Optimal Transport for UAV D2D Distributed Learning: Example using Federated Learning

Sherif B. Azmy*, Amr Abutuleb*, Sameh Sorour‡, Nizar Zorba◆, and Hossam S. Hassanein‡
* Department of Electrical and Computer Engineering, Queen's University, Kingston, ON, Canada
‡ School of Computing, Queen's University, Kingston, ON, Canada
◆ Electrical Engineering Department, Qatar University, Doha, Qatar
Email: {sherif.azmy, amr.abutuleb, sameh.sorour, hassanh}@queensu.ca, nizarz@qu.edu.qa

ICC 2021 - IEEE International Conference on Communications | 978-1-7281-7122-7/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICC42927.2021.9500304

***Abstract*—Federated Learning (FL) is a novel distributed learning paradigm in which local learning models are simultaneously trained using the stored data on multiple devices, then ultimately aggregated into a global model. A promising use case of FL is the training of a global model using the data collected by unmanned aerial vehicles (UAVs) during their flight, which is invaluable in scenarios in which an infrastructure cannot be accessed (e.g., disaster). However, this is challenging as limited resources are to be distributed between flight time, sensing, processing, and communication. In this paper, we address the resource problem for a set of heterogeneous UAVs with different computation and communication capabilities from distributed point of view. We propose the usage of Device-to-Device (D2D) communication to fairly distribute the data so-far collected by UAVs with different capabilities by posing it as an optimal transport problem. Our contribution is two-fold: (1) We obtain the fairest distribution of data given the UAVs' computational capabilities such that global learning time is minimal; (2) We devise a scheme using Optimal Transport (OT) to achieve such a fair distribution between UAVs. The performance of the proposed techniques is demonstrated in an FL setting with different UAV topologies with the FL training done using the MNIST dataset.**

***Index Terms*—Distributed Learning; Federated Learning; Optimal Transport; Flying Ad-Hoc Networks; Unmanned Aerial Vehicles; Mobile Edge Computing.**

I. Introduction

Machine Learning (ML) models have evolved rapidly over the last decade. Recently, there has been a trend towards making them more efficient. There have been efforts to allow mobile deployment of ML models in both hardware (e.g., specialized accelerators such as Eyeriss v2 [1]) and software (e.g., ML architectures such as MobileNets, which is aimed at mobile devices [2]). While there are still challenges to achieve optimal efficiency in terms of hardware design and model optimization, learning on mobile nodes, with mediocre capabilities, is no longer impossible.

Unmanned Aerial Vehicles (UAVs) have become an important part of the future of the smart city [3]. With the advent of Flying Ad-Hoc Networks (FANETs) [4], UAVs can potentially cater to a wide range of applications: from civil applications such as monitoring crops, surveying, emergency and disaster response, and even as dynamic communication infrastructure [5] to mitigating traffic loads, military reconnaissance, and edge computing [6].

Edge Computing (EC) is a nascent paradigm to push computation from the core of the network to its edge, making it possible to reduce latency, increase network reliability, and to provide a wide range of edge services, particularly intelligence at the edge [7].

In this paper, we propose an example of deploying a distributed learning scheme, Federated Learning (FL), over UAVs. Federated Learning (FL) is a distributed learning model that exploits the presence of various devices collecting/storing data upon which a machine learning model is trained. FL operates by first training local models on these devices using their collected/stored data, and then uploading these models to a server to aggregate them [8]. FL comes with various advantages: it reduces communication costs as only the model, not the data, needs be to transferred; it can potentially maximize the benefit from computational resources by pushing computation more towards the edge, which makes it suitable for edge intelligence. Moreover, nodes are no longer only dedicated to sensing, they can also perform computation. FL also ensures the nodes' privacy the data is analyzed locally on its source device.

In an FL-UAV setting, an MEC server is orchestrating the FL task. This MEC server can be a centralized, semi-centralized (e.g., a cluster head), or mobile node (e.g., smartphone, vehicle, UAV) [6]. In such a setting, the major cost is the learning time, as it ultimately translates to the consumption of resources (particularly, energy) [9]. In addition, many of learning-based edge applications in IoT environments (including those employing FL) are driven by fast online learning and rapid decision-making that prohibits the delays of relying on infrastructure-based resources [10].

Mobile Edge Computing (MEC) and edge intelligence, UAVs and mobile devices became capable of a more than just task execution and acquisition. These paradigms introduced data processing and analysis - on the fly - to make the best use of the collected data as soon as possible. A wide range of potential applications for learning on-the-fly exists. In particular, search and rescue applications that localize objects using convolutional neural networks [11], and data-driven optimizations of their operational procedures (e.g., power management, flight scheduling, trajectory planning) using FL [12].

In this paper, we address a case in which UAVs are heterogeneous in terms of their capabilities, particularly that which mixes less capable UAVs with more capable UAVs (e.g., older models with newer ones). This heterogenous mixture of UAVs in a mission would impact its performance; for FL, it would impact the time taken to learn over the models. Less capable UAVs may be burdened by data beyond their capacity, thus increasing the learning time significantly. On the other hand, more capable UAVs could be under-utilized. In this paper, we propose an adaptation approach for FL in UAV environments (FL-UAV) that considers the heterogeneity in UAV capabilities, and presents a two-fold solution to address them: the first is to identify the optimal distribution of data among UAVs to minimize the global time of the FL process, and the second is to use Optimal Transport (OT) [13] to minimize the time of device-to-Device (D2D) data sharing to reach this distribution of data among UAVs. In addition, our scheme aims to select the best times for UAVs to exchange their data along their trajectories to minimize the time of the FL.

The structure of the paper is as follows. Section II details the proposed FL-UAV scheme and considered fair data distribution problem. Section III describes the optimal transport and its usage for computing the data to be transmitted over the D2D links. Section IV highlights the simulation results of both steps of the FL-UAV technique. Section V concludes the paper.

II. Federated Learning over UAVs (FL-UAV)

A. FL-UAV System Description

The D2D Federated Learning model [9] includes a ground station acting as an MEC server for a set $\mathcal{U}$ of N_{UAV} UAV nodes in an area defined by $[-R_{xy}, R_{xy}]^2$ in the 2D plane. Each node in $\mathcal{U}$ is equipped with a sensor, a processor capable of training a local machine learning model, and a transceiver. Figure 1 shows a diagram of such a system, where UAVs could exchange data over communication links with heterogeneous qualities. Initially, the MEC server conveys to the UAV nodes the global model that will be locally trained by each of them using their previously or real-time collected data via their sensors. Once this training is done by each UAV, it uploads the trained models to the MEC server. This process involves four major times: the time to transmit the global model from the MEC server to the i^{th} node $\forall\ i \in 1, \ldots, N_{\text{UAV}}$, denoted $t_i^{(\text{I})}$:

$$t_i^{(\text{I})} = \frac{Q}{\left(B \log_2\left(1 + \frac{h_i P_s}{n_0 B}\right)\right)} \quad (1)$$

where Q denotes the size of the machine learning model in bits, B is the bandwidth of the MEC server's uplink transmission, P_s is the transmission power by the MEC server, h_i is the channel gain between the i^{th} node and the MEC server, and n_0 is the noise power spectral density. Clearly, the denominator in (1) is the channel capacity in bits per second between the MEC and the i-th node.

The second time, denoted by $t_i^{(\text{II})}$, is the time for the i^{th} node to update the model locally, which is expressed as:

$$t_i^{(\text{II})} = \frac{LF}{\zeta_i f_i} d_i \quad (2)$$

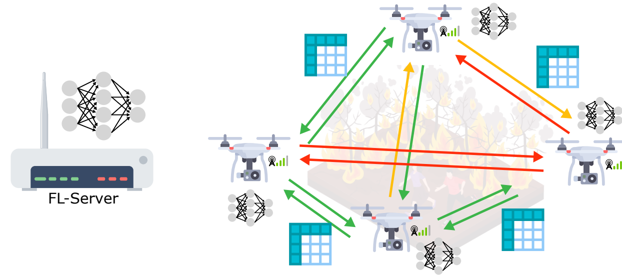

Fig. 1: D2D-Enabled FL-UAV System Diagram

where L is the number of local iterations to be performed for all nodes, F is the number of floating point operations per second (FLOPS) required per a single local iteration, ζ_i is the number of FLOPS per CPU cycle for the i^{th} node, f_i is the CPU frequency for the i^{th} node, and d_i is the number of training data instances of size a bits present in the i^{th} node. The parameters L and F are fixed global parameters set by the MEC server to perform synchronous FL. In asynchronous FL, these parameters could be different for each node.

The third time, denoted by $t_i^{(\text{III})}$, is the time for the i^{th} node to transmit its trained models to the MEC server:

$$t_i^{(\text{III})} = \frac{Q}{b_i \log_2\left(1 + \frac{h_i p_i}{n_0 b_i}\right)} \quad (3)$$

where b_i and p_i are the transmission bandwidth and power of the i^{th} node, respectively.

Finally, the fourth time $t^{(\text{IV})}$ is the time for the MEC server to fuse or aggregate the acquired models. This time is typically negligible for capable stationary MEC servers, but becomes significant for mobile MEC servers in a decentralized FL scheme (e.g., such as a cluster head UAV).

In a D2D-enabled FL scheme, a fifth time, denoted $t_i^{(\text{V})}$ is required for D2D exchange of data between the the i^{th} and the j^{th} nodes, expressed as:

$$t_i^{(\text{V})} = \frac{a d_{ij}}{b_{ij} \log_2\left(1 + \frac{h_{ij} p_{ij}}{n_0 b_{ij}}\right)} \quad (4)$$

where a is the number of bits per data frame (size of each sample), d_{ij} is the number of data instances (or frames) transmitted from node i to node j. Similarly, h_{ij}, b_{ij}, and p_{ij} correspond to the channel gain, transmission power, and bandwidth between the i^{th} and the j^{th} nodes.

The total global learning time can then be described as:

$$t_{\text{global}} = \max_{i \in \mathcal{U}} \left(t_i^{(\text{I})} + t_i^{(\text{II})} + t_i^{(\text{III})} + t^{(\text{IV})}\right) M + \max_{i,j \in \mathcal{U}} t_{ij}^{(\text{V})} \quad (5)$$

where M is the number of global iterations, and the max operator is included as the total time relies on the time taken by the worst-case UAV node. As mentioned above, $t^{(\text{IV})}$ could be assumed zero in case of a dedicated MEC server.

The channel gain is generally defined as $|h|^2 = 10^{\text{PL}(f,r)}$ where $\text{PL}(f, r)$ is the path loss defined as:

$$\text{PL}(f, r)_{\text{dB}} = 20 \log(f) + 10n \log(r) - 147.56 \text{ dB} \quad (6)$$

where f is the carrier frequency, and r is the distance between the transmitter and the receiver. The channel gain for UAVs, in free space (n=2), can then be defined as:

$$\begin{aligned} h(f,d) &= \sqrt{10^{(\mathrm{PL}(f,r)_{\mathrm{dB}}/10)}} \\ &= \sqrt{(10^{20\log(f)+20\log(d)-147.56)/10}} \end{aligned} \tag{7}$$

Eq. 7 is useful for computing the distance in the following subsection under the CARATE UAV trajectory model.

B. 2D CARATE Trajectory Model

In this paper, the mobility model governing the movement of the UAVs is a modified 2D variant based on the 3D CARATE UAV trajectory model [14], where the i^{th} UAV's position $(x^{(i)}, y^{(i)})$'s evolution is decided by:

$$x_t^{(i)} = x_{t-1}^{(i)} + T v_t^{(i)} \cos(\theta_t^{(i)}); \quad y^{(i)} = y_{t-1}^{(i)} + T v_t^{(i)} \sin(\theta_t^{(i)}) \tag{8}$$

where $v_t^{(i)}$ is the velocity at time t, T is the duration of a single step, and $\theta_t^{(i)} \in [0, 2\pi]$ is the yaw. The values $v_t^{(i)}$ and $\theta_t^{(i)}$ are defined as:

$$v_t^{(i)} = v_{\text{seed},t}^{(i)} + V_t^{(i)} \qquad \theta_t^{(i)} = \theta_{\text{seed},t}^{(i)} + \Theta_t^{(i)} \tag{9}$$

where $V_t^{(i)} \sim N(\mu_V, \sigma_V)$ is a normal random variable with mean speed μ_V and variance σ_V. Similarly $\Theta_t^{(i)} \in [0, 2\pi] \sim N(\mu_\Theta, \sigma_\Theta)$. The seed values $v_{\text{seed},t}^{(i)}$ and $\theta_{\text{seed},t}^{(i)}$ are defined as:

$$\begin{aligned} v_{\text{seed},t}^{(i)} &= \text{mean}\left[\left\{v_k^{(i)}\right\}_{k=t-T_{\text{hist}}}^{t-1}\right] \\ \theta_{\text{seed},t}^{(i)} &= \arctan\left(\frac{y_{t-1}^{(i)} - y_{t-T_{\text{hist}}}^{(i)}}{x_{t-1}^{(i)} - x_{t-T_{\text{hist}}}^{(i)}}\right) \end{aligned} \tag{10}$$

where T_{hist} is the number of steps in the past that influence the current step, which has been devised to avoid sharp turns and maintain a smooth trajectory.

However, this model requires initial values for $v_t^{(i)}$ and $\theta_t^{(i)}$. This needs defining the UAVs' positions for the first T_{init} epochs, which could be assumed to deploy from the MEC server's location. This can be done by assuming normal initial positions where $\left\{x_t^{(i)}\right\}_{t=2}^{T_{\text{init}}} \sim N(\text{MEC}_{(x,y)}, R_{xy})$, then the initial velocity components can be computed as:

$$v_{x_t}^{(i)} = \frac{x_t^{(i)} - x_{t-1}^{(i)}}{T} \qquad v_{y_t^{(i)}} = \frac{y_t^{(i)} - y_{t-1}^{(i)}}{T} \tag{11}$$

where $v_{x_t}^{(i)}$ and $v_{y_t}^{(i)}$ are the speed in x and y directions for the i^{th} UAV. The initial angle can then be defined as:

$$\theta_t^{(i)} = \arctan\left(\frac{y_t^{(i)} - y_{t-1}^{(i)}}{x_t^{(i)} - x_{t-1}^{(i)}}\right) = \arctan\left(\frac{v_{y_t}^{(i)}}{v_{x_t}^{(i)}}\right) \tag{12}$$

And the overall velocity is defined as:

$$v_t^{(i)} = v_{x_t}^{(i)} \cos(\theta_t^{(i)}) + v_{y_t}^{(i)} \sin(\theta_t^{(i)}) \tag{13}$$

With this in place, distances can then be calculated between every two points, where the distance, r, between the i^{th} and the j^{th} node can be described by:

$$r_t(i,j) = \sqrt{\left(x_t^{(i)} - x_t^{(j)}\right)^2 + \left(y_t^{(i)} - y_t^{(j)}\right)^2} \tag{14}$$

that is used in Eq. 7.

C. Fair Data Distribution with Minimum Time Per Global Iteration

After the UAV's deployment, they collect data. However, the volumes of data collected by each UAV might not be fair when it comes to the UAV's resource consumption. UAVs consume power at different rates, for example those facing strong wind resistance, lower temperatures, or interference would end up having to consume more power. Moreover, the FL-UAV administrator might have to deploy heterogenous UAVs with non-uniform capabilities to make use of older UAV models in combination with newer ones. As a result, each UAV will have different communication and computational resources.

In this subsection, we formulate an optimization problem that assigns to each node a fair volume of data, such that it can perform its learning time while minimizing the consumption. This is done by minimizing the time required for a single global iteration.

During a single iteration, from the perspective of a single node, the time remaining for each global iteration is $t_i^{(\text{I})} + t_i^{(\text{II})} + t_i^{(\text{III})}$, i.e., the times required to receive the global model, perform the local updates, and to upload the model. To be able to decide the the data volumes included in the D2D time, $t_i^{(\text{V})}$, we formulate the following optimization problem:

$$\begin{aligned} \min_{d_i}\ \max_{\forall i \in \mathcal{U}} \quad & \overbrace{\frac{Q}{\left(B\log_2\left(1+\frac{h_i P_s}{n_0 B}\right)\right)}}^{t_i^{(\text{I})}} + \overbrace{\frac{LF}{\zeta_i f_i} d_i}^{t_i^{(\text{II})}} \\ & + \underbrace{\frac{Q}{b_i \log_2\left(1+\frac{h_i p_i}{n_0 b_i}\right)}}_{t_i^{(\text{III})}} \\ \text{subject to} \quad & \sum_{i=1}^{N_{\text{UAV}}} d_i = D \end{aligned} \tag{15}$$

For a constant $D > 0$, equivalent to the number of data instances to be trained.

The problem in Eq. 15 can be recast as an auxiliary linear program with 1 equality constraint, N_{UAV} inequality constraints, and $N_{\text{UAV}}+1$ non-negativity constraints:

$$\begin{aligned} \min_{\substack{d_i \\ \forall i \in \mathcal{U}}} \quad & \tau \\ \text{subject to} \quad & \sum_{i=1}^{N_{\text{UAV}}} d_i - D = 0 \\ & \tau, \{d_i\}_{i=1}^{N_{\text{UAV}}} \geq 0 \\ & \left\{ \gamma_i d_i - \tau \leq -t_i^{(\text{I})} - t_i^{(\text{III})} \right\}_{i=1}^{N_{\text{UAV}}} \end{aligned} \tag{16}$$

where τ is an auxiliary variable, $\gamma_i = \frac{LF}{\zeta_i f_i}$. This can be interpreted as minimizing the upper bound on times $t_i^{(\text{II})}$ and $t_i^{(\text{III})}$ per global iteration.

This program can be made to minimize other parameters such as b_i, p_i as well, however the optimization problem is not straightforward as the $\log_2$ term becomes asymptotic to zero in the Lagrangian. Such extension is left for a future work.

III. D2D using Optimal Transport

A. Overview of the Optimal Transport

The optimal transport aims to find an optimal mapping, or transport plan, between two probability distributions, $\boldsymbol{\alpha}, \boldsymbol{\beta}$ such that the cost associated with such mapping is minimal [13], where $\boldsymbol{\alpha}, \boldsymbol{\beta}$ are discrete distributions:

$$\boldsymbol{\alpha} = \sum_{k=1}^{N_\alpha} \alpha_k \delta_{x_k}, \quad \boldsymbol{\beta} = \sum_{l=1}^{N_\beta} \beta_l \delta_{y_l} \tag{17}$$

where α_k, β_l are weights of impulses δ_{x_k}, δ_{y_l} located at locations x_k, y_l, and $\sum_{k=1}^{N_\alpha} \alpha_k = 1$ and $\sum_{l=1}^{N_\beta} \beta_l = 1$.

A mapping transporting the quantities in $\boldsymbol{\alpha}$ to $\boldsymbol{\beta}$ exists whose cost is defined by a cost function $c(x_k, y_l)$. For the discrete optimal transport, such mapping can be described by means of a weighted adjacency matrix, $\mathbf{C}$ where $C_{kl} = c(x_k, y_l) \in \mathbb{R}^{N_\alpha \times N_\beta}$.

For $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, and $\mathbf{C}$, there exists a minimal cost coupling, or transport plan, in a set of admissible coupling $\mathbf{U}(\boldsymbol{\alpha}, \boldsymbol{\beta})$, where all possible coupling $\mathbf{P} \in \mathbb{R}_+^{N_\alpha \times N_\beta} \subset \mathbf{U}(\boldsymbol{\alpha}, \boldsymbol{\beta})$, such that $\langle \mathbf{C}, \mathbf{P}^\star \rangle$ is minimal. For $\mathbf{P}$, the sum along the columns yields $\boldsymbol{\alpha}$, while the sum along the rows yields $\boldsymbol{\beta}$. P_{kl} describes a portion of α_k to be transported to its match β_l. Under the Monge-Kantorovich (Kantorovich Relaxation) formulation of the optimal transport, α_k can be splitted and distributed among more than one β_l [13]. The optimal transport problem can be described as:

$$\begin{aligned} \text{minimize} \quad & \langle \mathbf{C}, \mathbf{P} \rangle \\ \text{subject to} \quad & \mathbf{P} \in \mathbf{U}(\boldsymbol{\alpha}, \boldsymbol{\beta}) \end{aligned} \tag{18}$$

where $\langle \mathbf{C}, \mathbf{P} \rangle = \sum_{kl} C_{kl} P_{kl}$ is the inner product.

The Kantorovich-relaxed verison of this problem - in which the splitting of masses is permissible - can be cast as a linear program [13], where:

$$\begin{aligned} \text{minimize} \quad & \mathbf{c}^T \mathbf{P} \\ \text{subject to} \quad & \mathbf{p} \in \mathbb{R}^{n_\alpha n_\beta} \\ & \mathbf{A}\mathbf{p} = \left[\begin{smallmatrix} \boldsymbol{\alpha} \\ \boldsymbol{\beta} \end{smallmatrix} \right] \end{aligned} \tag{19}$$

where $\mathbf{c}$ is a flattened formulation of the matrix $\mathbf{C}$, $\mathbf{p}$ is the flattened form of matrix $\mathbf{P}$, and the matrix $\mathbf{A}$ is defined as:

$$\mathbf{A} = \begin{bmatrix} \mathbb{1}_{N_\alpha}^T \otimes \mathbb{I}_{N_\beta} \\ \mathbb{I}_{N_\alpha} \otimes \mathbb{1}_{N_\beta}^T \end{bmatrix} \in \mathbb{R}^{(N_\alpha + N_\beta) \times N_\alpha N_\beta} \tag{20}$$

where $\mathbb{1}_L$ is the indicator vector of length L, and $\mathbb{I}_L$ is the identity matrix of size $L \times L$, and $\otimes$ is the Kronecker-Product.

The constraint $\mathbf{P} \in \mathbf{U}(\boldsymbol{\alpha}, \boldsymbol{\beta}) \leftrightarrow \mathbf{A}\mathbf{p} = \left[\begin{smallmatrix} \boldsymbol{\alpha} \\ \boldsymbol{\beta} \end{smallmatrix} \right]$ ensures that the sum over the columns would yield $\boldsymbol{\alpha}$ and the sum over the rows would yield $\boldsymbol{\beta}$. By limiting the sum over a whole row to a specific set of "legal couplings", the set of admissible couplings $\mathbf{U}(\boldsymbol{\alpha}, \boldsymbol{\beta})$ can be redesigned [15]. This can be done by updating the lower half of the matrix $\boldsymbol{A}$ to have n_α horizontally concatenated $n_\alpha \times n_\beta$ matrices, however the feasibility of the solution, as a result, relies on the choice of $\boldsymbol{\beta}$ and $\mathbf{C}$ [15].

B. Optimal Transport for D2D FL-UAV

In the FL-UAV setting, the UAVs move with time and follow a trajectory. Such trajectory could be predefined as a mission parameter, or could be planned automatically as the set of UAVs proceeds with their tasks. In both cases, the available resources cannot be exactly predicted, especially in the latter case of spontaneous path planning. The optimal transport, as mentioned in the previous subsection, takes in three inputs: a source distribution, a target distribution, and a cost measure.

For the FL-UAV, source distribution, $\boldsymbol{\alpha}$ where $\alpha_k = d_k$, is the data collected by each of the UAVs. The target distribution, $\boldsymbol{\beta}$ where $\beta_l = d_l^\star$ is the data quantity resulting from the optimization problem solved in Eq. 16. The cost of this optimal transport has to be defined reflecting the quality of the D2D links. While Eq. 4 is a good candidate, the volume of data transferred d_{ij} is the objective sought in finding $\mathbf{P}^\star$. Thus, we consider the transmission of only a single data frame as the cost for this optimal transport, where:

$$c(i,j) = \frac{a}{b_{ij} \log_2 \left(1 + \frac{h_{ij} p_{ij}}{n_0 b_{ij}} \right)} \tag{21}$$

However, for $i = j$, the denominator is zero which would lead to degeneracies in solving Eq. 19. To solve this, the resulting infinites in $\mathbf{C}$ need to be replaced by a finite relatively large number (for example, 1000 seconds).

The resulting optimal coupling, $\mathbf{P}^\star$, is an adjacency matrix whose weights are the data transferred from node i to node j, d_{ij}.

Nevertheless, the distribution, the cost matrix, and the optimal coupling are only snapshots at a specific time, i.e., its optimality is local. We propose an extension to the optimal transport to ensure its optimality over an interval of time epochs. We define a discrete time window:

$$\Pi_{\text{ot}}(c_{\text{ot}}, N_{\text{ot}}) = \left\{ t_m \right\}_{m=c_{\text{ot}}-N_{\text{ot}}/2}^{m=c_{\text{ot}}+N_{\text{ot}}/2} \tag{22}$$

where c_{ot} is the center of the time window, and $N_{\text{ot}}+1$ is the length of the time window. $\Pi_{\text{ot}}(c_{\text{ot}}, N_{\text{ot}})$ contains the indices of the discrete time instances for which an optimal transport is performed, resulting in $N_{\text{ot}}+1$ optimal transports in total. The cost is then evaluated for time instance, and the cheapest optimal transport over this window is chosen to perform the FL-UAV learning. This is best described as:

$$t_{\text{opt}} = \arg\min_{t_i} \{\langle \mathbf{C}_i, \mathbf{P}_i^\star \rangle\}_{t_i \in \Pi_{\text{ot}}(c_{\text{ot}}, N_{\text{ot}})} \tag{23}$$

where $\mathbf{C}_i$ and $\mathbf{P}_i^\star$ are the cost and optimal coupling for a transport at $t_i \in \Pi_{\text{ot}}(c_{\text{ot}}, N_{\text{ot}})$. This can be interpreted as the transport at which the time is minimal over the interval $\Pi_{\text{ot}}(c_{\text{ot}}, N_{\text{ot}})$, which could be due to the trajectories of majority approaching each other, or the communication link among the majority of UAVs is best. However, it is possible that the resulting D2D transmission is unneeded, thus the administrator has to verify Eq. 5 before and after the optimal transport, to see whether it is worth performing the D2D data exchange in the first place.

The FL-UAV administrator will need to consider the choice $N_{\text{ot}}+1$ and c_{ot} with the application in mind. As UAVs are performing the task, the data they acquire at a specific time t might not be sufficient for the model to learn properly. The proposed technique is suitable for scenarios in which the FL-UAV scheme is performing an online-learning task, as it can be cater to near real-time applications, as the complexity of solving the linear program in Eq. 19 has a polynomial smoothed complexity. However, Eq. 5 could be modified to introduce another term, t_{ot}, for the computation of the optimal transport algorithm as:

$$t_{\text{global}} = \max_{i\in\mathcal{U}} \left(t_i^{(\text{I})} + t_i^{(\text{II})} + t_i^{(\text{III})} + t^{(\text{IV})}\right) M + \max_{i,j\in\mathcal{U}} t_{ij}^{(\text{V})} + t_{\text{ot}} \tag{24}$$

IV. Simulation and Performance Evaluation

A. Simulation Setup

To test the proposed algorithm, we simulated N_{UAV} UAVs in a $[\pm 50\text{m}]^2$ 2D box. For the machine learning model, we employed a 3-layer neural network with parameters $[784, 300, 124, 60, 10]$ over 60,000 28x28 images from the MNIST dataset [16]. We then solved the optimization problem for a number of UAVs and a number of global iterations $N_{\text{UAV}} \times N = \{10, 20\} \times \{5, 10\}$, to illustrate the impact of solving Eq. 16 on the learning process. For the optimal transport, $N_{\text{UAV}} = 6$ was chosen, and $a = 784$ bits, corresponding to the size of a single sample of the MNIST dataset. The simulation parameters are shown in Table I.

Figure 2 shows the learning accuracy achieved in the FL-UAV system for different setups, with values for global iterations of duration τ, minimized from the optimization problem described in Eq. 16. It can be seen that τ varies with different setups. Indeed, the global time per iteration described in Eq. 5 is directly proportional to the number of local iterations, and the data processed by each UAV decreases with the total number of UAVs. It can be seen that the accuracy

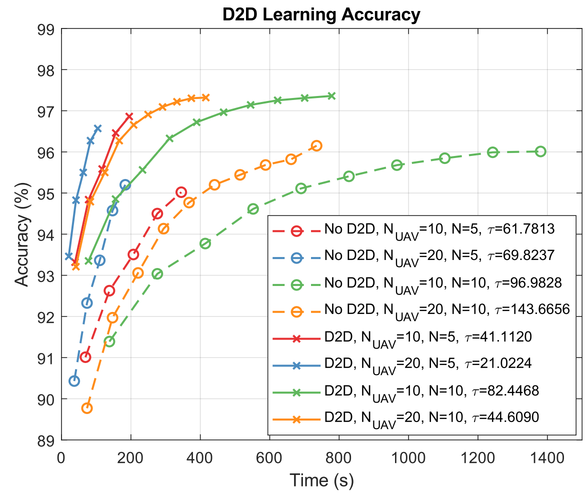

Fig. 2: Learning Accuracy for different N_{UAV} and local iterations N

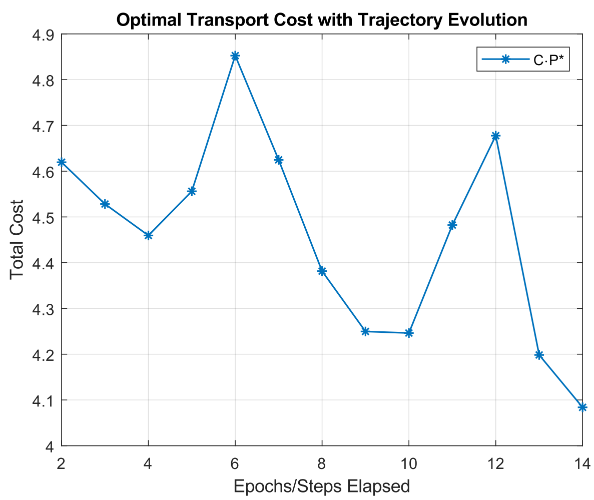

Fig. 3: Optimal transport cost $\langle \mathbf{C}, \mathbf{P}^\star \rangle$ at different times

at $N_{\text{UAV}} = 20$ saturates at less time for $L = 5$ local iterations, while $N_{\text{UAV}} = 10$ took most time to saturate for $L = 10$, thus the choice of local iterations and impacts the performance of the FL-UAV.

Figure 3 illustrates how the optimal transport cost varies with time (and thus with the trajectory). It can be seen that the local minima of the cost occur at $t = \{4, 10, 14\}$, with $t = 14$ being the global minimum. The choice of the window, $\Pi_{\text{ot}}(c_{\text{ot}}, N_{\text{ot}})$, could influence the minimum being observed.

Figure 4 illustrate the edges for the optimal transport at time $t = 10$. The trajectories illustrated represent the history of the UAV's position, whereas at t they are located at the edge of the transport. The values of $\mathbf{P}^\star$ at $t = 10$ where:

$$\mathbf{P}_{t=10}^\star = \begin{bmatrix} 0 & 0 & 10000 & 0 & 0 & 0 \\ 0 & 0 & 3326 & 0 & 0 & 6674 \\ 6121 & 2248 & 0 & 0 & 0 & 1631 \\ 0 & 0 & 0 & 0 & 10000 & 0 \\ 0 & 0 & 0 & 5873 & 0 & 4127 \\ 0 & 9451 & 0 & 0 & 549 & 0 \end{bmatrix} \tag{25}$$

Parameter	Value	Description
T	1	Epoch duration
T_{init}	3	Initial time steps
T_{hist}	2	Past steps
L_{xy}	50 meters	Box size
μ_V	1.8 m/s	Mean velocity
σ_V	0.25 m/s	Spread in velocity
μ_θ	0.35π	Mean angle (UAVs circling)
σ_θ	0.05π	Spread in angle
f_c	2.4 GHz	Carrier Frequency
n_0	1 nW/Hz (-60 dBm)	Noise Power
F	6 megaFLOPS	FLOPS per iteration
Q	8,974,080 bits	Model size
f_i	$N(700 \text{ MHz}, 50 \text{ MHz})$	CPU Frequency for node i
ζ_i	$\Pi(4, 8)$	FLOPS per cycle for node i
b_i	5 MHz	UAV↔Server Bandwidth
p_i	200 mWatt	UAV→Server Power
P_s	1 Watt	UAV←Server Power
b_{ij}	5 MHz	D2D Bandwidth
p_{ij}	200 mWatt	D2D Power
a	784 bits	Size of a single data frame
c_{ot}	4	Center of OT window
N_{ot}	4	Length of OT window

TABLE I: Simulation Parameters

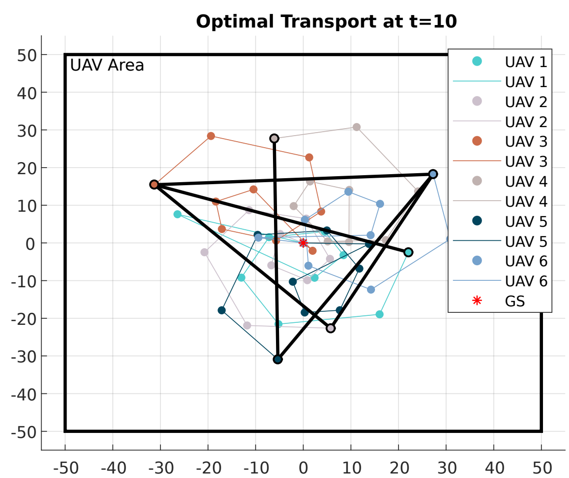

Fig. 4: Optimal Transport at $t = 10$ for $N_{\text{UAV}} = 6$

Where the entries correspond to the amount of data transferred between node i and node j, where the target distribution β's data sizes were obtained by solving Eq. 16:

$$\boldsymbol{\beta} = \begin{bmatrix} 6121 & 11699 & 13326 & 5873 & 10549 & 12432 \end{bmatrix} \quad (26)$$

And the source distribution was the uniform distribution:

$$\boldsymbol{\alpha} = \begin{bmatrix} 10000 & 10000 & 10000 & 10000 & 10000 & 10000 \end{bmatrix} \quad (27)$$

V. Conclusions

We proposed the FL-UAV scheme which finds the optimal distribution of data amounts given a set of UAVs' available computational and communication resources. We use the discrete optimal transport over the UAVs' trajectories to identify an optimum point, at which D2D communication is cheapest to exchange the distribution of data into the aforementioned optimal distribution. Both techniques are low complexity and are easily scalable, which makes them a viable candidate for near real-time learning. Nevertheless, the notions conceived in this paper are not limited to only optimizing the data amounts, but they can also be extended to optimizing communication parameters such as the bandwidth and the power. Such optimalities are of significance to FL-UAV administrators and designers, as the increased efficiency in performing the learning tasks have significant consequences, such as performing a search and a rescue with faster convergence time.

Acknowledgment

This research is supported by a grant from the Natural Sciences and Engineering Research Council of Canada (NSERC) under grant number ALLRP 549919-20.

References

[1] Y.-H. Chen, T.-J. Yang *et al.*, "Eyeriss v2: A flexible accelerator for emerging deep neural networks on mobile devices," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 9, no. 2, pp. 292–308, 2019.
[2] A. G. Howard, M. Zhu *et al.*, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.
[3] F. Mohammed, A. Idries *et al.*, "UAVs for Smart Cities: Opportunities and Challenges," *IEEE ICUAS*, pp. 267–273, 2014.
[4] O. S. Oubbati, M. Atiquzzaman *et al.*, "Routing in Flying Ad Hoc Networks: Survey, Constraints, and Future Challenge Perspectives," *IEEE Access*, vol. 7, pp. 81 057–81 105, 2019.
[5] Y. Kawamoto, H. Nishiyama *et al.*, "Toward Future Unmanned Aerial Vehicle Networks; Architecture, Resource Allocation and Field Experiments," *IEEE Wireless Communications*, vol. 26, pp. 94–99, 2019.
[6] F. Zhou, R. Q. Hu *et al.*, "Mobile Edge Computing in Unmanned Aerial Vehicle Networks," *IEEE Wireless Communications*, vol. 27, pp. 140–146, 2020.
[7] S. Sorour, U. Mohammad *et al.*, "Returning the Favor: What Wireless Networking Can Offer to AI and Edge Learning," *arXiv*, 2020.
[8] P. Kairouz, H. B. McMahan *et al.*, "Advances and Open Problems in Federated Learning," *arXiv*, 2019.
[9] X. Cai, X. Mo *et al.*, "D2D-Enabled Data Sharing for Distributed Machine Learning at Wireless Network Edge," *arXiv*, 2020.
[10] M. Mohammadi, A. Al-Fuqaha *et al.*, "Deep Learning for IoT Big Data and Streaming Analytics: A Survey," *IEEE Communications Surveys and Tutorials*, vol. 20, no. 4, pp. 2923 – 2960, 2018.
[11] Y. Lu, Z. Wang *et al.*, "Target Localization with Drones using Mobile CNNs," *IEEE/RSJ IROS*, pp. 2566–2573, 2018.
[12] T. Zeng, O. Semiari *et al.*, "Federated Learning in the Sky: Joint Power Allocation and Scheduling with UAV Swarms," *arXiv*, 2020.
[13] G. Peyré and M. Cuturi, "Computational Optimal Transport," *arXiv*, 2018.
[14] A. Papaiz and A. M. Tonello, "Azimuth and elevation dynamic tracking of uavs via 3-axial ULA and particle filtering," *International Journal of Aerospace Engineering*, vol. 2016, 2016.
[15] S. B. Azmy, N. Zorba, and H. S. Hassanein, "Optimal Transport for Mobile Crowd Sensing Participants," *2019 IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 1–6, 2019.
[16] Y. LeCun, L. Bottou *et al.*, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.