEC.

III. System Model and Problem Formulation

This section includes the system architecture for satellite constellations equipped with ISL. The satellites are heterogeneous and have different battery, CPU, and memory capacities. This paper considers a satellite constellation including a distinct number of orbit planes (OPs) and satellites. A unique ID is assigned to each satellite according to its OP number. An orbit set is defined as $O = \{o_i\}$ where $i = \{1, 2, ..., n\}$. In a similar manner, the satellite set is denoted as $S = \{s_{i,j}\}$ where $j = \{1, 2, \ldots, l\}$. Thus, each orbit o_i contains l equally spaced satellites with unique IDs $\{S_{i,1}, S_{i,2}, \ldots, S_{i,l}\}$.

Satellites form clusters with neighbouring nodes based on inter-satellite link range, terminals, and limitations. The clustering process, which takes into account both intra-plane and inter-plane inter-satellite links, together with the selection of master satellites for ground station communication, is illustrated in Figure 1. The satellites shown in yellow are the master nodes that maintain connection with the ground station, while the rest must await the subsequent visitation time.

The proposed FL approach has multiple phases: Initialisation, Client Selection, Cluster Formation and Master Selection, Training, and Aggregation phase. Initially, one satellite per orbit receives the model from GS and distributes it to its intra-orbital neighbours [26]. Upon distribution, satellites advance to the next phase. Satellites are divided into c clusters, where satellite $j \in \{1, 2, \ldots, l\}$ in orbit i gathers and retains a dataset $D_{i,j}$. This local dataset is utilised to train a machine learning model using the Federated Average peer-to-peer [20]. Each cluster C_k ($k \in \{1, 2, \ldots, c\}$) comprises a collection of satellites, wherein the training process is coordinated by one of the satellites within the cluster. The satellite, designated as the *master* satellite, transmits a request to its cluster neighbours within the ISL range at round t, to establish a cluster and get their updated weights for aggregation. Upon completion of the parameter distribution and clustering phases, all satellites commence their local training and aggregation phases, utilising the computational model delineated in Section III.A. Thereafter, the master satellites convey their revised weights to the GS at the subsequent communication opportunity. The GS executes the global aggregation phase by integrating the updated weights received from the master satellites and disseminates the new global model to the satellites during the subsequent communication interval.

A. Computation and Communication Model

Satellites collaborate to learn the global model by minimising a global objective function in cluster k as follows:

$$w^* = \min F_k(w) \tag{1}$$

The global loss function is calculated as:

$$F_k(w) = \sum_{i,j \in O,S} \frac{D_{(i,j)k}}{D_k} f_{i,j}(w) \tag{2}$$

where $D_{i,j} = |D_{i,j}|$, $D_k = \sum_{i,j \in OS} D_{(i,j)k}$ is the size of the whole dataset in the cluster k, w is the model parameter, and

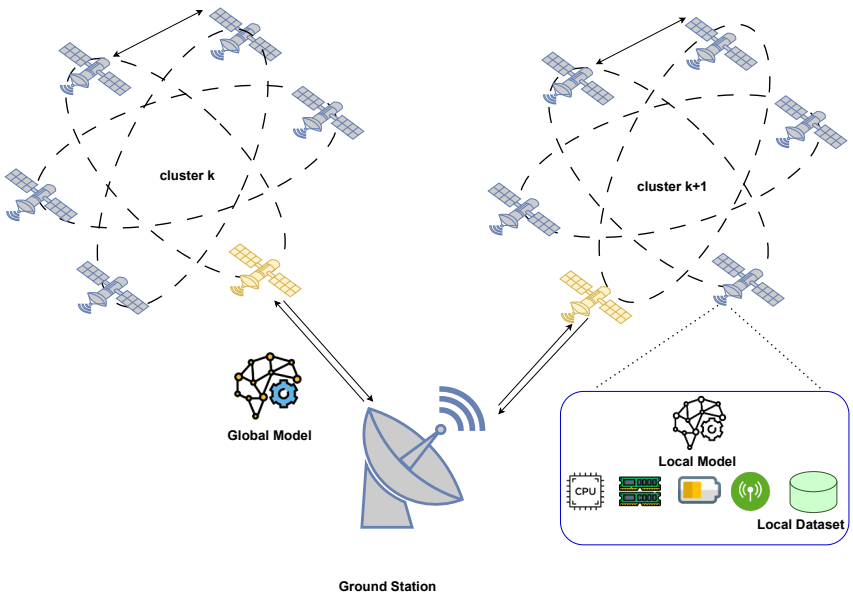

Fig. 1. Orbital Edge Computing Architecture

$f_{i,j}(w)$ is the local loss function that is computed for each dataset $D_{i,j}$ as follows:

$$f_{i,j}(w) = \frac{1}{D_{i,j}} \sum_{x \in D_{i,j}} f_{i,j}(w; x) \tag{3}$$

where $f_{i,j}(w; x)$ is training loss for a data point x and model parameter w. Each satellite receives its own model via the first phase and can communicate with its cluster master satellite via ISLs and continue the collaboration as proposed in [15].

In this paper, we assume the existence of free space optics (FSO) communication links between satellites to tackle sporadic connectivity with ground stations. These linkages are classified into four types: terrestrial, aerial (nonterrestrial), space, and deep-space [13]. In this study, FSO links between satellites are referred to as InterSatellite Links (ISLs). In our approach, GS communication is only required at the initialisation phase and final aggregation at each training step, where master satellites send their clusters' local updates to GS for final aggregation. For successful communication, the satellite must be positioned within the ground station's field of view, exceeding a specified minimum elevation angle above the horizon, so that satellites and GS are visible to each other. The minimum elevation angle ζ_e, the satellite $S_{oz,i}$ and GS can visit each other if $\frac{\pi}{2} - \angle(p_{GS}, p(o_{z,i}) - p_{GS}) \geq \zeta_e$, where p_{GS} and $p(o_{z,i})$ present the position of GS and satellite $S_{oz,i}$, respectively.

B. Energy Consumption Model

The energy consumption of laser communication satellites can be categorised into three main components: (i) the utilisation of fundamental functions, such as sustaining the satellite's height and adjusting its solar panels; (ii) the energy consumption of communication; and (iii) the energy consumed by the on-board processor during the training process.

$$E_{\text{total}} = E_{\text{base}} + E_{\text{comm}} + E_{\text{comp}} \tag{4}$$

In this paper, we focus on the communication energy E_{comm}, and the computation energy E_{comp} as the main factor for the consumed energy. The base energy consumption is considered constant. The computational energy consumption of satellites comprises both cluster-based computation energy and aggregation energy. During a training task, a model that incorporates the most critical CPU parameters is considered [31] [21]. The total energy consumption during processing is expressed as:

$$E_{\text{comp}} = E_{\text{comp\_cluster}} + E_{\text{aggregation}} \tag{5}$$

$$E_{\text{comp\_cluster}} = \sum_{k=1}^{c} \sum_{s_{i,j} \in C_k} N_k \cdot (\frac{1}{2} \omega c_{i,j} Z(D_{i,j}) f_{i,j}^2) \tag{6}$$

$$E_{\text{aggregation}} = \sum_{s_{i,j} \in S | master} \frac{1}{2} \omega c_{i,j} M_{i,j} f_{i,j}^2 \tag{7}$$

where $\omega_{i,j}$ represents the effective capacitance coefficient, $c_{i,j}$ is the number of CPU cycles to process one bit of data for $s_{i,j}$, $Z(D_{i,j})$ is the number of bits in dataset, $f_{i,j}$ is the CPU frequency, N_k is the number of nodes in a cluster k, and $M_{i,j}$ is the model size. Please note that equation 7 is only applied for the master satellites.

The communication energy consumption of satellites includes the energy consumed by laser terminals for establishing ISLs and the energy consumed by microwave communication

antennas for connecting GS with satellites. For each component, the energy will be obtained as follows:

$$E_{\text{comm}} = E_{ISL} + E_{GS} \tag{8}$$

The energy consumption for ISL communications is given by:

$$E_{ISL} = \sum_{k=1}^{c} \sum_{s_{i,j} \in C_k} (P_T + aR_{i,j}^{ISL}).T_{i,j} \tag{9}$$

The energy consumption for GS communications is given by:

$$E_{GS} = \sum_{s_{i,j} \in S | master} (P_A + bR_{i,j}^{GS}).T_{i,j} \tag{10}$$

where P_T, P_A are the basic power of the satellite laser terminal and microwave antenna, respectively. Also, a is ISL coefficient, b is GS coefficient with the values of $a = 1$, $b = 5$ [14]. $R_{i,j}^{ISL}$ is the ISL data rate and $R_{i,j}^{GS}$ is the GS data rate. The communication time between satellites or with the GS can be calculated as follows [18]:

$$T_{i,j} = \frac{M_{i,j}}{R_{i,j} \log_2(1 + \frac{Ghp}{\sigma})} \tag{11}$$

where $R_{i,j}$ should be $R_{i,j}^{ISL}$ in Equation 9, and $R_{i,j}^{GS}$ in Equation 10, G is gain to noise temperature ratio, h is system loss, p is transmit power and σ is noise power.

C. Resource Model

Traditional FL approaches select clients randomly, which may lead to inefficiencies when clients with limited resources are chosen, as it increases the energy consumption, the training time, and the system latency. Using predictive models that can predict the resource availability of clients before their selection represents a promising approach to improve the system's performance. Resource prediction enables more intelligent client selection in FL, where it optimises task allocation of computational load according to the predicted availability. Developing prediction models for client resources requires considering the temporal patterns of resource usage, especially for the satellite systems with heterogeneous capabilities of the client's (e.g., battery, CPU, and memory). To quantify the resources availability, the prediction model will predict the percentage of battery, CPU, and memory that is available for each satellite in LEO constellations for model training. Based on that, the estimated training rounds for each satellite with a focus on energy consumption can be calculated as below:

$$R_{i,j} = \min\left(\left\lfloor \frac{b_{i,j}}{E_{total}}.R_{total} \right\rfloor, R_{total}\right) \tag{12}$$

where R_{total} is the total number of rounds in the FL process that a satellite with sufficient resources can perform. Each satellite is scored based on three key resources: battery level ($b_{i,j}$), CPU availability ($u_{i,j}$), and memory availability ($y_{i,j}$). The score is calculated as follows:

$$score_{i,j} = \alpha.\frac{b_{i,j}}{E_{total}} + \beta.u_{i,j} + \gamma.y_{i,j} \tag{13}$$

The weighting constants (α, β, and γ) are assigned values reflecting the relative importance of each resource.

For accurate resource prediction in FL, many machine learning methodologies can be used, and among these, Long Short Term Memory (LSTM) appears to be particularly well suited to this task. LSTMs are a form of recurrent neural network to manage sequential data and capture long-range relationships. In contrast to traditional RNNs, LSTM networks proficiently retain information across prolonged sequences without experiencing the vanishing gradient problem [29]. This capacity arises from their advanced gating mechanism—employing forget, input, and output gates—that manages memory upgrades and inhibits information deterioration over extended sequences. LSTMs are proficient in learning from past observations and preserving that knowledge, rendering them suitable for capturing the nonlinear dynamics characteristic of resource utilisation patterns. Their capacity to consistently adjust to changing data patterns makes them an ideal selection for resource monitoring in distributed systems [10].

IV. The Proposed Algorithm

This section discusses the proposed FedSCS approach. The proposed algorithm has five phases, including initialisation, client selection, clustering and master selection, local training, and aggregation, which are explained in Algorithm 1. The initial phase distributes model parameters to satellites before training begins. Afterwards, client selection is performed based on Algorithm 2. The algorithm applies Clustering and Master Satellite Selection using the reinforcement learning (RL) approach presented in [15], and the master selection is discussed in Algorithm 3. Once master satellites and clusters are established, satellites proceed with local training and aggregation. In the final phase, master satellites transmit updated weights to the GS during their next visit. The GS executes global aggregation using the received updates and broadcasts the new global model to satellites during subsequent visits.

A. Client Selection

Client selection is one of the main steps in FedSCS, which is intended to determine the most appropriate satellites for participation in the current training cycle of FL. The selection procedure prioritises satellites according to resource availability and energy efficiency. The selection process operates through a two-step approach: resource prediction and client scoring. The system relies on a 2-layer LSTM with 64 hidden units and a fully connected layer for the resource prediction. Initially, the system generates random initial values within realistic ranges and begins collecting actual measurements of battery, CPU, and memory usage during each training round. These measurements are normalised and fed into the LSTM model as sequential data, enabling the prediction of future resource states. The system maintains its accuracy by implementing a sliding window approach, where in each training round, new measurements are added while older ones are removed, and the LSTM model is retrained with this updated data.

Algorithm 1 FedSCS Algorithm

Parameter Distribution Phase
Input: communication rounds R; satellites set S; orbits set O; local training epochs n; fraction of neighbours e; learning rate η; number of required clients p
Output: Trained global model parameters
$w_0 \leftarrow$ Initial model parameters
for each round $t = 1, 2, ..., R$ **do**
  Client Selection Phase
  $S_t \leftarrow$ CLIENTSELECTION(S, p, t) in Algorithm 2
  Cluster Formation Phase
  $C_k \leftarrow$ CLUSTERFORMATION(S_t)
  $L \leftarrow$ MASTERSELECTION$(l_{candidates})$ in Algorithm 3
  // Local Training Phase
  for all satellite $s_{i,j} \in$ cluster C_k **do**
    LocalTraining$(s_{i,j}, w)$
    Partition data D: $D_{i,j} \leftarrow (s_{i,j})$
    for each local epoch e from 1 to n **do**
      $w_{i,j}^{t+1} \leftarrow w_{i,j}^{t} - \eta \nabla f_{i,j}(w_{i,j}^{t})$
    end for
  end for
  // Local Aggregation Phase
  for each master satellite L in parallel **do**
    $v = \max\{m \times e, 2\}$
    $Ni, j =$ (random set of v neighbors in cluster k)
    for each neighbor $s_{i,j} \in N_{i,j}$ in parallel **do**
      $w_{i,j}^{t} =$ LocalTraining$(s_{i,j}, w)$
    end for
    $w_{i,j}^{t+1} = w_{i,j}^{t} - \eta \nabla f_{i,j}(w_{i,j}^{t}) + \sum_{s \in N_{i,j}} (w_{i,j} - \eta \nabla f_{i,j}(w_{i,j}^{t}))$
  end for
  // Global Aggregation phase
  GS receives local updates $w_{i,j}^{t+1}$ from all master satellites
  $w^{t+1} \leftarrow \frac{1}{c} \sum_{j \in S} \sum_{i \in O} \frac{D_{i,j}}{D} w_{i,j}^{t}$
end for=0

The actual selection process as presented in Algorithm 2 begins with inputs including the complete set of available satellites, the required number of clients, the current round, and resource parameters (battery, CPU, memory). For each satellite, the system first uses the LSTM to predict future resource availability through three functions, PREDICTBATTERY, PREDICTCPU, PREDICTMEMORY. Each function processes normalised historical data through dedicated input features. The PREDICTBATTERY function considers battery consumption rates, implements a decay factor that prioritises recent measurements, and accounts for solar recharging patterns. The Battery Consumption Rate is given as:

$$CR_t = B_t - B_{t-1} \tag{14}$$

Where B_t is the battery level at time t. Decay Factor is given as [35]:

$$DF_t = e^{-0.1 \cdot \frac{T-t}{T}} \tag{15}$$

where T is the total time sequence length. Solar Recharging Pattern is given as [7] :

$$ToD_t = \frac{t \bmod 24}{24} \tag{16}$$

to capture the 24-hour solar cycle. The battery feature vector:

$$X_t^{\text{battery}} = [B_t, CR_t, DF_t, ToD_t] \tag{17}$$

The values are normalised and passed through the model, the output value is inverse transformed, and the predicted battery $\hat{B}_{t+1}$ for the next time step is found. The PREDICTCPU function analyses the utilisation rates and captures the training cycle. CPU utilisation rate is calculated as:

$$UR_t = CP_t - CP_{t-1} \tag{18}$$

where CP_t is the CPU usage percentage at time t. The training cycle is given as [6]:

$$TC_t = \frac{t \bmod 8}{8} \tag{19}$$

This captures the 8-hour training cycle. The CPU feature vector is:

$$X_t^{\text{CPU}} = [CP_t, UR_t, TC_t] \tag{20}$$

All features are normalised and passed through the model, the output value is inverse transformed, and the CPU predicted value $\hat{C}_{t+1}$ is predicted for the next round. The PREDICTMEMORY function examines usage patterns and memory release information. It checks the memory growth as below:

$$GR_t = M_t - M_{t-1} \tag{21}$$

where M_t is the memory usage at time t. The memory release cycle in 12 hours is given as [16]:

$$MR_t = \frac{t \bmod 12}{12} \tag{22}$$

The memory vector feature is:

$$X_t^{\text{memory}} = [M_t, GR_t, MR_t] \tag{23}$$

The features are normalised, passed through the model, and the output value is inverse transformed to find the predicted memory $\hat{M}_{t+1}$ in the next step. The system continuously monitors prediction accuracy by comparing predictions with actual measurements and adjusts weights accordingly. The selection process begins with these predictions for each satellite, which calculates energy requirements by computing both computational energy based on CPU and memory usage, and communication energy, combining these for total energy requirements.

Satellites with available battery power are evaluated using two metrics: possible participation rounds as explained in Equation 12 to determine how many rounds they can participate in without exceeding their resource limitations, and resource-weighted score as given in Equation 13. Clients are scored and sorted in descending order, and the m satellites with the highest scores are selected for participation in the training round, forming the final selected set. After that, all

Algorithm 2 Client Selection Algorithm

Input: Set of satellites S, required clients m, current round t, battery b, CPU u, memory y
Output: Selected set of satellites S_t
$scores \leftarrow \emptyset$
for each satellite $s_{i,j} \in S$ **do**
  // Resource Prediction
  $b_{(i,j)} \leftarrow$ PREDICTBATTERY$(_{i,j})$ **Equation 17**
  $u_{(i,j)} \leftarrow$ PREDICTCPU$(_{i,j})$ **Equation 20**
  $y_{(i,j)} \leftarrow$ PREDICTMEMORY$(_{i,j})$ **Equation 23**
  // Energy Calculations
  $E_{comp} \leftarrow$ COMPUTEECOMP$(_{i,j})$ **Equation 5**
  $E_{comm} \leftarrow$ COMPUTEECOMM$(_{i,j})$ **Equation 8**
  $E_{total} \leftarrow E_{comp} + E_{comm}$
  if $b_{(i,j)} > 0$ **then**
    $possible\_rounds_{(i,j)} \leftarrow$ ROUNDS **Equation 12**
    $score_{(i,j)} \leftarrow$ SCORE **Equation 13**
    $scores.append((s_{i,j}, score_{(i,j)}))$
  end if
end for
Sort $scores$ by $score_{(i,j)}$ in descending order
$S_t \leftarrow$ First m satellites from $scores$
return S_t =0

prediction functions continuously improve through feedback loops that measure actual resource usage after each training round, calculate prediction errors, incorporate new measurements through sliding windows, and periodically retrain models and tune hyperparameters, ensuring the system adapts to changing satellite conditions and maintains efficiency for the selection process.

B. Clustering and Master Selection

Due to the rapid movement of satellites, their communication window with a specific GS is limited. Given the intermittent yet predictable connectivity of satellites due to orbital motion, it is crucial to consider these patterns to predict their connectivity times with GS. To determine the optimal number of satellites per cluster, we utilise the cluster determination algorithm proposed in [15], which applies a reinforcement learning (RL) approach using deep Q-learning and takes into account the total number of satellites, communication range, and master satellite count. The model is trained on a predefined reward function to facilitate efficient cluster formation. The RL agent performs a number of clustering operations, such as satellite addition and removal, and cluster formation and dissolution. Penalty mechanisms are triggered by invalid clustering decisions, such as exceeding cluster capacity. The model is trained on a predefined reward function to facilitate efficient cluster formation.

In the cluster formation stage, master satellites are selected from satellites that are scheduled to visit the GS immediately after completing their cluster training. A satellite qualifies as a master satellite candidate if its visiting time with the GS falls within the window of training completion [15]. In Algorithm 3 line 1, a set of candidate master satellites is chosen as a master candidate based on their visiting patterns and ensuring finishing cluster training time. In lines 5-9, to find the best satellite among the candidates, the communication bandwidth is scored, and the candidate with the best score is selected as a master satellite. This ensures that the satellite with the highest bandwidth is selected as the master, optimising transmission efficiency. Following master satellite selection and cluster formation, all satellites initiate local training and aggregation. After that, master satellites send their updated parameters to the GS for aggregation. The GS receives the updates and broadcasts a new global model to satellites during their next visits.

Algorithm 3 Master Selection Algorithm

1: **Input:** Set of candidate master satellites $L_{candidates}$
2: **Output:** Selected master satellite $l_{i,j}^{best}$
3: $l_{i,j}^{best} \leftarrow$ null
4: $best\_comm\_score \leftarrow -\infty$
5: **for all** candidate $l_{i,j} \in L_{candidates}$ **do**
6: $comm\_score \leftarrow$ GETCOMMUQUALITY$(l_{i,j})$
7: **if** $comm\_score > best\_comm\_score$ **then**
8: $best\_comm\_score \leftarrow comm\_score$
9: $l_{i,j}^{best} \leftarrow l_{i,j}$
10: **end if**
11: **end for**
12: **return** $l_{i,j}^{best}$ =0

TABLE I
EXPERIMENTAL PARAMETERS

Parameter	Value
Constellation	Walker Delta (Starlink)
Number of LEO satellites	720
Number of orbits	36
Number of LEOs per Orbit	20
Inclination	70°
Altitude	570 km
Bandwidth of ISL/GS2S	2.5/1.25 GHz
System loss (h)	3 dB
Noise power (σ)	2.2×10^{-16} W
Frequency	27 GHz
Gain to noise temperature ratio (G)	5 dB/K
Data rate	16 Mbps
Transmit power (p)	40 Watt
Total FL rounds	40
Local training epochs	30
Batch size	10

TABLE II
CLIENT SELECTION SCORING WEIGHTS

Parameter	Weight
Battery efficiency (α)	0.6
CPU utilisation (β)	0.2
Memory usage (γ)	0.2

V. PERFORMANCE EVALUATION

Experiments across various models and datasets were performed to demonstrate the performance of the proposed algorithm. The model accuracy, energy consumption, and training

time are the evaluation metrics in this work. We developed the FedSCS algorithm in the Flower framework [1], combined with the machine learning framework PyTorch for FL implementation [23]. The Satellite Communications Toolbox in MATLAB is employed to analyse and compute the visiting patterns of LEO satellites relative to the GS.

A. Experimental Setup

A Walker-Delta constellation is considered, comprising 720 LEO satellites with an inclination of 70° and an altitude of 570 km. The constellation is structured into 36 orbital planes (OPs), each containing 20 satellites, with an inter-plane spacing of 10° and an intra-plane satellite spacing of 18°. Each satellite is assigned a unique identifier (ID) based on its orbital plane and position, resulting in 720 distinct IDs. A ground station is positioned in Canberra, Australia, at latitude -35.40139 and longitude 148.98167. The ISL range varies within $\{659, 1319, 1500, 1700\}$ km, enabling a master satellite to form a cluster with 2, 4, 6, or 10 neighbouring satellites. For resource availability (CPU, memory, and battery), we consider random values in [20%-40%] for each satellite, considering that they have several other tasks to perform for their space mission. To ensure a fair comparison, we use the same parameters employed in previous studies [14] in Table I. Also, Table II presents the client selection parameters, prioritising energy efficiency with a higher weight.

Baselines: Our proposed approach is compared with several existing approaches, including FedSyn [22], FedProx [17], FELLO [3], and FedOrbit [15]. FedSyn represents the traditional centralised FL approach, where local models are transmitted to a central GS for aggregation. FedProx is a decentralised FL framework designed to tackle heterogeneity in FL networks. FELLO is a decentralised FL framework for LEO satellites, where the GS selects a master LEO satellite as the FL edge server. This edge server clusters satellites based on optical ISLs and communication link quality. FedOrbit is a hierarchical FL for LEO satellites, minimising GS dependency through reinforcement learning-based clustering. The setting utilises 40 satellites sampled from the constellation. The satellites are clustered with RL with the formation (5-7-3-3-5-7-3-7) where the numbers represent the size of each cluster [15].

Datasets and Models: We used the MNIST and CIFAR-10 in the IID and non-IID data format. Additionally, we incorporate EuroSat [12], a real-world satellite dataset containing 27,000 images of 64×64 pixels. MNIST and CIFAR-10 are trained using a deep convolutional neural network (CNN). EuroSat is trained using the ResNet18 model, which is well-suited for high-resolution satellite imagery classification.

B. Experimental Results

This section provides a detailed comparative analysis of the experimental results, emphasising the performance of FedSCS and the impact of the client selection algorithm on the consumed energy reduction. Figure 2 shows the comparison of the resource's predicted and actual values over time (left: full range, right: zoomed view). The predicted value for the battery is approximately 3% higher than the actual ones, while the predicted value for CPU and memory is 2.5%, 4% higher than the actual, respectively. This reveals that we can predict the resource availability with relatively high accuracy, which will be the key factor in the proposed client selection algorithm.

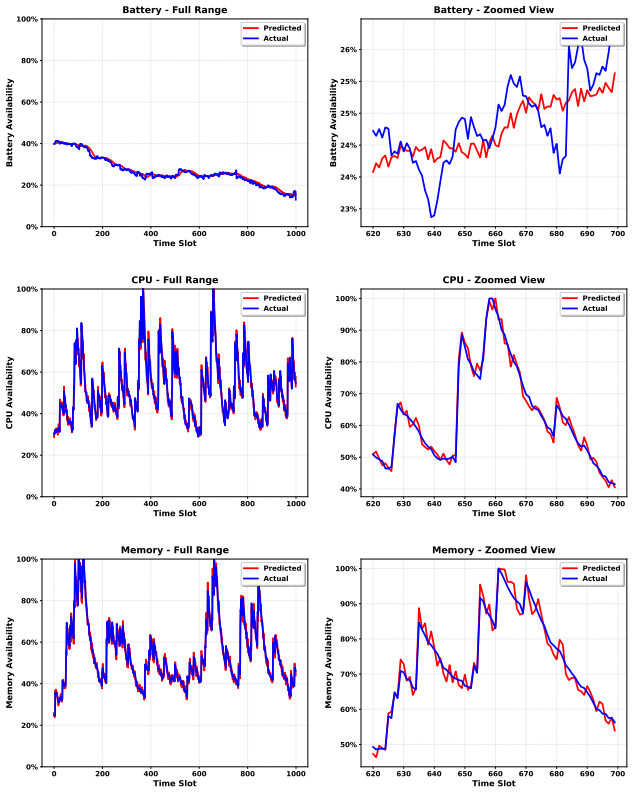

Fig. 2. Resources Availability Prediction (Battery, CPU, Memory)

As plotted in Figure 3 for MNIST dataset, FedSCS achieved 86% accuracy, outperforming FedOrbit by 3.61%, FELLO by 10.26%, and FedProx by 8.86%. For CIFAR10 dataset, Figure 4 shows that FedSCS achieved 85% accuracy and FedOrbit 82%, and outperforms FELLO and FedProx by 10.39% and 11.84%, respectively. FedProx approach had the lowest final accuracy and the slowest improvement rate. FedSyn as a centralised approach, had the highest accuracy among the baselines. When evaluated on real satellite non-IID image dataset using ResNet18 model, FedSCS achieves 84% accuracy and demonstrates a strong convergence rate and higher accuracy compared to FedOrbit, FELLO, and FedProx. FedSCS surpassed 80% accuracy by round 12, whereas FedOrbit only reached 76%, reinforcing that FedSCS had a more stable and efficient learning process as shown in Figure 5.

For energy consumption and training time, metrics are at a consistent value of accuracy, which is stated as a comparable accuracy value (Comp. Acc.). Figure 6 shows the energy consumption and training time for the CNN model and MNIST dataset at Comp. Acc. of 79%. FedSCS achieved the lowest

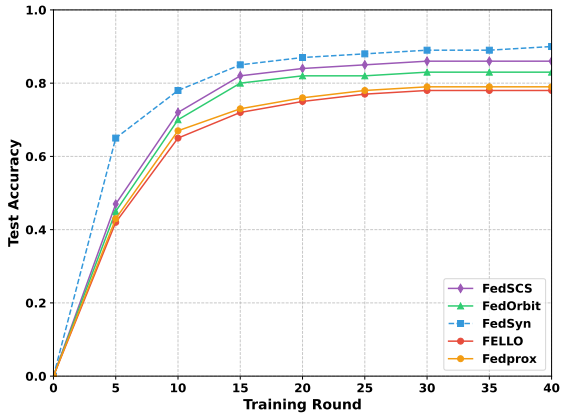

Fig. 3. Accuracy for CNN-MNIST (IID)

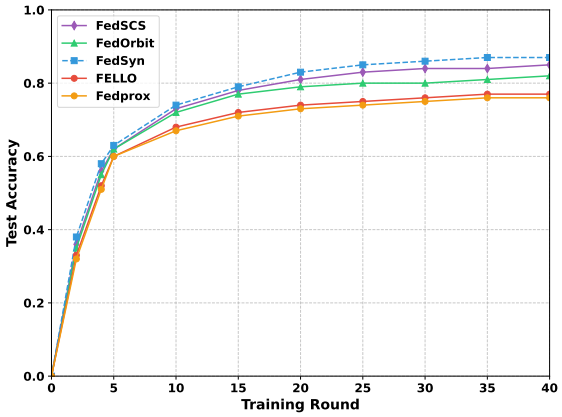

Fig. 4. Accuracy for CNN-CIFAR10 (IID)

power consumption in 2.5 hours training time, making it faster than FedSyn by 86.98% and 16.67% faster than FedOrbit and less by 6.76% of the consumed energy. Figure 7 shows the consumed energy for the CNN model and the CIFAR10 dataset and training time for Comp. Acc. of 77%. FedSCS completed training time 17.95% faster than FedOrbit and 10.34% less consumed energy as well. In Figure 8 the Comp. Acc. is set to 75% for the ResNet18 model and the EuroSat dataset. It shows that FedSCS training time is 9.09% faster than FedOrbit, 46.81% faster than FELLO, and 62.83% faster than FedProx. The consumed energy is 9.09% less power than FedOrbit, 39.2% less than FELLO, and 57.45% less than FedProx. This is due to the utilisation of the energy efficient client selection in satellites and the communication selection strategy for the master satellite.

Table III shows the accuracy, training time and energy consumption for MNIST and CIFAR10 in the non-IID setting for FedSCS and FedOrbit. For MNIST, the proposed FedSCS outperforms FedOrbit by 3.57% in accuracy and 30% reduced training time. For the CIFAR10 dataset, FedSCS achieved 80% accuracy while FedOrbit achieved 77% and the training time was reduced by 23.08%. The energy consumption is reduced by 39.7%, 36.3% for MNIST and CIFAR10, respectively. FedSCS outperforms FedOrbit with a reduced energy consumption by 6.67%, 10.34% for MNIST, CIFAR10 in the IID setting and 39.7% and 36.3% in the non-IID setting, respectively. This is due to selecting sufficient clients that achieve fast convergence and the master selection approach, showing great efficiency in the non-IID setting.

VI. Conclusion and Future Work

In this paper, we proposed an energy-efficient client selection FL approach for LEO satellite constellations. FedSCS introduces a resource-aware client selection strategy that dynamically prioritises satellites based on their available computational resources, ensuring the selection of the most suitable clients. FedSCS incorporates a resource prediction model and client selection algorithm to enhance training

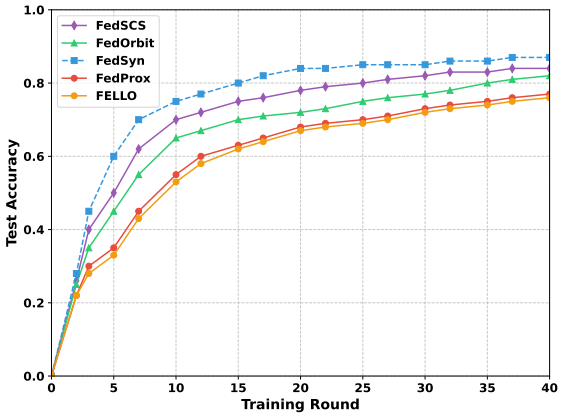

Fig. 5. Accuracy for ResNet18-EuroSat (non-IID)

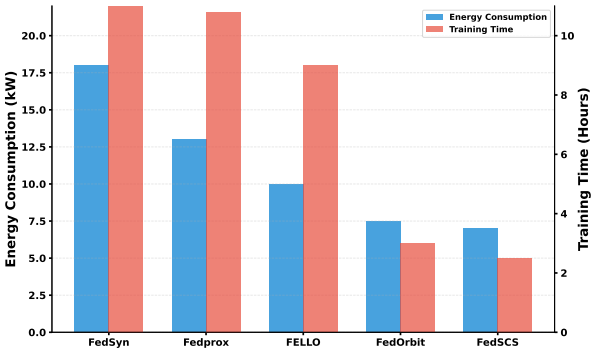

Fig. 6. CNN-MINST (Comp. Acc. 79%)

efficiency and minimise energy consumption. Experimental results show that FedSCS surpasses existing centralised and decentralised FL techniques by reducing training time and energy consumption for both IID and non-IID datasets while achieving a slight improvement in accuracy. Future work will focus on optimising cluster formation to enhance efficiency

TABLE III
PERFORMANCE COMPARISON FOR NON-IID DATASETS

FL Approaches	Accuracy (%)		Training Time (h)		Energy (kW)	
	MNIST	CIFAR10	MNIST	CIFAR10	MNIST	CIFAR10
FedSCS	84	80	5.6	10	5	7
FedOrbit	81	77	8	13	8.3	11

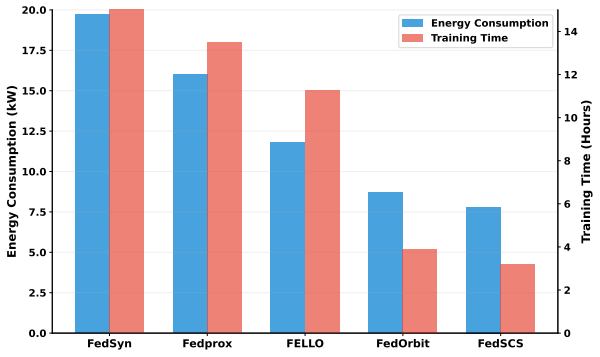

Fig. 7. CNN-CIFAR10 (Comp. Acc. 77%)

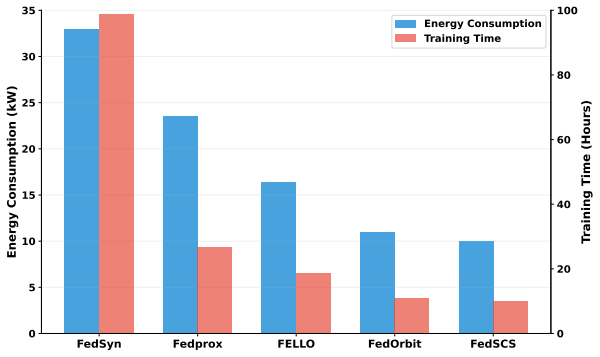

Fig. 8. ResNet18-EuroSat (Comp. Acc. 75%)

and further minimise energy consumption.

REFERENCES

[1] Daniel J Beutel, Taner Topal, Akhil Mathur, Xinchi Qiu, Javier Fernandez-Marques, Yan Gao, Lorenzo Sani, Kwing Hei Li, Titouan Parcollet, Pedro Porto Buarque de Gusmão, et al. Flower: A friendly federated learning research framework. *arXiv preprint arXiv:2007.14390*, 2020.

[2] Aizaz U Chaudhry and Halim Yanikomeroglu. Laser intersatellite links in a starlink constellation: A classification and analysis. *IEEE vehicular technology magazine*, 16(2):48–56, 2021.

[3] Chih-Yu Chen, Li-Hsiang Shen, Kai-Ten Feng, Lie-Liang Yang, and Jen-Ming Wu. Edge selection and clustering for federated learning in optical inter-leo satellite constellation. In *2023 IEEE 34th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, pages 1–6. IEEE, 2023.

[4] Hao Chen, Ming Xiao, and Zhibo Pang. Satellite-based computing networks with federated learning. *IEEE Wireless Communications*, 29(1):78–84, 2022.

[5] Bradley Denby and Brandon Lucia. Orbital edge computing: Machine inference in space. *IEEE Computer Architecture Letters*, 18(1):59–62, 2019.

[6] Peter A Dinda. Design, implementation, and performance of an extensible toolkit for resource prediction in distributed systems. *IEEE Transactions on Parallel and Distributed Systems*, 17(2):160–173, 2006.

[7] Mian Dong and Lin Zhong. Self-constructive high-rate system energy modeling for battery-powered mobile systems. In *Proceedings of the 9th international conference on Mobile systems, applications, and services*, pages 335–348, 2011.

[8] Mohamed Elmahallawy and Tie Luo. Asyncfleo: Asynchronous federated learning for leo satellite constellations with high-altitude platforms. In *2022 IEEE International Conference on Big Data (Big Data)*, pages 5478–5487. IEEE, 2022.

[9] Mohamed Elmahallawy and Tie Luo. Optimizing federated learning in leo satellite constellations via intra-plane model propagation and sink satellite scheduling. In *ICC 2023-IEEE International Conference on Communications*, pages 3444–3449. IEEE, 2023.

[10] Tolga Ergen and Suleyman S Kozat. Online training of lstm networks in distributed systems for variable length data sequences. *IEEE transactions on neural networks and learning systems*, 29(10):5159–5165, 2017.

[11] Lei Fu, Huanle Zhang, Ge Gao, Mi Zhang, and Xin Liu. Client selection in federated learning: Principles, challenges, and opportunities. *IEEE Internet of Things Journal*, 2023.

[12] Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(7):2217–2226, 2019.

[13] Louis J Ippolito Jr. *Satellite communications systems engineering: atmospheric effects, satellite link design and system performance*. John Wiley & Sons, 2017.

[14] Mohammad Reza Jabbarpour, Bahman Javadi, Philip Leong, Rodrigo N. Calheiros, David Boland, and Chris Butler. Performance analysis of federated learning in orbital edge computing. In *Proceedings of the IEEE/ACM 16th International Conference on Utility and Cloud Computing*, pages 1–10, 2023.

[15] Mohammad Reza Jabbarpour, Bahman Javadi, Philip HW Leong, Rodrigo N Calheiros, and David Boland. Fedorbit: Energy efficient federated learning for orbital edge computing using block minifloat arithmetic. *IEEE Transactions on Services Computing*, 2024.

[16] Richard Jones, Antony Hosking, and Eliot Moss. *The garbage collection handbook: the art of automatic memory management*. Chapman and Hall/CRC, 2012.

[17] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *Proceedings of Machine learning and systems*, 2:429–450, 2020.

[18] Lumin Liu, Jun Zhang, SH Song, and Khaled B Letaief. Client-edge-cloud hierarchical federated learning. In *2020 IEEE international conference on communications (ICC)*, pages 1–6. IEEE, 2020.

[19] Siyang Liu, Tao Meng, Zhonghe Jin, and Renting Song. Optimal deployment of heterogeneous microsatellite constellation based on kuhn-munkres and simulated annealing algorithms. *Journal of Aerospace Engineering*, 35(6):04022090, 2022.

[20] Dylan Mäenpää. Towards peer-to-peer federated learning: Algorithms and comparisons to centralized federated learning, 2021.

[21] Yuyi Mao, Changsheng You, Jun Zhang, Kaibin Huang, and Khaled B Letaief. A survey on mobile edge computing: The communication perspective. *IEEE communications surveys & tutorials*, 19(4):2322–2358, 2017.

[22] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.

[23] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein,

Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[24] Tobias Pfandzelter, Jonathan Hasenburg, and David Bermbach. Towards a computing platform for the leo edge. In *Proceedings of the 4th International Workshop on Edge Systems, Analytics and Networking*, pages 43–48, 2021.

[25] Nasrin Razmi, Bho Matthiesen, Armin Dekorsy, and Petar Popovski. Ground-assisted federated learning in leo satellite constellations. *IEEE Wireless Communications Letters*, 11(4):717–721, 2022.

[26] Nasrin Razmi, Bho Matthiesen, Armin Dekorsy, and Petar Popovski. On-board federated learning for dense leo constellations. In *ICC 2022-IEEE International Conference on Communications*, pages 4715–4720. IEEE, 2022.

[27] Nasrin Razmi, Bho Matthiesen, Armin Dekorsy, and Petar Popovski. Scheduling for ground-assisted federated learning in leo satellite constellations. In *2022 30th European Signal Processing Conference (EUSIPCO)*, pages 1102–1106. IEEE, 2022.

[28] Nasrin Razmi, Bho Matthiesen, Armin Dekorsy, and Petar Popovski. On-board federated learning for satellite clusters with inter-satellite links. *IEEE Transactions on Communications*, 2024.

[29] Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. *Advances in neural information processing systems*, 28, 2015.

[30] Jinhyun So, Kevin Hsieh, Behnaz Arzani, Shadi Noghabi, Salman Avestimehr, and Ranveer Chandra. Fedspace: An efficient federated learning framework at satellites and ground stations. *arXiv preprint arXiv:2202.01267*, 2022.

[31] Yuanjun Wang, Jiaxin Zhang, Xing Zhang, Peng Wang, and Liangjingrong Liu. A computation offloading strategy in satellite terrestrial networks with double edge computing. In *2018 IEEE international conference on communication systems (ICCS)*, pages 450–455. IEEE, 2018.

[32] Changhao Wu, Siyang He, Zengshan Yin, and Chongbin Guo. Towards client selection in satellite federated learning. *Applied Sciences*, 14(3):1286, 2024.

[33] Chenrui Wu, Yifei Zhu, and Fangxin Wang. Dsfl: Decentralized satellite federated learning for energy-aware leo constellation computing. In *2022 IEEE International Conference on Satellite Computing (Satellite)*, pages 25–30. IEEE, 2022.

[34] Lingling Wu and Jingjing Zhang. Fedgsm: Efficient federated learning for leo constellations with gradient staleness mitigation. In *2023 IEEE 24th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, pages 356–360. IEEE, 2023.

[35] Lide Zhang, Birjodh Tiwana, Zhiyun Qian, Zhaoguang Wang, Robert P Dick, Zhuoqing Morley Mao, and Lei Yang. Accurate online power estimation and automatic battery behavior based power model generation for smartphones. In *Proceedings of the eighth IEEE/ACM/IFIP international conference on Hardware/software codesign and system synthesis*, pages 105–114, 2010.