Towards Opportunistic Federated Learning Using Independent Subnetwork Training

Victor Romero II [*§], Tomokazu Matsui [*†], Yuki Matsuda [‡*†], Hirohiko Suwa [*†] and Keiichi Yasumoto [*†]
[*] Nara Institute of Science and Technology, Ikoma, Nara, Japan
[†] RIKEN Center for Advanced Intelligence Project AIP, Chuo, Tokyo 103-0027, Japan
[‡]Okayama University, Okayama, Okayama, Japan
[§]University of the Philippines Tacloban College, Tacloban City, Philippines
Emails: {romero.victor_militante.rs1, m.tomokazu, yukimat, h-suwa, yasumoto}@is.naist.jp

***Abstract*—Enabling federated learning in opportunistic networks unlocks the potential for machine learning in challenging environments like disaster zones and remote regions. However, the divergent models induced by dynamic node encounters, combined with complete parameter overlap in model-homogeneous training lead to catastrophic interference, which disrupts training progress. Furthermore, when whole models must be transmitted, nodes with shorter contact duration are limited from participating in the training process. To address these challenges, we propose a different approach for training neural networks in opportunistic settings that leverages independent subnetworks and sequential training. We partition the original neural network into non-overlapping subnetworks and assign each to a unique node. These subnetworks are then trained and exchanged repeatedly during node encounters, exposing them further to diverse datasets. As a consequence, we achieve parallel and conflict-free progress while minimizing participation costs. Our experiments demonstrate that continuous training and subnetwork accumulation foster the development of a more robust model. Moreover, by utilizing pre-trained backbones as feature extractors, we achieve a test accuracy of 75.06% on MEDIC's disaster damage severity assessment task, demonstrating that the approach can be adopted in resource-constrained and dynamic scenarios in the real world.**

***Index Terms*—opportunistic networks, federated learning, partial training**

I. Introduction

The sustained growth of data produced by sensor-equipped edge devices and growing concerns in data privacy underscore the importance of federated learning as a key enabler of edge intelligence [1]. By 2025, IoT devices are estimated to generate 79.4 zettabytes of data, presenting new opportunities to train machine learning models [2]. However, these data often reside on private devices or are too costly to transport. Federated learning [3], by allowing data to reside where they are generated, not only safeguards potentially sensitive data but also eliminates the bandwidth and storage costs associated with its centralization. Moreover, by spreading computational demand, federated learning lowers entry barriers for participation and leads to a more democratic training process.

Integrating federated learning with Opportunistic Networks (OppNets) [4] unlocks collaborative machine learning in challenging environments where traditional network infrastructure is impractical or unavailable. These environments range from remote regions lacking conventional network access to scenarios arising from natural disasters or other disruptive events. Furthermore, opportunistic networking techniques can be leveraged in urban areas to capitalize on on-device communication resources and alleviate infrastructure network load [5]. By allowing nodes to exchange model updates over device-to-device (D2D) connections, OppNets enable nodes to aggregate knowledge and collectively improve the performance of machine learning models even in the absence of continuous network connectivity [5]–[7]. Ultimately, this opens new avenues for smart computing in diverse fields, such as rural IoT and disaster response.

The unique characteristics of OppNets necessitate adjustments in federated learning's original formulation to accommodate the additional constraints they present. In opportunistic networks, communication windows are transient and last only for the duration of encounters between nodes [4]. Unless assumptions are made about the nodes pausing to facilitate prolonged transmissions, natural encounters are likely to support only conservative data payloads. In addition, the dynamic and often partitioned network topology makes it impractical to deploy a central server, as typically done in traditional federated learning, to orchestrate the training process. Thus, in existing works, nodes perform model aggregation within the context of their time-varying neighborhood [5]–[7]. In the extreme case, a node can be isolated for extended periods, making its model *stale* in comparison to the rest of the nodes in the network. The divergent models induced by the dynamic nature of encounters among nodes, combined with the complete overlap between trained parameters lead to catastrophic interference, which disrupts training progress.

In this study, we devise a different strategy to train neural networks in opportunistic scenarios that addresses the challenges mentioned above. Motivated by recent work in Independent Subnetwork Training [8] and Sequential Federated Learning [9], our approach divides the original neural network into nonoverlapping subnetworks that are uniquely assigned to each participant. These subnetworks are then trained locally and exchanged with other nodes upon encounter. As subnetworks move from one node to another, they are exposed to different local datasets, allowing them to learn further. Since subnetworks do not share neurons, their individual training

This work was in part supported by JSPS KAKENHI JP21H03431.

progress is unaffected by the status of other subnetworks, making their integration into the full model inherently conflict-free. Finally, the smaller size of the subnetworks implies lower bandwidth and shorter contact duration required for model exchange, which encourages higher participation.

Contributions: Our main contributions are as follows:

1) We devise a new strategy for training neural networks in opportunistic settings, utilizing independent subnetworks and sequential training to prevent aggregation conflicts and enhance generalization through the subnetwork's continual exposure to diverse datasets while reducing participation cost;
2) We develop a discrete event simulator for opportunistic federated learning by extending FedML [10]. Through various experiments, we explore our approach's training dynamics and the practical use of pre-trained backbones in transfer learning scenarios;
3) We apply our approach with pre-trained backbones to classify damage severity levels in disaster scene images where node movement is governed by the random waypoint mobility pattern on real-world road networks, demonstrating its applicability in dynamic and resource-constrained settings;
4) We compare our approach with the model-homogeneous scheme and show its robustness to divergent and stale models induced by dynamic and disjoint topologies.

II. Related Work

In this section, we explore the transition from centrally orchestrated to decentralized federated learning, alongside a review of existing research on opportunistic federated learning. We then discuss Independent Subnetwork Training and highlight its interesting properties, particularly its role in enabling federated learning in resource-constrained and dynamic environments. Lastly, we contextualize the significance of our study within this landscape.

A. Decentralization of Federated Learning

Since its original formulation as described in the seminal paper by McMahan et al. [3], federated learning had been subject to multiple modifications to make it more suited to different computing scenarios where the standard assumptions do not apply. One of the main limitations of federated learning's vanilla formulation is the central server, which synchronizes the learning process into training rounds. The server's role in aggregating the local models received from participating nodes is important to model convergence. However, as in any distributed system, the presence of a central entity signifies a potential bottleneck, single point of failure, and scalability issues [2]. In other cases, such as multi-vendor or multi-institutional collaborations, a third-party server may be undesirable due to trust issues [7], [11].

Decentralized federated learning (DFL) [2] was proposed to address the limitations imposed by the central orchestrator. In DFL, participants establish peer-to-peer connections, exchange models with each other, and perform local model averaging. Assuming full connectivity among all nodes, a consistent global model can be derived at each round. However, if the nodes are not fully connected, additional techniques are required to ensure consistent aggregation of the models. Hence, studies such as [12], [13] leverage consensus theory, treating participating nodes and model parameters as agents and states, respectively. The objective is for these agents to reach a common state through an iterative process of exchanging state information and updating local states over successive rounds of communication.

Opportunistic federated learning is a step further into realizing a truly infrastructure-free federated learning system. A subclass of DFL first introduced in [6] considers the training of machine learning models without relying on fixed infrastructure. Instead, model exchanges are supported using on-device wireless networking interfaces such as Bluetooth or WiFi. The limited range of these interfaces, combined with device mobility, results in a topology that is both time-varying and partitioned. Consequently, communication windows are ephemeral, lasting only as long as the encounter between two nodes. These characteristics render the implementation of consensus-based solutions impractical and challenging.

B. Federated Learning in Opportunistic Settings

Several works have explored implementing federated learning in the opportunistic setting [5]–[7], [14]. These approaches assume a model-homogeneous scheme where participants train all parameters of the same model. Concurrent training of the same model in full under opportunistic scenarios inevitably results in model divergence due to variations in node mobility. Hence, current works on opportunistic federated learning abandon the notion of a global model and pursue the training process as an egocentric or personalized one. Lee et al. [6] tackle the problem of training models that are tailor-fitted to each participant's target distribution. In their approach, training occurs within the duration of an encounter. Hence, to ensure the utility of collaboration, participants first consider each other's label distribution and the predicted encounter duration. Tomita et al. [5] leverage federated learning for training a tourism object detection system using only pairwise connection between participants with a limited number of model exchange. A regressor predicts the resulting accuracy of a potential model aggregation between two participants using their current accuracy. Ochiai et al. [7] adapt federated learning in wireless ad-hoc networks. They assume that the participants are able to send and receive their updated models to all nearby nodes, which under short-lived encounters is difficult unless model sizes are constrained. Finally, Suzuki et al. [14] propose a federated learning framework over an extraterrestrial delay-tolerant network. As encounters between participants (space crafts) are likely to be sparse, they assume a centralized approach where a ground station acts as an orchestrator.

C. Independent Subnetwork Training

Recently, Independent Subnetwork Training (IST) [8] was proposed as a distributed scheme for training neural networks.

IST employs a partial training (PT) approach, wherein participants train smaller parts of the original network. However, unlike other PT-based schemes [15], [16], subnetworks in IST are non-overlapping making their training progress independent and unaffected by the status of other subnetworks. In the context of opportunistic federated learning, this eases the requirement for nodes to frequently encounter all other nodes to achieve consistent model aggregation and simplifies the integration of subnetwork updates into the full model since no conflict resolution is required. Also, the smaller subnetworks significantly reduces the compute and communication requirements, resulting in increased participation despite the short-lived encounters between nodes.

D. Position of this Study

Existing solutions to federated learning in the opportunistic setting assume the model-homogeneous case where participants collaborate to train all parameters of the same model. The complete parameter overlap, in conjunction with the time-varying and likely partitioned topology, inevitably leads to model divergence. Furthermore, when participants are isolated for a long time, their model parameters become stale. Aggregating model parameters under such conditions leads to catastrophic interference. In addition, when whole models must be exchanged, nodes with shorter encounter duration will be limited from participating in the training process.

To address these problems, this study leverages independent subnetwork training to allow participants to train non-overlapping subnetworks of the whole model. However, in contrast to its original implementation, and due to the absence of a central server, subnetworks are not retrieved, and the original network is not repartitioned at each round. Instead, subnetworks are exchanged by the participants on encounter, exposing them to diverse local datasets continuously.

III. Methodology

A. Preliminaries

Let $C = \{c_0, c_1, ..., c_{N-1}\}$ denote the set of N participating nodes in the opportunistic federated learning scenario. Each node c_i possesses its own dataset $\mathcal{D}_i$, and the virtually merged dataset is represented as $\mathcal{D} = \bigcup_{i=0}^{N-1} \mathcal{D}_i$. Typically, for any pair of nodes $c_i, c_j \in C$, $i \neq j$, $\mathcal{D}_i \cap \mathcal{D}_j = \emptyset$.

Let θ denote the weights of the "whole" neural network being trained. The objective of the federated learning process is to determine the optimal parameterization of the weights, denoted as θ^*, that performs well on the virtually merged dataset $\mathcal{D}$ by minimizing the loss function $\mathcal{L}(\theta)$:

$$\theta^* = \arg\min_{\theta}\{\mathcal{L}(\theta) = \frac{1}{N}\sum_{i=0}^{N-1} \ell(\theta, \mathcal{D}_i)\} \tag{1}$$

Here, $\ell(\theta, \mathcal{D}_i)$ represents the loss incurred by θ on the local dataset $\mathcal{D}_i$, defined as:

$$\ell(\theta, \mathcal{D}_i) = \frac{1}{|\mathcal{D}_i|}\sum_{(x,y)\in\mathcal{D}_i} \ell(y, f(x;\theta)) \tag{2}$$

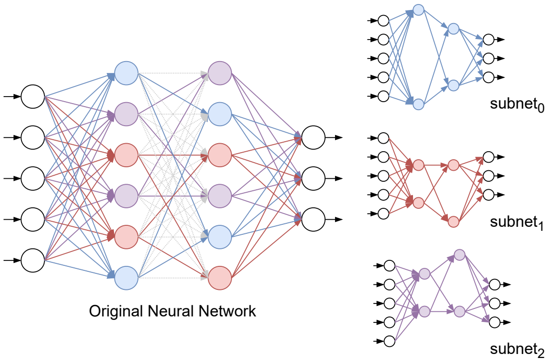


Fig. 1. Extracting multiple subnetworks for $K = 3$; hidden neurons with the same color belong to the same subnetwork. The input and output neurons (white) are common across all subnetworks.

where $f(x;\theta)$ denotes the model's output as parametrized by θ on input sample x.

Under the vanilla federated learning training regime, optimization of the model weights occur through the iterative collaboration of participants as orchestrated by the central server. In the opportunistic setting, collaboration is made possible by encounters that vary from node-to-node as a consequence of their dynamic movement across different geographic locations and evolving interactions with other nodes and the environment. A direct adaptation of vanilla federated learning, in effect, leads to divergent models. Hence, we adopt an alternative methodology that combines independent subnetwork training and sequential federated learning.

B. Generating Independent Subnetworks

Given a neural network with L hidden layers, for $l \in \{1, 2, ..., L\}$, n_l denotes the number of neurons in each of its hidden layers. Similarly, the dimensions of its input and output layers are denoted as n_0 and n_{L+1}, respectively. Before actual training commences, the neural network is first divided into K nonoverlapping subnetworks. This process, shown in Fig. 1, entails iterating through each of its hidden layers and randomly splitting the hidden neurons into K distinct groups. For $k \in \{0, 1, ..., K-1\}$, the hidden neurons of the k^{th} subnetwork is formed by collecting the k^{th} group at each hidden layer. Consequently, the weight associated with the connection between two neurons p and q in the original neural network is activated in a subnetwork if and only if both neurons are contained in its neuron set. This non-overlapping partitioning scheme ensures that all neurons undergo training, while avoiding concurrent updates on the same neuron across different subnetworks. As a consequence, subnetwork updates can be integrated into the original network in a straightforward fashion. All subnetworks incorporate all neurons from the input and output layers of the original network, allowing them to function autonomously as independent models.

C. Sequential Training of Independent Subnetworks

At the onset of the training process, each node is assigned a unique subnetwork and a copy of the original network, which we also refer to as its *full model*. Note that the full model is not exchanged at any time during the training process. It is only used by a node to accumulate trained subnetworks from other nodes that it encounters. By default, we set $K = N$ to ensure unique subnetwork assignments. However, in practical implementation, we allow K to vary to explore alternative scenarios. For example, when $K < N$, complete participation of the nodes in the training process leads to redundant subnetworks, necessitating additional aggregation strategies. Ultimately, when $K = 1$, the federated learning scenario aligns with the model-homogeneous case, where all nodes update all parameters of the original network collectively.

The proposed approach is an attempt to achieve parallel training progress and conflict-free aggregation by combining independent subnetworks with sequential federated learning [9]. For simplicity of exposition, we assume that the training phase is divided into T discrete time steps. At each time step $t \in \{0, 1, ..., T-1\}$ training progress is achieved through each node performing local updates on its assigned subnetwork for E iterations. Let $\hat{\theta}^t_{i,e}$ denote the weights of the subnetwork assigned to node i after $e \in \{0, 1, ..., E\}$ local steps in the t^{th} time step. Using Stochastic Gradient Descent (SGD) as the local optimizer, subnetwork weights are updated as follows:

$$\hat{\theta}^t_{i,e+1} = \hat{\theta}^t_{i,e} - \eta \nabla \ell(\hat{\theta}^t_{i,e}, \mathcal{D}_i) \tag{3}$$

where $\nabla \ell(\hat{\theta}^t_{i,e}, \mathcal{D}_i)$ refers to the gradient of the loss of the current subnetwork parameters on the local dataset $\mathcal{D}_i$ and η is the learning rate. The final updated weights for the current time step $\hat{\theta}^t_{i,E}$ are then integrated into the node's full model.

Nodes utilize opportunistic encounters to exchange models, aiming to advance training of each subnetwork and incorporate additional subnetworks into their full model. Let $\mathcal{N}^t_i$ denote the neighborhood of node c_i at time t. Here, we assume neighbor relationships are symmetric, such that $\forall c_i, c_j \in C : c_j \in \mathcal{N}^t_i \iff c_i \in \mathcal{N}^t_j$. Consider the scenario shown in Fig. 2 where at time $t = 1$, c_2 only has c_1 as its neighbor. After model exchange, at time $t = 2$, c_2 takes custody of c_1's subnetwork, and vice versa, such that $\hat{\theta}^2_{2,0} = \hat{\theta}^1_{1,E}$ and $\hat{\theta}^2_{1,0} = \hat{\theta}^1_{2,E}$. Local training is then conducted using the newly received subnetworks. We refer to such an exchange as a *reassignment exchange*. When $|\mathcal{N}^t_i| = 0$, c_i does not update its subnetwork to prevent overfitting to its local dataset.

In cases where $|\mathcal{N}^t_i| > 1$, c_i may engage in additional model exchanges that do not alter its assigned subnetwork. We refer to these exchanges as *accumulative exchanges*. Although the latter type of exchange enables nodes to incorporate more subnetworks into their full models and improve its performance in the interim, it does not impact the final accuracy that can be achieved by collecting all subnetworks after they have been trained. Therefore, decisions regarding the distribution of subnetworks to all nodes may be deferred until certain training termination criteria, such as maximum training times, are met, at which point existing *store-carry-and-forward* algorithms [17] can be employed to facilitate delivery of subnetworks beyond direct encounters. This final distribution step does not incur additional storage cost, as nodes can simply use their full models as buffer.

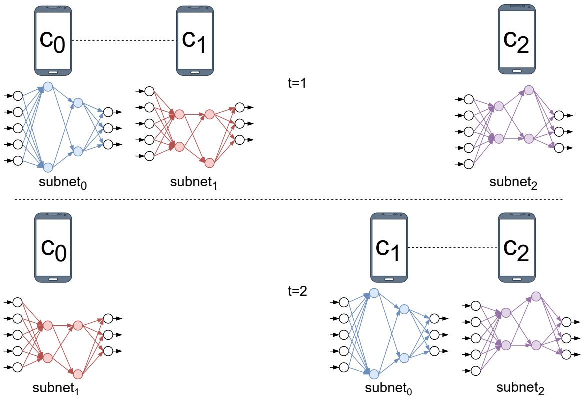


Fig. 2. Subnetwork exchange and sequential training: Nodes train *assigned* subnetworks and exchange them on encounter. Highlight: the movement of subnetwork 0 from c_0 to c_1 at $t = 1$ and from c_1 to c_2 at $t = 2$.

D. Computational Requirements

Given a fully connected neural network, the number of parameters is derived by taking the sum of the products of the widths of its consecutive layers as shown in (4).

$$\sum_{l=1}^{L+1} n_{l-1} n_l \tag{4}$$

In contrast, the number of parameters in each independent subnetwork is computed using (5).

$$\frac{1}{K}(n_0 n_1 + n_L n_{L+1} + \sum_{l=2}^{L} \frac{n_{l-1} n_l}{K}) \tag{5}$$

With the reduced number of parameters, the computational requirements for training the model as well as the bandwidth requirements for exchanging subnetworks among participating nodes are also reduced.

IV. Experiments

We extend the FedML [10] library to develop a discrete event simulator to characterize the performance of the proposed approach. The simulation is organized into discrete time steps during which nodes (1) update assigned subnetworks, (2) exchange subnetworks with neighbors, and (3) update local full models. Additionally, node connections are refreshed each time step to reflect the dynamic nature of opportunistic networks.

Our evaluations focus on the performance of the system under both IID (Independent and Identically Distributed) and Non-IID scenarios. At each time step, we compute the accuracy of each node's local full model against a separate universal test set. In addition, we track the performance evolution of each subnetwork as it is exposed to different local datasets. We also assess the performance of a hypothetical *oracle model*, constructed by integrating the most recent versions of each subnetwork into the original neural network. While

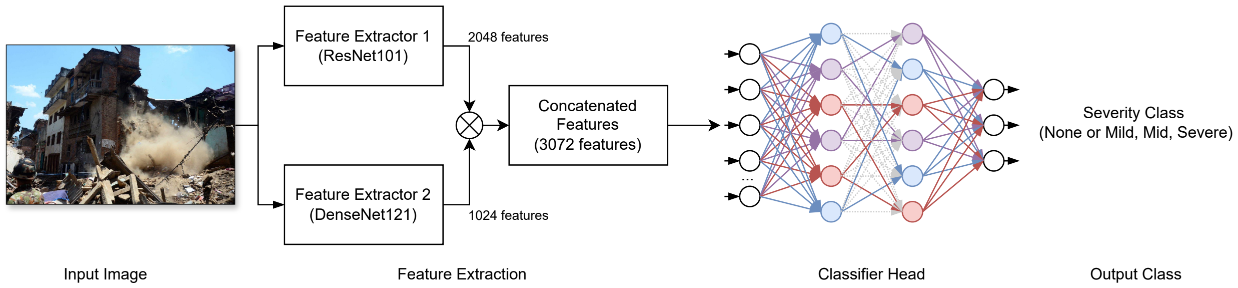


Fig. 3. Using pre-trained backbones as feature extractor.

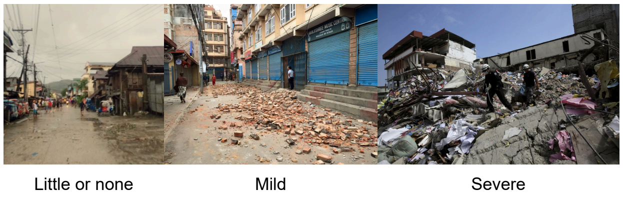


Fig. 4. Images of different damage classes in the MEDIC dataset.

it is unlikely to recover all subnetworks at each time step in the opportunistic setting, the oracle model's performance helps us assess whether improvements in the accuracy of local full models stem from local training of subnetworks or integration of additional subnetworks from encountered nodes.

A. Dataset and Distributions

Three different datasets are used: MNIST, CIFAR-10, and MEDIC. The first two are well-known benchmark datasets used for testing machine learning algorithms. On the other hand, MEDIC [18] consists of $71,198$ images for training machine learning models in disaster response tasks. The images, gathered from social media and the Internet, vary in quality and resolution. For the disaster damage severity task, each image is labeled with one of three classes (*little or none, mild, severe*) as shown in Fig. 4. We use MNIST and CIFAR-10 to gain a general understanding of the behavior of the proposed approach and MEDIC for the more practical scenario of transfer learning, where pre-trained backbones are used as feature extractors in conjunction with the proposed approach to classify damage severity in disaster scene images. The datasets are divided into $N = 100$ nodes in the federation. We use concentration parameter $\alpha = 0.5$ when generating the Non-IID local datasets using Dirichlet sampling.

B. Network Architectures

In our experiments, we used two variants of fully connected networks: 1L5K and 2L5K. Both architectures have 5000 hidden neurons; however, 1L5K has 1 hidden layer, whereas 2L5K has 2 hidden layers. Each hidden layer is followed by a batch normalization layer. Depending on the dataset, the neuron counts of the input and output layers are adjusted; for example, in the case of CIFAR-10, the dimensions of the input and output layers are 3072 and 10, respectively. These models are used to generate $K = 100$ subnets. In the transfer learning case, the model is preceded by pre-trained ResNet101 and DenseNet121 models that extract feature representations of an input image in parallel [19]. The resulting network architecture implemented using Pytorch is shown in Fig. 3. The weights of these pre-trained models are frozen, which allows feature representations for each image to be computed only once. Nodes perform 1 round of training using SGD as optimizer with a batch size of 32 and a learning rate η set to 0.01.

C. Node Connections

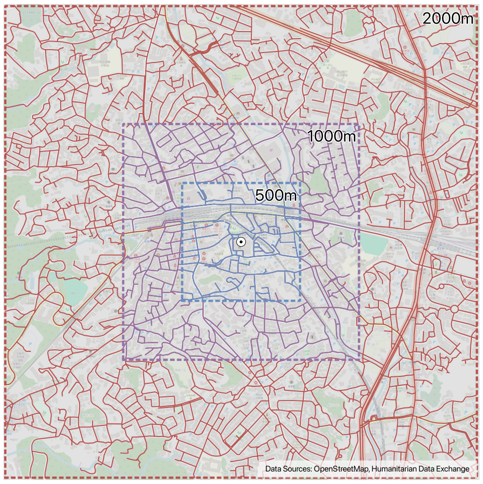
Dynamic connections among nodes are bidirectional and established using two methods: random pairing and the random waypoint model (RWP). In random pairing, the nodes are randomly split into two groups and connections are formed between two nodes chosen from each set. While the resulting topology may not mirror real-world scenarios, it serves as a useful baseline, emphasizing specific properties of the training process (as discussed in Section V). In the RWP, nodes start at random locations, move toward a selected destination within a specified speed range (assumed to be $[0.5, 1.5]$ m/s, corresponding to pedestrian walking speeds), pause upon reaching their destination for a random duration, then resume movement to a new destination. Connection links form when nodes are within each other's communication range (50 meters). RWP connection traces are generated using the ONE Simulator [20] by configuring road networks covering increasing areas ($500m^2$, $1000m^2$, and $2000m^2$) centered on Ikoma City Hall in Nara, Japan as shown in Fig. 5.

D. Experimental Setups

Our experimental design comprises three main groups as summarized in Table I. The initial experiments aim to uncover the fundamental training characteristics of the proposed approach, while the subsequent set investigates the impact of pre-trained backbones, offering insights into its applicability in real-world scenarios where feature extractors are prevalent. In the final phase of experimentation, we apply the proposed approach alongside pre-trained backbones across three different real-world map sizes, comparing its performance with the model-homogeneous approach, which relies on model averaging for aggregation. The sizes of models transmitted are 601.367 KB and 58.68 MB, respectively. Each experiment is replicated three times and executed on an RTX 3080 GPU.

TABLE I
EXPERIMENTAL SETUPS

Setup	Dataset	Architecture	Topology Source	Steps	Distribution	Subnetworks	with Pre-trained Backbone
I	MNIST, CIFAR-10	1L5K, 2L5K	RP	100	IID, NON-IID	100	False
II	CIFAR-10, MEDIC	1L5K	RP	100	IID, NON-IID	100	True
III	MEDIC	1L5K	RWP	500	NON-IID	100, 1	True



Data Sources: OpenStreetMap, Humanitarian Data Exchange

Fig. 5. Portion of Ikoma City, Nara, Japan used for the simulation

V. Results and Discussions

In this section, we present the results of our experiments and discuss their implications. We start by summarizing key metrics across different conditions, and then analyze them to draw important insights.

A. Results of Experimental Setup I

Table II summarizes the results of the first set of experiments on training neural networks from scratch. Fig. 6 shows the evolution of the average test accuracy across all nodes, examining both the IID and Non-IID cases for the MNIST dataset. It reveals a consistent upward trend in accuracy, indicating the progressive refinement of the full model's performance. A notable divergence in accuracy is evident between the IID and Non-IID scenarios, emphasizing the influence of dataset distribution on model learning. The inclusion of an additional layer in the neural network architecture appears to initially impede accuracy gain during the early stages of training. However, as training progresses, the deeper network exceeds the performance of its shallower counterpart.

The accuracy of each node's full model is influenced by the sequential training of the subnetworks and the integration of other subnetworks received through node encounters. To gain deeper insights into the impact of these factors, we analyze the trends in average test accuracy in conjunction with the oracle model accuracy (see Fig. 7). On examination, it is evident that following the initial round of training on the 1L5K model, the oracle accuracy quickly climbs to 84.14% and 61.6% for the IID and Non-IID cases, respectively. This observation indicates that in the case of simpler datasets like MNIST, where subnetworks are able to capture patterns within data quickly, nodes can improve full model performance by incorporating subnetworks from other nodes early.

Finally, we turn our attention to the evolution of an independent subnetwork's accuracy as it transitions from one local dataset to another. Fig. 8 illustrates the test accuracy for a single subnetwork as it is continuously trained on different partitions of the MNIST dataset. We excluded the remaining subnetworks for brevity and because the overarching trend remains consistent. When the local datasets are IID, the subnetwork, despite its limited capacity, exhibits relatively good performance, achieving an accuracy of 71.82% after the first training round, which increases to 90.99% over the course of 100 time steps. On the other hand, in scenarios characterised by Non-IID datasets, the performance (40.88%) of the subnetwork is markedly inferior after the first round of training. This is expected due to the disparity between the sample distributions in each local dataset and that of the virtually merged dataset. Nevertheless, as the subnetwork is exposed to a variety of datasets, its accuracy steadily improves, ultimately reaching a final accuracy of 70.45%. This result suggests that in both the IID and Non-IID scenarios, exposure to diverse local datasets enhances subnetworks' generalization. We observed similar behavior when training on the CIFAR-10 dataset, albeit with lower results.

TABLE II
SUMMARY OF RESULTS FOR SETUP I

Architecture	**ACC**	**MNIST**		**CIFAR-10**	
		IID	**NON-IID**	**IID**	**NON-IID**
2L5K	Oracle	0.9250	0.7780	0.3930	0.3679
	Ave	0.9071	0.6810	0.3695	0.3168
1L5K	Oracle	0.9152	0.7487	0.4090	0.3965
	Ave	0.9050	0.7186	0.3906	0.3661

TABLE III
SUMMARY OF RESULTS FOR SETUP II

Architecture	**ACC**	**CIFAR-10**		**MEDIC**	
		IID	**NON-IID**	**IID**	**NON-IID**
1L5K	Oracle	0.9043	0.9050	0.7506	0.7167
	Ave	0.8996	0.8919	0.7421	0.6948

B. Results of Experimental Setup II

Table III summarizes the result of the second set of experiments where we consider transfer learning scenarios. Given

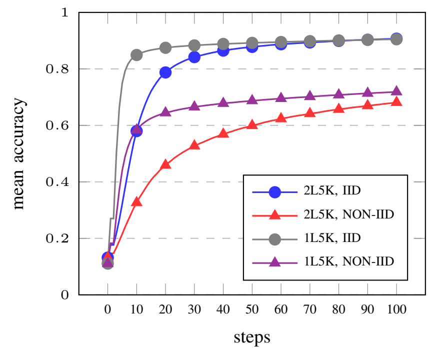


Fig. 6. Mean test accuracy on the MNIST dataset.

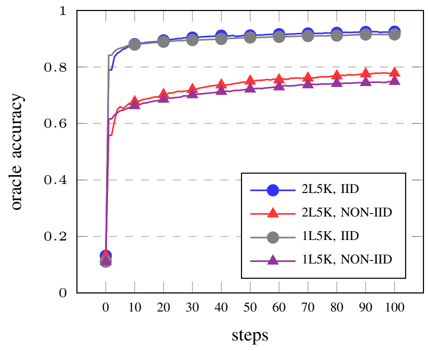


Fig. 7. Oracle test accuracy on the MNIST dataset.

the more complex nature of real-world datasets and the challenging environment our approach is intended for, leveraging pre-trained backbones as feature extractors is preferred over training convolutional layers from scratch. Fig. 9 shows the test accuracy for both the CIFAR-10 and MEDIC datasets. We see the learning task simplifying effect of feature extractors elevating the final accuracy for the CIFAR-10 from 39.05% and 36.60% (see Table II) to 89.96% and 89.19% for the IID and Non-IID cases, respectively. In the MEDIC dataset, the corresponding results are 74.21% and 69.48%, respectively.

Considering the fewer target classes in MEDIC's disaster damage severity classification task, one might question why it exhibits lower accuracy compared to CIFAR-10. However, this difference can be attributed to various factors. Firstly, unlike CIFAR-10, the input images in MEDIC are more complex. Secondly, while *severe damage* may be more readily discernible in images, the distinction between *little or none* and *mild-level damage* presents a significant hurdle. These align with findings in [18], where the authors reported a maximum accuracy of 82.8% in the centralized training setting.

C. Results of Experimental Setup III

Table IV presents the findings from the third and final set of experiments, focusing on the MEDIC damage severity classification task in a more realistic scenario where node connections are governed by the RWP mobility pattern. Fig. 10 and 11 show the corresponding test and oracle accuracy. The

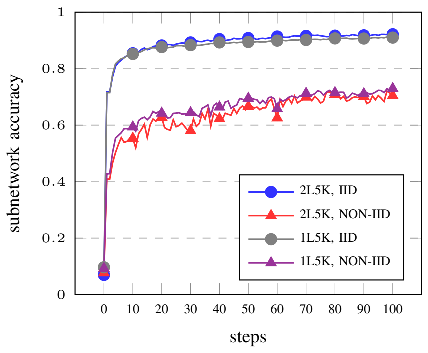


Fig. 8. Test accuracy for subnetwork 0 on the MNIST dataset.

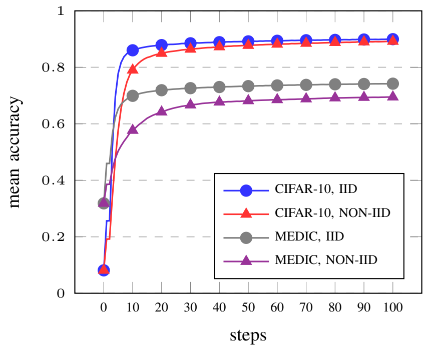


Fig. 9. Mean test accuracy on CIFAR-10 and MEDIC with feature extractors.

result show that the performance of the proposed approach translates into the more realistic setting despite the biasing effect that road network configurations impose on node encounters (compared to open space). In the smallest simulation area, the final test accuracy reaches 68.86%, which is lower than the results we found in the previous experiments where connections are guaranteed at each time step. However, this is supported by a final oracle accuracy of 72.00% indicating that additional accuracy may be achieved by distributing the trained subnetworks beyond directly reachable nodes.

As the simulation area increases, we observe a slower growth in test accuracy. This is expected since an increase in area corresponds to a lower likelihood of node encounters. However, we notice that the oracle accuracy does not suffer as much performance degradation indicating that overall training progress is robust to the clustering of nodes in different areas.

Finally, we contrast the proposed approach with a model-

TABLE IV
SUMMARY OF RESULTS FOR SETUP III

K	ACC	AREA		
		500^2	1000^2	2000^2
100	Oracle	0.7200	0.7244	0.7088
	Ave	0.6886	0.6439	0.5617
1	Oracle	0.6940	0.6556	0.6722
	Ave	0.6356	0.6098	0.5855

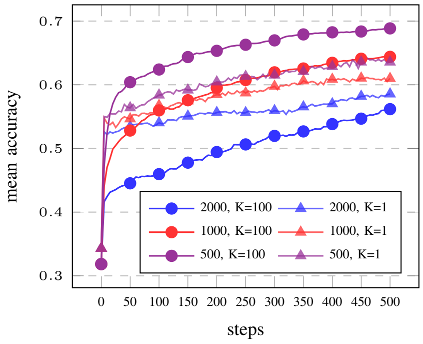


Fig. 10. Mean test accuracy, K=1 vs. K=100 using real-world map

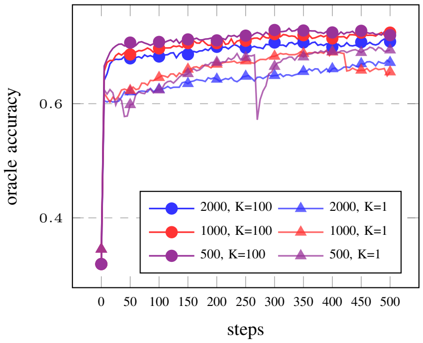


Fig. 11. Oracle test accuracy, K=1 vs. K=100 using real-world map

homogeneous approach by setting the number of subnetworks $K = 1$. Upon examining the results across all simulation areas, we observe that allowing nodes to train all parameters of the neural network initially yields higher test accuracy. However, as time progresses, the partial and independent training-based approach gradually outperforms the former. This can be attributed to model divergence, as shown by the oracle performance depicted in Fig. 11. We observe that at certain times, especially in smaller simulation area, aggregating all models result in a model with significantly reduced accuracy, showing catastrophic interference.

VI. Conclusion

In this study, we proposed a new approach to training neural networks in opportunistic scenarios. By leveraging the conflict-free nature of IST and combining it with SFL, we achieve a training framework that effectively reinstates the notion of a global model without the need for a central server or costly consensus protocols. It is worth noting that its performance in training from scratch on Non-IID data has room for improvement. Looking ahead, we plan to explore the integration of continual learning approaches to enable better retention of previously acquired knowledge from past nodes.

References

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

[2] E. T. M. Beltrán, M. Q. Pérez, P. M. S. Sánchez, S. L. Bernal, G. Bovet, M. G. Pérez, G. M. Pérez, and A. H. Celdrán, "Decentralized federated learning: Fundamentals, state of the art, frameworks, trends, and challenges," *IEEE Communications Surveys & Tutorials*, 2023.

[3] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*. PMLR, 2017, pp. 1273–1282.

[4] S. Trifunovic, S. T. Kouyoumdjieva, B. Distl, L. Pajevic, G. Karlsson, and B. Plattner, "A decade of research in opportunistic networks: challenges, relevance, and future directions," *IEEE Communications Magazine*, vol. 55, no. 1, pp. 168–173, 2017.

[5] S. Tomita, J. P. Talusan, Y. Nakamura, H. Suwa, and K. Yasumoto, "Fedtour: Participatory federated learning of tourism object recognition models with minimal parameter exchanges between user devices," in *2022 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops)*. IEEE, 2022, pp. 667–673.

[6] S. Lee, X. Zheng, J. Hua, H. Vikalo, and C. Julien, "Opportunistic federated learning: An exploration of egocentric collaboration for pervasive computing applications," in *2021 IEEE International Conference on Pervasive Computing and Communications (PerCom)*. IEEE, 2021, pp. 1–8.

[7] H. Ochiai, Y. Sun, Q. Jin, N. Wongwiwatchai, and H. Esaki, "Wireless ad hoc federated learning: A fully distributed cooperative machine learning," *arXiv preprint arXiv:2205.11779*, 2022.

[8] B. Yuan, C. R. Wolfe, C. Dun, Y. Tang, A. Kyrillidis, and C. Jermaine, "Distributed learning of fully connected neural networks using independent subnet training," *Proceedings of the VLDB Endowment*, vol. 15, no. 8, pp. 1581–1590, 2022.

[9] Y. Li and X. Lyu, "Convergence analysis of sequential federated learning on heterogeneous data," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[10] C. He, S. Li, J. So, M. Zhang, H. Wang, X. Wang, P. Vepakomma, A. Singh, H. Qiu, L. Shen, P. Zhao, Y. Kang, Y. Liu, R. Raskar, Q. Yang, M. Annavaram, and S. Avestimehr, "Fedml: A research library and benchmark for federated machine learning," *CoRR*, vol. abs/2007.13518, 2020. [Online]. Available: https://arxiv.org/abs/2007.13518

[11] S. Warnat-Herresthal, H. Schultze, K. L. Shastry, S. Manamohan, S. Mukherjee, V. Garg, R. Sarveswara, K. Händler, P. Pickkers, N. A. Aziz *et al.*, "Swarm learning for decentralized and confidential clinical machine learning," *Nature*, vol. 594, no. 7862, pp. 265–270, 2021.

[12] S. Savazzi, M. Nicoli, and V. Rampa, "Federated learning with cooperating devices: A consensus approach for massive iot networks," *IEEE Internet of Things Journal*, vol. 7, no. 5, pp. 4641–4654, 2020.

[13] A. Giuseppi, S. Manfredi, and A. Pietrabissa, "A weighted average consensus approach for decentralized federated learning," *Machine Intelligence Research*, vol. 19, no. 4, pp. 319–330, 2022.

[14] L. C. Suzuki, V. G. Cerf, J. L. Torgerson, and T. S. Suzuki, "A novel federated computation approach for artificial intelligence applications in delay and disruption tolerant networks," in *2023 IEEE Cognitive Communications for Aerospace Applications Workshop (CCAAW)*. IEEE, 2023, pp. 1–8.

[15] S. Caldas, J. Konečny, H. B. McMahan, and A. Talwalkar, "Expanding the reach of federated learning by reducing client resource requirements," *arXiv preprint arXiv:1812.07210*, 2018.

[16] S. Alam, L. Liu, M. Yan, and M. Zhang, "Fedrolex: Model-heterogeneous federated learning with rolling sub-model extraction," *Advances in Neural Information Processing Systems*, vol. 35, pp. 29 677–29 690, 2022.

[17] J. Gonçalves Filho, A. Patel, B. L. A. Batista, and J. C. Júnior, "A systematic technical survey of dtn and vdtn routing protocols," *Computer Standards & Interfaces*, vol. 48, pp. 139–159, 2016.

[18] F. Alam, T. Alam, M. A. Hasan, A. Hasnat, M. Imran, and F. Ofli, "Medic: a multi-task learning dataset for disaster image classification," *Neural Computing and Applications*, vol. 35, no. 3, pp. 2609–2632, 2023.

[19] E. Hu, Y. Tang, A. Kyrillidis, and C. Jermaine, "Federated learning over images: vertical decompositions and pre-trained backbones are difficult to beat," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 19 385–19 396.

[20] A. Keränen, J. Ott, and T. Kärkkäinen, "The one simulator for dtn protocol evaluation," in *Proceedings of the 2nd international conference on simulation tools and techniques*, 2009, pp. 1–10.