A Novel Hybrid Split and Federated Learning Architecture in Wireless UAV Networks

Xiaolan Liu*, Yansha Deng †, Toktam Mahmoodi †
* Loughborough University, †King's College London
Email:*xiaolan.liu@lboro.ac.uk, †{yansha.deng, toktam.mahmoodi}@kcl.ac.uk

Abstract—**The ever-growing use of unmanned aerial vehicles (UAVs) as flying users is becoming a major part of the sixth generation (6G) networks, which could provide various applications, like object detection and video surveillance, by exploiting machine learning (ML) algorithms. However, the training of conventional centralized ML algorithms causes high communication overhead due to the transmission of large datasets and may reveal user privacy. Hence, distributed learning algorithms, including federated learning (FL) and split learning (SL), are proposed to train ML models in a distributed manner via sharing model parameters rather than raw data. Due to the different learning structures, they have different communication and learning efficiency. We propose a new distributed learning architecture, namely hybrid split and federated learning (HSFL), by adopting the parallel model training mechanism of FL and the network splitting structure of SL. Through the simulations in wireless UAV networks, the HSFL algorithm is demonstrated to have higher learning accuracy than FL and less communication overhead than SL under non-IID data. We further propose a Multi-Arm Bandit (MAB) based best channel (BC) and best 2-norm (BN2) (MAB-BC-BN2) UE selection scheme to select the UEs with better channel quality and larger local model updates in each round. Numerical results demonstrate it achieves higher learning accuracy than the benchmark schemes, BC, MAB-BC, and MAB-BN2 UE selection schemes.**

I. Introduction

Wireless unmanned aerial vehicles (UAVs) network is foreseen to be an integral component in the upcoming sixth-generation (6G) networks [1], [2], which enables various applications, such as video streaming, fire and flood surveillance. To accomplish those complex tasks, UAV-users (UAV-UEs) fly over the area under the control of the base stations (BSs) to collect data (e.g., images and videos). Each UAV-UE *(short-handed hereinafter as UE)* can collect a sub-dataset that only contains partial environment information, so transmitting all the collected sub-datasets from the UEs to the BSs can get better data processing (e.g., training machine learning (ML) models including deep neural network (DNN) and convolutional neural network (CNN)). However, due to the transmission of large datasets, the traditional centralized ML model training at the BS causes high communication overhead, large delay, and large energy consumption for the UEs, as well as requires wide bandwidth and may potentially reveal user privacy [3]. Thanks to the increased powerful UEs, distributed learning becomes more attractive by enabling local learning at the UEs and model aggregation at the BS with only sharing model parameters rather than raw data.

The two state-of-art distributed learning algorithms, federated learning (FL) and split learning (SL), have different learning architectures and therefore are suitable for different application scenarios. In FL, all the UEs collaboratively train an entire ML model (e.g., DNN) with a central server collecting and performing model aggregation using the received local model updates from the UEs [4]. Different from FL, SL was recently proposed in [5], [6] by splitting the entire ML model into several sub-models with the cut layer and distributing them to different entities for distributed training, which facilitates distributed learning via sharing the smashed data of the cut layer. In this case, SL limits the UE-side model down to a few layers, thus reducing the computational overhead of the UEs compared to FL. In practice, considering the diversity of the UEs with different computational capabilities, heterogeneous resources, and data distributions, either deploying FL or SL may not be efficient. In [7], splitfed learning (SFL) was proposed to exploit the parallel model training mechanism in FL and model splitting structure of SL. It shortens the training time in SL and becomes more communication efficient than FL when the number of UEs is large. However, the SFL algorithm still exhibits high communication overhead similar to SL when the number of UEs is small and the dataset over UEs is highly imbalanced. To address this, there is an urgent need to propose a hybrid solution that can well leverage the advantages from both FL and SL even for small number of UEs and the highly imbalanced datasets.

While deploying distributed learning algorithms in wireless networks, not all the UEs can access the BS in each communication round due to the unreliable and randomly fading wireless channels, so it's essential to select an appropriate subset of UEs for model aggregation in each round. The UE selection schemes in FL have been widely investigated by selecting the UEs considering the UE information (e.g., the channel qualities and the importance of local model updates) and resource information (e.g., power, spectrum, and computational resource) [8]–[11]. The UE selection in FL has been studied either based on channel qualities [8], [9] or based on the importance of local model updates [10], [11]. In [11], [12], the UE selection scheme taking into account both channel conditions and the importance of local model updates was proposed. However, they assumed that the UE information was known in advance, which is difficult to be obtained accurately in practice and also consumes extra computation and communication resources to estimate each UE's local model updates before UE selection. To address this, the dynamic UE selection scheme for FL based on Multi-Arm Bandit (MAB) algorithm has been studied [13]–[15], in which the FL operator selects the UEs through exploration and exploitation processes according to the estimated local model updates of UEs [13] or the estimated channel qualities of UEs [14], [15]. Unfortunately, the dynamic UE selection scheme jointly considering both of them have never been explored.

The main contribution of this work is to develop a novel distributed learning architecture, a hybrid split and federated learning (HSFL) approach, and a new MAB-based UE selection scheme. Our key contributions are stated as follows:

- We propose a novel HSFL algorithm to reap the advantages from the FL and the SL, which allows a fraction of UEs to train an entire ML model locally, named as federated training; and allows the other UEs to train the entire model collaboratively with the BS (a sub-model locally and another sub-model at the BS), named as split training. Our results show that the learning accuracy follows: SL > HSFL > FL/SFL, under non-independent and identically distributed (non-IID) data distribution.
- To select a subset of UEs with better wireless channel quality and larger local model updates for participating in model aggregation in each round, we propose a novel MAB-best channel (BC) -best l_2-norm (BN2) (MAB-BC-BN2) UE selection scheme based on the discounted MAB algorithm for our proposed HSFL framework.
- Our results demonstrate that the proposed HSFL algorithm achieves around half less communication overhead and faster convergence performance than SL and SFL. Moreover, the HSFL achieves higher communication efficiency than FL, and which improves with the increasing number of UEs. The proposed MAB-BC-BN2 UE selection scheme is demonstrated to have better learning accuracy performance than the benchmarks under non-IID data.

II. System Model and Problem Formulation

As illustrated in Fig. 1, we consider a single-cell wireless UAV network, consisted of a BS located at the center of the cell, and a set of UAV-UEs *(short-handed hereinafter as UEs)* $\mathcal{N} = \{u_1, ..., u_N\}$ distributed in the BS coverage area as predefined flight paths and remained spatially static during the ML model training process. In this network, the total system bandwidth B_w is equally divided into M radio access channels, where $M < N$. The BS S is assumed to have a single antenna and equipped with high computational capability, and located at the origin of the 3D coordinates system with the antenna installed at the altitude h_s above the ground. Each UE is also equipped with a single antenna and a lightweight GPU (e.g., NVIDIA JETSON) [16]. We assume that all the UEs transmit data with a constant power $P_n = P$. The location of UE u_n is denoted as (x_n, y_n, h_n). Each UE is assumed to fly at the fixed altitude h_n above the ground while the horizontal coordinates (x_n, y_n) of each UE vary over time.

A. Channel Propagation Model

The UE-BS communication link is an air-to-ground (ATG) propagation channel, where the possibility of occurrence of line of sight (LoS) link is affected by the environment. We adopt the channel model presented in [17], which shows the communication paths of ATG channels depend on both LoS and NLoS propagations. Thus, we consider the spatial expectation of the pathloss for LoS and NLoS groups as the

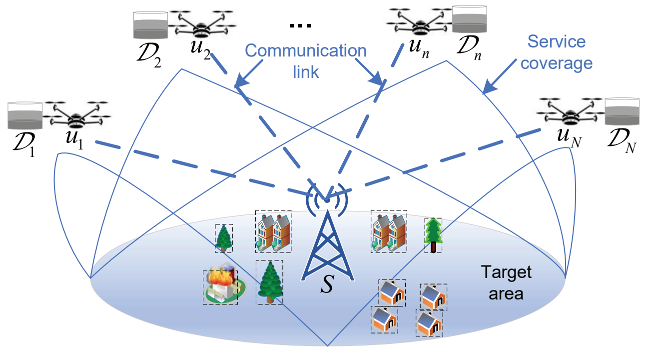


Fig. 1. System model

pathloss model to describe the UE-BS channel, which is given by

$$\bar{\xi}_{ij} = \mathbb{P}^{LoS}_{n,s}\varphi_l(\frac{4\pi f \cdot dist_{n,s}}{c})^{\alpha} + \mathbb{P}^{NLoS}_{n,s}\varphi_n(\frac{4\pi f \cdot dist_{n,s}}{c})^{\alpha}, \tag{1}$$

where $\mathbb{P}^{NLoS}_{n,s} = 1 - \mathbb{P}^{LoS}_{n,s}$ is the NLoS probability, f is the system carrier frequency, c is the light speed, and α denotes the path loss exponent, φ_l and φ_n are the additional path loss coefficients of LoS and NLoS, respectively. $\mathbb{P}^{LoS}_{n,s}$ is the probability of LoS communication links, which is given as

$$\mathbb{P}^{LoS}_{n,s} = \frac{1}{1 + a \cdot \exp(-b[\theta_{n,s} - a])}, \tag{2}$$

where a and b are the environmental parameters indicating the type of environment, like rural or urban, $\theta_{n,s}$ is the elevation angle of the UE-BS communication link. In (2), $\theta_{n,s} = \frac{180}{\pi} \times sin^{-1}(\frac{h_n - h_s}{dist_{n,s}})$, where $dist_{n,s}$ is the Euclidean distance between UE u_n and BS S, calculated by $dist_{n,s} = \sqrt{x_n^2 + y_n^2 + (h_n - h_s)^2}$.

B. Problem Formulation

At the BS, the goal is to learn a statistical model with the dataset distributed over N UEs under its service, that is, the BS has to obtain an optimal vector $\boldsymbol{\omega}$ to minimize an empirical loss function $L(\boldsymbol{\omega})$ (e.g., $L(\mathbf{x}^T\boldsymbol{\omega}) = \frac{1}{2}\left\|y - \phi(\boldsymbol{\omega}^T\mathbf{x})\right\|$). The local loss function of the u_n that measures the prediction error of its local dataset $\mathcal{D}_n$, $d_n = |\mathcal{D}_n|$, can be defined as

$$L_n(\boldsymbol{\omega}) = \frac{1}{d_n}\sum_{i=1}^{d_n} l(\boldsymbol{\omega}, \mathrm{x}_n^i), \quad \forall n \in \mathcal{N}. \tag{3}$$

where $l(\boldsymbol{\omega}, \mathrm{x}_n^i)$ is an empirical loss function defined by the learning task and quantifies the loss of model at sample x_n^i.

The objective of the considered learning task is to find the optimal model weights $\boldsymbol{\omega}^*$ that minimize the global loss function $L(\boldsymbol{\omega})$ as

$$\mathcal{OP}_1 \quad \min_{\boldsymbol{\omega} \in R} L(\boldsymbol{\omega}) \tag{4}$$

To solve the optimization problem $\mathcal{OP}_1$, two distributed learning approaches, FL and SL, can be used to train the ML model by exploiting the computational capabilities of the UEs in a distributive manner. However, FL has higher requirements on the computational resource at the UEs and SL has higher communication overhead when the dataset is large at the UE. To efficiently obtain the solution to $\mathcal{OP}_1$ with the dataset distributed over the heterogeneous UEs, we propose a novel distributed learning architecture, namely the HSFL algorithm, which keeps the parallel model training mechanism of FL and the model splitting structure of SL.

III. Hybrid Split and Federated Learning

In this section, we propose a novel HSFL algorithm to reap the benefits of both FL and SL. We first introduce its learning mechanism and then propose a wireless HSFL algorithm.

A. HSFL Learning Mechanism

Inspired by [7], we propose a novel HSFL algorithm with the detailed learning mechanism illustrated in Fig. 2. In our proposed HSFL algorithm, we allow a portion of selected UEs to implement the SL method, namely split training, while allowing another portion of the UEs to use the FL method, namely federated training. Here, the local model updates of federated training and split training in the HSFL algorithm are defined as follows.

1) Federated Training: In our proposed HSFL algorithm, we allow a set of UEs $\mathcal{N}_{F_t}$ to train the received entire ML model independently using their own dataset as in FL. Each federated training UE $u_n, n \in \mathcal{N}_{F_t}$ computes its local model updates following the same rule as in FL,

$$\Delta \boldsymbol{\omega}_t^n = \boldsymbol{\omega}_{t+1}^n - \boldsymbol{\omega}_t = -\eta_t \boldsymbol{g}_t^n, \tag{5}$$

where η_t denotes the learning rate, and $\boldsymbol{g}_t^n$ denotes the gradient computed at UE u_n.

2) Split Training: In our proposed HSFL algorithm, another set of UEs $\mathcal{N}_{S_t}$ train the entire ML model collaboratively with the BS using their local dataset as in SL, where the entire ML model is split into two sub-models including UE-side model and BS-side model, so the UEs train the received UE-side model while the BS trains the BS-side model. Each split training UE $u_n, n \in \mathcal{N}_{S_t}$ trains the UE-side model $\boldsymbol{\omega}_t^l$ with their local datasets to the cut layer in parallel, and then sends the activations $\boldsymbol{a}_t^n$ of the cut layer to the BS. The BS is supposed to be super resourceful and can provide fast model training such that it sequentially performs forward propagation to the BS-side model $\boldsymbol{\omega}_t^e$ with the activations $\boldsymbol{a}_t^n, n \in \mathcal{N}_{S_t}$ received from split training UEs to calculate the loss function $L_n(\boldsymbol{\omega}_t^n)$. Then it calculates the gradients of the loss function and performs backpropagation to the cut layer, where its gradients are computed and sent back to split training UEs for their backpropagation and local updating of the UE-side models, respectively. Therefore, the local model updates of each split training UE u_n is given by

$$\begin{aligned}
&\Delta \boldsymbol{\omega}_t^1 = \boldsymbol{\omega}_{t+1}^1 - \boldsymbol{\omega}_t^1 = -\eta_t \boldsymbol{g}_t^1 \\
&\Delta \boldsymbol{\omega}_t^2 = \boldsymbol{\omega}_{t+1}^2 - \boldsymbol{\omega}_t^1 = -\eta_t \boldsymbol{g}_t^1 - \eta_t \boldsymbol{g}_t^2 \\
&\ldots \\
&\Delta \boldsymbol{\omega}_t^{N_S} = \boldsymbol{\omega}_{t+1}^{N_S} - \boldsymbol{\omega}_t^1 = -\eta_t \boldsymbol{g}_t^1 - \eta_t \boldsymbol{g}_t^2 - \ldots - \eta_t \boldsymbol{g}_t^{N_S}.
\end{aligned} \tag{6}$$

where $\boldsymbol{\omega}_t^1 = \boldsymbol{\omega}_t$, and the gradients of each $u_n, \forall n \in \mathcal{N}_{S_t}$ are calculated by $\boldsymbol{g}_t^1 = \nabla L_1(\boldsymbol{\omega}_t), \boldsymbol{g}_t^2 = \nabla L_2(\boldsymbol{\omega}_t - \boldsymbol{g}_t^1)$, **...** ,$\boldsymbol{g}_t^{N_S} = \nabla L_{N_S}(\boldsymbol{\omega}_t - \boldsymbol{g}_t^1 -, \ldots, -\boldsymbol{g}_t^{N_S - 1})$.

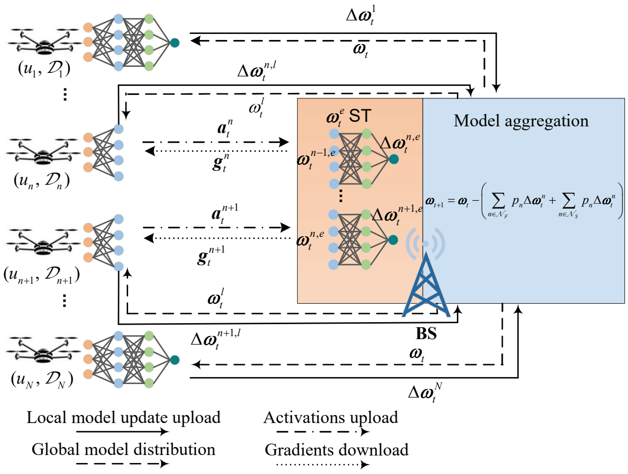


Fig. 2. The learning procedure of wireless HSFL algorithm

3) Model Aggregation of HSFL: Accordingly, the HSFL algorithm adopts the same model aggregation rule as the FedAvg algorithm [4], the global model is updated by performing model aggregation of all the local model updates obtained by federated training UEs and split training UEs.

$$\begin{aligned}
\boldsymbol{\omega}_{t+1} &= \boldsymbol{\omega}_t - \Delta \boldsymbol{\omega}_t = \boldsymbol{\omega}_t - \Big(\sum_{n \in \mathcal{N}_{F_t}} p_n \Delta \boldsymbol{\omega}_t^n + \sum_{n \in \mathcal{N}_{S_t}} p_n \Delta \boldsymbol{\omega}_t^n \Big) \\
&= \boldsymbol{\omega}_t - \Big(\sum_{n \in \mathcal{N}_F} p_n \eta_t \boldsymbol{g}_t^n + \sum_{n \in \mathcal{N}_S} p_n \eta_t \boldsymbol{g}_t^n + \sum_{n=2}^{N_S} p_n \Delta_{\boldsymbol{g}_n} \Big).
\end{aligned} \tag{7}$$

where $\Delta_{\boldsymbol{g}_2} = \eta_t \boldsymbol{g}_t^1, \Delta_{\boldsymbol{g}_3} = \eta_t \boldsymbol{g}_t^1 + \eta_t \boldsymbol{g}_t^2, \ldots, \Delta_{\boldsymbol{g}_{N_S}} = \eta_t \boldsymbol{g}_t^1 + \eta_t \boldsymbol{g}_t^2 +, \ldots, + \eta_t \boldsymbol{g}_{N_S - 1}^t$.

Since split training enables sequential training at the BS, the split training UEs from the second UE $u_n, n \in \{2, \ldots, N_S\}$ will receive more local model updates $\Delta_{\boldsymbol{g}_n}$, which means if having the same number of federated training UEs and split training UEs in one communication round, $N_F = N_S$, split training will provide more local model updates.

B. Wireless HSFL Algorithm

Our proposed HSFL algorithm can potentially be used in wireless UAV networks due to the heterogeneous computation capacity, energy resource and different data distributions over the UEs. The HSFL algorithm has the advantages of both SL and FL, which enables the UEs with weak computation capacity to use split training because it only needs to train a few layers of the ML model locally, while enables the powerful UEs with large dataset to use federated training and train the entire ML model locally. In this section, the detailed learning steps of the wireless HSFL algorithm are presented in **Algorithm 1**. The combination of the steps is defined as one communication round.

IV. UE Selection

In wireless UAV networks, not all the UEs are able to access to the BS and participate in the model aggregation in each communication round due to the limited bandwidth and randomly fading channels. Moreover, the local model updates from different UEs are of dissimilar importance to

Algorithm 1 Wireless HSFL Algorithm

1: Initialize global model $\boldsymbol{\omega}_t$, global UE-side model $\boldsymbol{\omega}_t^l$ and global BS-side model $\boldsymbol{\omega}_t^e$, set $t=0$
2: **Repeat**
3: The BS selects a subset of UEs $\mathcal{K}$, then schedules K_S split training UEs and K_F federated training UEs.
4: The BS distributes $\boldsymbol{\omega}_t$ to federated training UEs $\mathcal{K}_F$ and distributes $\boldsymbol{\omega}_t^l$ to split training UEs $\mathcal{K}_F$.
5: **for** UE $n \in \mathcal{K}$ **do**
6: **if** $n \in \mathcal{K}_F$ **then**
7: UE u_n computes $\Delta\boldsymbol{\omega}_t^n$ as in (5)
8: **else if** $n \in \mathcal{K}_S$ **then**
9: UE u_n computes $\Delta\boldsymbol{\omega}_t^n$ as in (6)
10: **end if**
11: **end for**
12: The BS computes the new global model as in (7)
13: Set $t=t+1$
14: **Until** the desired convergence performance is achieved or the final iteration arrives

the model convergence [12]. Thus, how to choose a subset of UEs for training in each round to achieve the best learning performance using limited resources is necessary but full of challenges. In practice, it is difficult to obtain accurate channel conditions and the information of local model updates before the learning procedure is conducted, and it consumes extra computation and communication resources to estimate each UE's local model updates. Therefore, we propose the dynamic MAB-based UE selection scheme by jointly considering both channel qualities and the significance of local model updates, which selects a subset of UEs according to the estimated information based on the trail-and-error rule.

Knowing that the importance of local model updates $\|\Delta\boldsymbol{\omega}_t^n\|_2$ and the channel quality γ_t^n of UE u_n are non-stationary during communication rounds, we apply the discounted MAB algorithm. By taking into account both channel qualities and the importance of local model updates, we first propose a novel MAB-BC-BN2 UE selection scheme for use in the proposed wireless HSFL algorithm, which also can be extended for use in other wireless distributed learning algorithms (e.g., FL, SFL). Our proposed MAB-BC-BN2 UE selection scheme is based on UCB policy, which achieves efficient UE selection depending on the UCB score, and performs exploration by selecting UEs that are selected less often, and exploitation by selecting the UEs with the largest reward. We view the UEs as the arms in the MAB problem and separately compute discounted cumulative values of the $\|\Delta\boldsymbol{\omega}_t^n\|_2$ and the SNRs, i.e., $\Omega_t^n(\lambda)$ and $\Gamma_t^n(\lambda)$, as the cumulative rewards, and a discounted count of the number of times each UE has been selected, $M_t^n(\lambda)$, till communication round t.

Thus, the discounted UCB score for each UE u_n in communication round t is defined as

$$A_t^n(\lambda) = p_n f(\lambda, n), \tag{8}$$

where p_n is the dataset size ratio of UE u_n, λ denotes the discount factor. $f(\lambda,n)$ is the UCB index function, which is given by

$$f(\lambda,n) = \beta\bar{\Omega}_t^n(\lambda) + (1-\beta)\bar{\Gamma}_t^n(\lambda), \tag{9}$$

In (9), two terms represent the importance of local model updates and the channel quality, respectively. β is the balance factor between them. Here, if $\beta=0$, we can obtain $f(\lambda,n)=\bar{\Gamma}_t^n(\lambda)$, which is formulated the MAB-BC UE selection scheme. If $\beta=0$, we can obtain $f(\lambda,n)=\bar{\Omega}_t^n(\lambda)$, which is formulated the MAB-BN2 UE selection scheme. In (9),

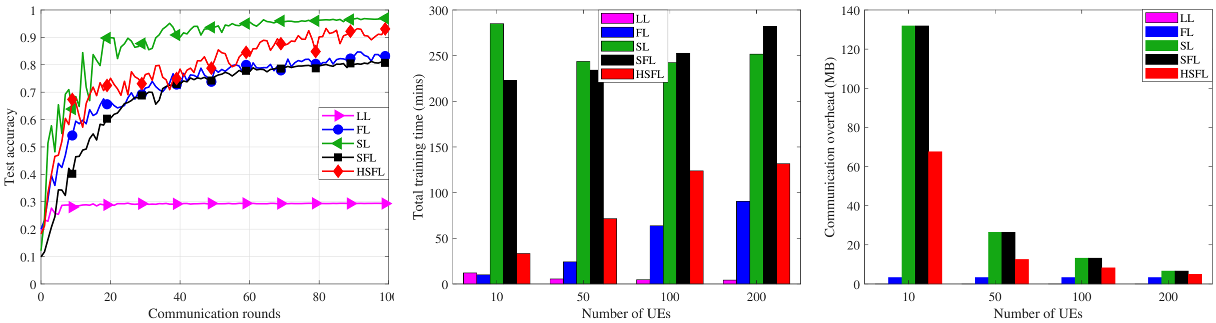
$$\begin{aligned}
\bar{\Omega}_t^n(\lambda) &= \underbrace{\frac{\Omega_t^n(\lambda)}{M_t^n(\lambda)}}_{\text{exploitation}} + \underbrace{\sqrt{2\sigma_t^2\frac{\log(T_t(\lambda))}{M_t^n(\lambda)}}}_{\text{exploration}} \\
\bar{\Gamma}_t^n(\lambda) &= \frac{\Gamma_t^n(\lambda)}{M_t^n(\lambda)} + \sqrt{2\sigma_t^2\frac{\log(T_t(\lambda))}{M_t^n(\lambda)}} \\
\Omega_t^n(\lambda) &= \sum_{\tau=1}^{t}\lambda^{t-\tau}\mathbb{I}_{\{n\in\mathcal{K}_{t-1}\}}\Delta\boldsymbol{\omega}_t^n \\
\Gamma_t^n(\lambda) &= \sum_{\tau=1}^{t}\lambda^{t-\tau}\mathbb{I}_{\{n\in\mathcal{K}_{t-1}\}}\gamma_t^n \\
M_t^n(\lambda) &= \sum_{\tau=1}^{t}\lambda^{t-\tau}\mathbb{I}_{\{n\in\mathcal{K}_{t-1}\}},\ \ T_t(\lambda) = \sum_{\tau=1}^{t}\lambda^{t-\tau}
\end{aligned} \tag{10}$$

Here, the discount factor $0 \le \lambda \le 1$ influences the significance of stale values, $\lambda=1$ means all the past rewards contribute equally to the calculation of $\Omega_t^n(\lambda)$ and $\Gamma_t^n(\lambda)$, and $\lambda=0$ indicates that only the latest reward is used to estimate the value. Thus, $0<\lambda<1$ means putting less weight of stale rewards to calculate the $\Omega_t^n(\lambda)$ and $\Gamma_t^n(\lambda)$. This can avoid the noise in the latest evaluation and the discounted stale rewards computed in the past while computing the estimated values. In the exploration term $\sqrt{2\sigma_t^2\frac{\log(T_t(\lambda))}{M_t^n(\lambda)}}$, σ_t is a hyper-parameter controlling the degree of exploration, which is defined as the maximum standard deviation in the reward computed over the latest update of the UEs. If the UE hasn't been selected very often, or not at all, then $M_t^n(\lambda)$ will be small so that the exploration term will be large, making this UE more likely to be selected. As time progresses, the exploration term gradually decreases (due to $(logn)/n$ goes to zero as n goes to infinity) until eventually UEs are selected based only on the exploitation term. Therefore, we propose a MAB-BC-BN2 UE selection algorithm as in **Algorithm 2**.

V. Experiments and Numerical Results

For the simulations, we conduct the experiments of image recognition on MINST dataset at the laptop with one NVIDIA RTX 2070 GPU and Intel i7-10750H CPUs, where the BS programming is running on the GPU while the UE's programming is running on the CPU. We consider a CNN model with two convolution layers and two fully connected layers, which uses two 5×5 sized kernels in its layers. In SL, SFL, and HSFL algorithms, the DNN model is divided by the second layer.

We simulate a wireless UAV network with one BS located at the origin of the cell and multiple UAVs uniformly distributed within the cell. The cell radius is 500 m, the height of the BS antenna is 20 m and the UAV's flying height is in



(a) Learning accuracy (b) Total training time (c) Communication overhead

Fig. 3. Learning performance comparisons of our proposed HSFL, LL, FL, SL and SFL algorithms on non-IID data with the BC UE selection scheme in terms of test accuracy, the total training time and the communication overhead, $N = 100, \tau = 5, b = 10$.

Algorithm 2 MAB-BC-BN2 UE Selection Algorithm

Initialization

Input: K, K_S, K_F, β λ, p_n for $n \in N$

Initialization: Randomly select $\mathcal{K}_0$, $\mathcal{K}_{S_0}$ and $\mathcal{K}_{F_0}$; a list $\mathcal{A}$ of length N; $t = 1$

Learning:

- **for** $t \leq T$ **do**
  - **for** $i \in K$ **do**
    - The BS distributes $\boldsymbol{\omega}_t$ to $u_n, n \in \mathcal{K}_{F_{t-1}}$ and $\boldsymbol{\omega}_t^l$ to $u_n, n \in \mathcal{K}_{S_{t-1}}$.
    - UEs respectively train the global model with respect to their local dataset.
    - UEs compute the l_2-norm $\|\Delta\boldsymbol{\omega}_t^n\|_2$ of the local model update as (5) and (6), and then upload them to the BS.
  - **end for**
  - The BS receives the local model updates and measures the received SNR γ_t^n of each UE.
  - The BS calculates the UCB score $A_t^n(\lambda)$ and updates list $\mathcal{A}[n] = A_t^n(\lambda)$.
  - The BS generates a UE set $\mathcal{K}_t$ including K clients, and schedules the UE set $\mathcal{K}_{S_t}$ and $\mathcal{K}_{F_t}$
  - Update the elements in $\mathcal{A}$ by $\mathcal{A} = \lambda\mathcal{A}$
- **end for**
- **Return** selected UE set $\mathcal{K}_t$, $\mathcal{K}_{S_t}$ and $\mathcal{K}_{F_t}$.

TABLE I
SIMULATION PARAMETERS OF WIRELESS UAV NETWORKS

Parameters	Value
φ_l, φ_n	$21, 1$
a, b	$5.0188, 0.3511$
Rician factor Kdb	2 dB
system carrier frequency	2 GHz
Noise power σ^2	-130 dBm
P_s, P_n	40 dBm, 23 dBm
B_s, B_w	5 MHz, 1 MHz

the range of $(20-80)$ m. The detailed simulation parameters of the UAV networks are provided in Table I.

A. Learning Performance Comparisons

We adopt BC UE selection scheme to select $K = 10$ UEs from $N = 100$ UEs for model aggregation in each round, and set $K_S = K_F = 5$ for the HSFL algorithm. The non-IID data distribution follows the setting in [4], and we set the local training rounds $\tau = 5$ and the batch size $b = 10$.

1) Learning Accuracy Performance: From Fig. 3 (a), we can observe that the HSFL algorithm provides similar test accuracy performance as SL (sequential SL can be viewed as optimal as the centralized learning) and better test accuracy performance than LL, FL and SFL under non-IID data. This is because in HSFL algorithm, half of UEs perform split training which brings the superiority in test accuracy performance.

2) Training Time and Communication Overhead: We consider four scenarios with the number of UEs $N = 10, 50, 100, 200$, and select $K = 1, 5, 10, 20$ UEs in each scenario, respectively, in each round for model aggregation on non-IID data. In HSFL, the number of split training UEs in each scenario is set as $K_S = 0/1, 2, 5, 10$. We set $B_w = 1$ MHz, which is shared by the selected UEs in each round. SL adopts sequential training, where only one UE takes up the whole bandwidth in each round in the considered scenarios. In contrast, FL, SFL and HSFL are performing local training in parallel so that all the selected UEs in each round share the whole bandwidth.

Fig. 3 (b) shows the total training time over UEs $N = 10, 50, 100, 200$ four scenarios. The training time of FL increases with the increasing number of UEs because the bandwidth allocated to each UE is decreasing. The SL and SFL experience similar training time since the total bandwidth is fixed and the training time mainly depends on the communication latency. Compared to SL and SFL, HSFL uses less training time because here only half of the selected UEs are sharing the total bandwidth while performing split training, which reduces the communication latency for each round.

Fig. 3 (c) plots the communication overhead performance of each UE per communication round. The FL has the same communication overhead over UEs $N = 10, 50, 100, 200$ scenarios because it only depends on the size of the model parameters. The communication overhead of SL, SFL and HSFL decreases with increasing the number of UEs. This is because each UE needs to upload the activations and download the gradients of the cut layer while using SL, in this case, the communication overhead depends on the size of the local dataset at each UE. Especially, HSFL has almost half less communication overhead than SL and SFL in

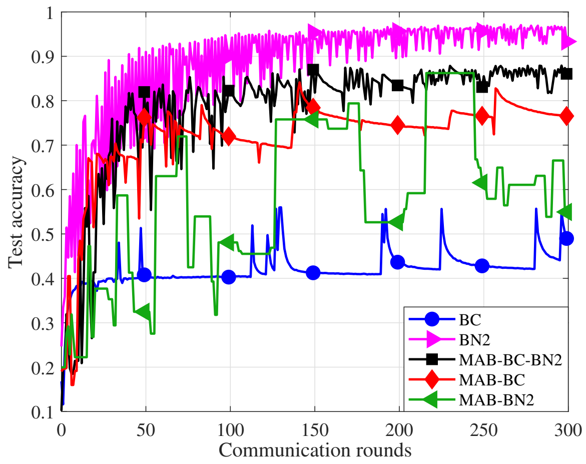


Fig. 4. Test accuracy comparisons of different UE selection schemes within the proposed HSFL framework on non-IID data, $N = 30, K = 3, K_S = 2, \tau = 5, b = 64$, $\lambda = 0.99, \beta = 0.9$

each scenario since it only includes half of the UEs for split training and the other half of the UEs performing federated training.

B. Performance Comparisons of UE Selection Schemes

We set $N = 30$, $\tau = 5$ and $b = 64$. The non-IID dataset is also set as in [4], where the dataset is first sorted by digit label, and it is divided into 60 shards of size 1000, and then each of 30 UEs is assigned with 2 shards. In Fig. 4, it is demonstrated that the BN2 scheme achieves the highest test accuracy because it can obtain perfect information of $\|\Delta\boldsymbol{\omega}_t^n\|_2$ through pre-estimation step. In contrast, the BC scheme has the worst learning performance since it always selects UEs with the best channel qualities and completely neglects the importance of their local model updates. The MAB-BC-BN2 scheme has a similar test accuracy performance as the BN2 scheme, which jointly considers both channel conditions and the importance of local model updates. The MAB-BN2 scheme and the MAB-BC scheme have lower test accuracy than the MAB-BC-BN2 scheme. This is because, in the MAB-BN2 scheme, the selected UEs with large local model updates may fail to upload the local model updates due to bad channel conditions. In the MAB-BC scheme, the selected UEs with good channel conditions may have low local model updates. Noted that the MAB-BC scheme has better performance than the BC scheme because the MAB-BC scheme adopts exploitation-exploration rule and it explores the UEs with less optimal channel conditions. It increases the chance to include the UEs with larger local model updates for model aggregation.

VI. Conclusion

In this paper, we proposed a novel distributed learning architecture, namely hybrid split and federated learning (HSFL) algorithm, and compared its performance with the state-of-art distributed learning algorithms, including federated learning (FL) and split learning (SL) in wireless UAV networks. Our results demonstrated the HSFL algorithm achieved higher learning accuracy than FL and less communication overhead than SL under non-IID data. When implementing wireless HSFL algorithm with limited bandwidth, we developed a Multi-Arm Bandit (MAB) based best channel (BC) and best 2-norm (BN2) (MAB-BC-BN2) UE selection scheme based on the discounted MAB algorithm to select the UEs with larger local model updates and better channel qualities for model aggregation in each communication round. Our results have shown that the MAB-BC-BN2 scheme achieved higher learning accuracy compared to the BC, MAB-BC and MAB-BN2 schemes.

Acknowledgements

This work has received funding from the European Union's Horizon 2020 research and innovation programme, and Primo-5G: virtual presence in moving objects, e.g. drones, through 5G under grant agreement No. 815191.

References

[1] U. Challita, W. Saad, and et al, "Interference management for cellular-connected UAVs: A deep reinforcement learning approach," *IEEE Trans. Wireless Commun.*, vol. 18, no. 4, pp. 2125–2140, Mar. 2019.

[2] F. Tang, Y. Kawamoto, N. Kato, and J. Liu, "Future intelligent and secure vehicular network toward 6G: Machine-learning approaches," *Proceedings of the IEEE*, vol. 108, no. 2, pp. 292–307, Feb. 2020.

[3] B. Brik, A. Ksentini, and M. Bouaziz, "Federated learning for UAVs-enabled wireless networks: Use cases, challenges, and open problems," *IEEE Access*, vol. 8, pp. 53 841–53 849, Mar. 2020.

[4] B. McMahan, E. Moore, and et al, "Communication-efficient learning of deep networks from decentralized data," in *Artificial Intelligence and Statistics*, Feb. 2017, pp. 1273–1282.

[5] O. Gupta and R. Raskar, "Distributed learning of deep neural network over multiple agents," *Journal of Network and Computer Applications*, vol. 116, pp. 1–8, Oct. 2018.

[6] P. Vepakomma, O. Gupta, and et al, "Split learning for health: Distributed deep learning without sharing raw patient data," *arXiv preprint arXiv:1812.00564*, Dec. 2018.

[7] C. Thapa and et al, "Splitfed: When federated learning meets split learning," *arXiv preprint arXiv:2004.12088*, Aug. 2020.

[8] H. H. Yang, Z. Liu, T. Q. S. Quek, and H. V. Poor, "Scheduling policies for federated learning in wireless networks," *IEEE Trans. Commun.*, vol. 68, no. 1, pp. 317–333, Jan. 2020.

[9] T. Nishio and R. Yonetani, "Client selection for federated learning with heterogeneous resources in mobile edge," in *IEEE Int. Conf. on Commun. (ICC2019)*, Apr. 2019, pp. 1–7.

[10] J. Kang and et al, "Reliable federated learning for mobile networks," *IEEE Wireless Commun.*, vol. 27, no. 2, pp. 72–80, Oct. 2020.

[11] J. Ren, Y. He, and et al, "Scheduling for cellular federated edge learning with importance and channel awareness," *arXiv:2004.00490*, Jan. 2020.

[12] M. M. Amiri, D. Gunduz, and et al, "Convergence of update aware device scheduling for federated learning at the wireless edge," *arXiv:2001.10402*, May 2020.

[13] Y. J. Cho, S. Gupta, and et al, "Bandit-based communication-efficient client selection strategies for federated learning," *arXiv:2012.08009*, Dec. 2020.

[14] N. Yoshida, T. Nishio, M. Morikura, and K. Yamamoto, "MAB-based client selection for federated learning with uncertain resources in mobile networks," *arXiv:2009.13879*, Sep. 2020.

[15] W. Xia and T. Q. S. Q. et al, "Multi-Armed Bandit-Based client scheduling for federated learning," *IEEE Trans. Wireless Commun.*, vol. 19, no. 11, pp. 7108–7123, Nov. 2020.

[16] NVIDIA, "Nvidia jetson," *online: https://www.nvidia.com/pt-br/autonomous-machines/uavs-drones-technology/*, 2020.

[17] J. H. et al, "Elevation dependent shadowing model for mobile communications via high altitude platforms in built-up areas," *IEEE Trans. on Ant. and Prop.*, vol. 56, no. 4, pp. 1078–1084, Apr. 2008.