Optimal-Transport-Based One-Shot Federated Learning for Artificial Intelligence of Things

Yi-Han Chiang, *Member, IEEE*, Koudai Terai, Tsung-Wei Chiang, Hai Lin, *Senior Member, IEEE*, Yusheng Ji, *Fellow, IEEE*, and John C. S. Lui *Fellow, IEEE*

***Abstract*—Federated learning (FL) is an emerging distributed machine learning (ML) paradigm in the Artificial Intelligence of Things (AIoT). FL enables AIoT devices to collaboratively train an ML model on the network edge, while protecting data privacy and solving the problem of isolated data islands. Contemporary FL is typically realized through the model aggregation of locally trained models and the model dissemination of a globally averaged model; such a procedure iteratively proceeds until a predefined convergence criterion is met. However, FL necessitates frequent information exchanges between AIoT devices and a parameter server, which inevitably induces tremendous communication costs. Therefore, this article proposes a new design for efficient one-shot FL for AIoT systems, so that the model aggregation and dissemination can be completed within a single communication round. To this end, we leverage optimal transport (OT) theory to design the coupled model averaging (CODE) algorithm to fuse the model weights of the neural networks (NNs) on AIoT devices. The CODE algorithm initially performs OT-based layer-by-layer model averaging (MA) over two NNs to form a fused NN, which will then be averaged with another NN. The CODE algorithm progressively determines a pair of NNs, and continues until all NNs have been examined to achieve one-shot FL. In addition, we provide a detailed convergence analysis for the proposed solution. Our simulation results show that the proposed solution outperforms other one-shot MA mechanisms under various parameter settings.**

***Index Terms*—Artificial Intelligence of Things (AIoT), federated learning (FL), one-shot learning, optimal transport (OT).**

Manuscript received 12 December 2022; revised 7 March 2023 and 2 May 2023; accepted 24 June 2023. Date of publication 7 July 2023; date of current version 8 January 2024. This work was supported in part by JSPS KAKENHI under Grant JP20H00592, Grant JP20K19794, and Grant JP23K16871; in part by the Kayamori Foundation of Informational Science Advancement; in part by the Support Center for Advanced Telecommunications Technology Research, Foundation (SCAT); in part by the Telecommunications Advancement Foundation (TAF); and in part by the RGC's RIF under Grant R4032-18. *(Corresponding author: Yi-Han Chiang.)*

Yi-Han Chiang, Koudai Terai, and Hai Lin are with the Department of Electrical and Electronic Systems Engineering, Osaka Metropolitan University, Osaka 599-8531, Japan (e-mail: chiang@omu.ac.jp; sb22481t@st.omu.ac.jp; lin@omu.ac.jp).

Tsung-Wei Chiang is with MediaTek Inc., Hsinchu, Taiwan (e-mail: d98942029@ntu.edu.tw).

Yusheng Ji is with the Information Systems Architecture Science Research Division, National Institute of Informatics, Tokyo 101-8430, Japan (e-mail: kei@nii.ac.jp).

John C. S. Lui is with the Department of Computer Science Engineering, The Chinese University of Hong Kong, Hong Kong (e-mail: cslui@cse.cuhk.edu.hk).

Digital Object Identifier 10.1109/JIOT.2023.3293230

I. Introduction

MACHINE learning (ML) has demonstrated its great achievements in various emerging applications (e.g., smart healthcare, intelligent transportation, industrial automation, and smart homes/cities), which are greatly revolutionizing people's daily lives. In light of the unprecedented growth of research attention in the past decade, ML is nowadays believed to bring transformative changes across industries, and in practice, many companies have already started using ML solutions due to their potentials to provide more accurate predictions and business decisions.

Both of the model training and inference phases of ML are conventionally carried out atop a cloud server (typically dwelt in a data center) which has full accessibility to the entirety of datasets. In fact, the European Union (EU) has introduced the general data protection regulation (GDPR) [1] to clarify that personal data can only be gathered legally under strict conditions for legitimate purposes, thereby safeguarding the personal data of its citizens. Apparently, traditional cloud-centric frameworks that require all training data to be uploaded from different sources to the cloud inevitably incurs security and privacy issues. Therefore, researchers are motivated to consider shifting model training and inference of ML from the cloud to the network edge.

In contrast to the cloud-centric ML, federated learning (FL) [2], [3], [4] evolves as a next-generation distributed ML technology by allowing the Artificial Intelligence of Things (AIoT) [5], [6], [7], [8] devices to collaboratively train ML models and then conducting inferences with their locally cached data. In the presence of FL, AIoT devices do not have to upload their collected raw data to the cloud, thereby saving energy and bandwidth consumption, reducing response delays, and alleviating privacy and security concerns. In addition, Cisco [9] has revealed that by 2021 the total amount of data driven by IoT will reach 847 zettabytes (ZB), whereas the storage capacities installed in data centers will only grow to 2.6 ZB. Therefore, it is increasingly desirable to let AIoT devices participate in the ML procedures collaboratively without exposing their raw data to the network.

One of the most well-known realization of FL is the weight-averaging FL (often referred to as FedAvg) [10], which is a distributed ML framework where multiple AIoT devices collaborate in solving a distributed ML problem under the coordination of a parameter server (PS). In particular, FedAvg allows AIoT devices to collaboratively train a global model

without sharing their raw data through iteratively performing the following *model aggregation* and *model dissemination* procedures until convergence.

1) For the *model aggregation*, the PS receives the local model updates from AIoT devices over multiple-access channels, and then updates the global model by averaging over the received local model updates.
2) For the *model dissemination*, the PS broadcasts its updated global model to AIoT devices, each of which updates its local model based on its own local dataset.

Since the FL process merely involves the uploading of model updates rather than raw data from AIoT devices during the model aggregation, it is therefore advantageous in the preservation of data privacy, the reduction of network congestion, and the facilitation of distributed on-device computation.

Due to the emergence of 5G, IoT, and AI applications, FL has lately opened up a wide variety of research areas, such as edge computing [11], [12], [13], differential privacy [14], [15], [16], Byzantine robustness [17], [18], [19], nonidentically and independently distributed (NIID) data [20], [21], [22], personalization [23], [24], [25], and compression and quantization [26], [27], [28]. Despite the existing works devoted to FL from various aspects, the requirement of iterative information exchange among clients in FL may result in excessive communication costs (e.g., energy and bandwidth). To resolve such a communication bottleneck, we are thus motivated to design a one-shot FL, namely, a global model, can be produced within a single communication round, thereby reducing communication costs while achieving satisfactory learning performance and data privacy protection for AIoT systems.

Conceptually, paving the way toward constructing an AIoT system exhibits a mathematical analogy with the characterization of the geometry among probability spaces. For example, given two (parent) neural networks (NNs) that were trained differently, how can we align the neurons of the two NNs thereby producing another (child) NN? For this purpose, the optimal transport (OT) theory can be regarded as a viable solution approach due to its strength in aligning probability distributions according to the underlying geometry of the considered metric space (e.g., earth mover distance (EMD) [29] or Wasserstein distance [30]), which has recently raised interests in several emerging research fields (e.g., NN comparison [31], [32], [33], domain adaptation [34], [35], [36], Sinkhorn divergence [37], [38], [39], and wireless communications [40], [41], [42], [43]). Despite the applicability of OT in several AI-related research fields, it remains unclear how to leverage OT to construct AIoT systems. In particular, we aim at designing an AIoT system in which the model aggregation and dissemination can be performed in a one-shot fashion. Meanwhile, each AIoT device only has to upload the model weights of its locally trained model to a model averaging (MA) server without exposing its raw data, thereby achieving data privacy. To this end, we have to address the following two fundamental questions.

1) **Q1:** How to design an AIoT system that completes model aggregation and dissemination in merely one communication round without sacrificing the data privacy of AIoT devices?
2) **Q2:** How to perform MA based on the locally trained models, thereby achieving promising learning performance for AIoT devices?

In this article, we aim at studying how to leverage MA and OT techniques to construct an AIoT system in response to **Q1** and **Q2**. For this, we consider that multiple AIoT devices are connected to a PS, where each AIoT device is equipped with an NN that trains its local model based on its own dataset, and the locally trained models will be aggregated by the PS. The PS first sorts the locally trained models according to their individual test accuracies, and then perform MA over two of them at one time. Initially, the PS averages the model weights of two physical NNs to form a virtual NN, and the virtual NN will be averaged with another physical NN. After performing MA over all physical NNs, the PS broadcasts the globally averaged model to all AIoT devices for the purpose of local inference. To assess the cost-effectiveness of the considered AIoT system, we view the model weights of each locally trained model as probability measures, and the MA is analogous to transporting probability masses from AIoT devices to the PS. In this way, we propose the coupled model averaging (CODE) algorithm to characterize the progressive MA, where the inherent OT problems will be tackled by the ϵ-OT subroutine that yields an ϵ-approximate solution, and ϵ is an OT convergence parameter. In addition, we further provide a convergence analysis for the integrated solution of the CODE algorithm and the ϵ-OT subroutine. Finally, we conduct simulations based on the MNIST dataset under various degrees of NIID-ness to demonstrate the learning performance of our proposed solution. The contributions of this article can be summarized as follows.

1) We design an AIoT system in which locally trained models can be averaged progressively to produce a globally averaged model in a one-shot fashion.
2) We propose the CODE algorithm to perform progressive MA, wherein the ϵ-OT subroutine is invoked to produce ϵ-approximate OT solutions.
3) We analyze the convergence of the integrated solution of the CODE algorithm and the ϵ-OT subroutine.

The remainder of this article is organized as follows. In Section II, we briefly introduce the related works regarding distributed ML and model averaging. In Section III, we present an AIoT system, where a PS is deployed to perform MA for a set of AIoT devices, and then we overview the widely used OT problem formulations, the discretization and regularization techniques. In Section IV, we leverage OT techniques to formulate the problem of model fusion in the considered AIoT systems. In Section V, we propose the ϵ-OT subroutine to fuse the locally trained models layer-by-layer and the inherent OT problems are solved approximately, and present the convergence analysis. In Section VI, we describe our simulations. Finally, this article is concluded in Section VII. The notations used throughout this article are summarized in Table I.

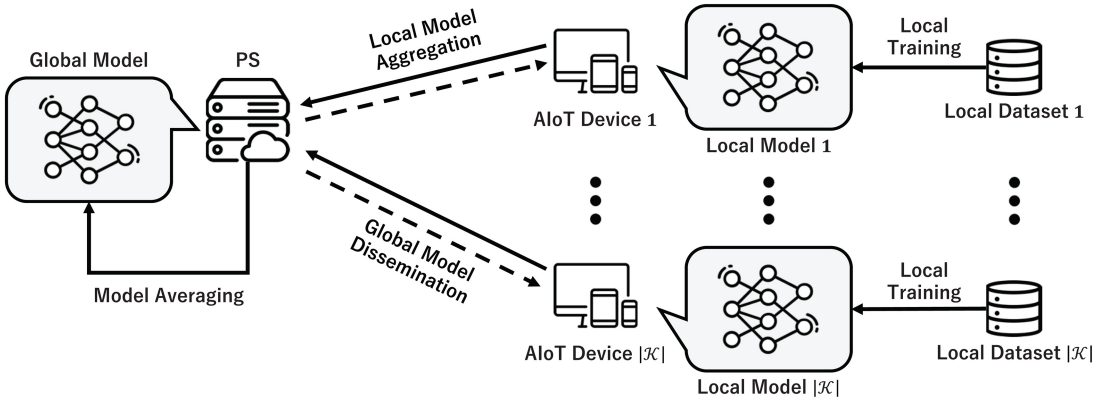


Fig. 1. Illustration of an AIoT system.

TABLE I
SUMMARY OF NOTATIONS

Notation	Meaning
$\mathcal{K}$	the set of AIoT devices
$\mathcal{D}_k$	the local dataset owned by AIoT device k
L	the number of layers of the NN in each AIoT device
$N_{k,l}$	the number of neurons of the l-th layer of AIoT device k
$\mathbf{W}_{k,l}$, $\mathbf{W}_{k,l}^{[i]}$	the column vector of the incoming edge weights of the l-th layer of AIoT device k, the i-th entry of $\mathbf{W}_{k,l}$
$\mathcal{V}(\cdot,\cdot)$	the set of probabilistic couplings (or transport polytopes) between two empirical probability measures
$\mathbf{T}$, $\mathbf{T}^{[i,j]}$	the transport matrix, the (i,j)-th entry of $\mathbf{T}$
$\mathbf{C}$, $\mathbf{C}^{[i,j]}$	the transport cost matrix, the (i,j)-th entry of $\mathbf{C}$
λ, $\Omega(\mathbf{T})$	regularization parameter, the corresponding regularization term in terms of $\mathbf{T}$
p, $\rho_k(\cdot,\mathbf{a}_k)$	the order of Wasserstein distance, the weighting coefficients of Wasserstein distance for the probability measure $\mathbf{a}_k$
$\mathbf{K}$, $\mathbf{K}^{[i,j]}$	the scaled transport matrix, the (i,j)-th entry of $\mathbf{K}$
$\psi(\mathbf{f},\mathbf{g})$	the Lagrangian in terms of the dual variables $\mathbf{f}$ and $\mathbf{g}$
$\mathbf{T}_{k,l}$	the transport matrix of the l-th layer of NN k
$\widehat{\mathbf{W}}_{k,l}$, $\widehat{\mathbf{W}}_{k,l}^{[i]}$	the pre-processed model weights of the l-th layer of NN k, the i-th enry of $\widehat{\mathbf{W}}_{k,l}$
$\widetilde{\mathbf{W}}_{k,l}$, $\widetilde{\mathbf{W}}_{k,l}^{[i]}$	the post-processed model weights of the l-th layer of NN k, the i-th enry of $\widetilde{\mathbf{W}}_{k,l}$
$\overline{\mathbf{W}}_{k,l}$, $\overline{\mathbf{W}}_{k,l}^{[i]}$	the globally averaged model weights of the l-th layer of NN k, the i-th enry of $\overline{\mathbf{W}}_{k,l}$
ϵ	convergence parameter for OT

II. Related Works

A. One-Shot Federated Learning

In the extant literature, the designed one-shot FL frameworks are primarily based on knowledge distillation or dataset distillation techniques. The main idea is to use public datasets or distilled synthetic data from clients to do model distillation on a PS, where the locally trained models play as teachers and the PS acts like a student. Guha et al. [44] leveraged ensemble methods to design one-shot FL under supervised and semi-supervised settings. Shin et al. [45] developed an XOR-based mixup data augmentation method XorMixFL to correct the NIID-ness of data distributions for one-shot FL. Zhou et al. [46] proposed a distilled one-shot FL framework DOSFL, where each client distills its private dataset and sends its synthetic data, instead of transmitting bulky gradients or weights, to a PS. Li et al. [47] proposed a knowledge transfer-based algorithm FedKT for one-shot FL under cross-silo settings. Zhang et al. [48] designed a data-free one-shot FL framework FedSyn that trains a global model through a data generation stage and a model distillation stage. Song et al. [49] proposed FedD3 based on the concept of decentralized dataset distillation, which allows clients to upload distilled data instead of models to a PS in a one-shot manner. Despite the strength of knowledge or dataset distillation revealed from the above works, it is unclear whether and how OT can be used to pave the way toward an efficient one-shot FL.

B. Model Averaging of Neural Networks

MA evolves as a promising solution to fuse multiple NNs into one. Kamp et al. [50] proposed an efficient dynamic averaging protocol for decentralized training of deep NNs (DNNs) from distributed data sources. Yu et al. [51] provided a thorough and rigorous theoretical study on how MA works as well as parallel mini-batch SGD but with greatly reduced communication overhead. Lin et al. [52] proposed a distillation framework for robust federated model fusion, where a central classifier is trained through unlabeled data on the outputs of the models from clients. Wang et al. [53] designed a layer-wise FL algorithm for convolutional NNs (CNNs) and long short-term memories (LSTMs) that appeal to Bayesian nonparametric methods to adapt to data heterogeneity. The above works have coped with the MA of NNs from various aspects, but it is unknown how OT can be leveraged to design one-shot FL. Although [54] dropped hints about applying OT to model fusion, it does not guide us to design CODE across NNs.

III. System Model

A. Network Architecture

We consider an AIoT system consisting of a PS and a set $\mathcal{K}$ of AIoT devices (as shown in Fig. 1). The AIoT devices are connected to the PS through orthogonal multiple access channels. That is, the transmission of an AIoT device to the PS will not be interfered by that of another AIoT device. In

addition, each AIoT device collects data through monitoring and sensing, and the acquired data will be stored in its own dataset $\mathcal{D}_k$. For the sake of data privacy protection, the dataset of each AIoT device will not be revealed to the PS, and no collaboration or information sharing can take place between AIoT devices.

Each AIoT device locally performs DL over an L-layer NN, where the first and the Lth layers represent the input and output layers, respectively. The number of neurons of the lth layer of AIoT device k is denoted as $N_{k,l}$. Based on its own dataset, each AIoT device performs DL to train its local model. To characterize the locally trained model, we denote by $\mathbf{W}_{k,l}$ the column vector of the incoming edge weights of the lth layer of AIoT device k, and $\mathbf{W}_{k,l}^{[i]}$ indicates the ith entry of the column vector, where $1 \leq i \leq N_{k,l-1}N_{k,l}$ and $1 \leq l \leq L$.

In the considered AIoT system, multiple AIoT devices collaborate in solving a distributed ML problem, under the coordination of the PS. The details of the considered FL algorithm can be described as follows.

1) *Local Training:* Each AIoT device performs a local training based on its own dataset. Once the local training completes, the AIoT device will have its trained model, which is specified by the model weights (i.e., the edge weights between two adjacent layers of an NN).
2) *Model Aggregation:* Once the local training of an AIoT device completes, the PS will schedule the AIoT device to upload its trained model weights.
3) *Model Averaging:* Once the PS receives all of the trained models from AIoT devices, it will then perform MA over the model weights of those locally trained models.
4) *Model Dissemination:* The PS broadcasts its globally averaged model, which will be used for local inference (i.e., prediction) on each AIoT device.

Note that the AIoT system only involves the uploading of model updates rather than simply updating raw data by each individual AIoT device during the model aggregation. This offers several distinct advantages, such as the preservation of data privacy, the reduction of network congestion, and distributed on-device computation. Unlike conventional FL requiring multiple communication rounds to converge, the considered AIoT system aims to perform the steps 1–4 in a noniterative way, thereby reducing the consumption of computing and communication resources for AIoT devices.

Remark 1: In the MA in step 3, the PS needs to produce a globally averaged model based on the locally trained models aggregated from AIoT devices. To ensure that the produced globally averaged model works similarly to the underlying data distribution of those locally trained models, we are motivated to leverage OT to help perform the MA in merely one communication round.

Remark 2: Orchestrating FL algorithms with the consideration of channel impairments (e.g., [55], [56], and [57]) can take more communication rounds to reach the same convergence criterion. To address this issue, the co-design of other advanced techniques (e.g., resource allocation and client selection) and/or the migration from "one-shot" to "few-shot" paradigm could be more appropriate.

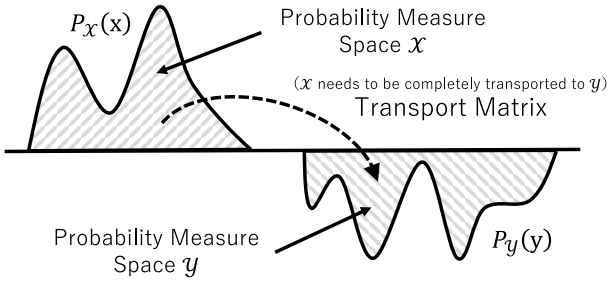


Fig. 2. Illustration of OT.

B. Optimal Transport: Preliminaries

A simple way to understand the concept of OT [30] is to use an analogy of moving a pile of sand. Suppose that we are given a hole and we have to fill it up with a pile of sand, where the pile and the hole have the same volume (see Fig. 2 for a conceptual example). Consider that the mass of the pile is normalized to 1. Then, the pile and the hole can be modeled as probability measures μ and ν, which are defined on some measure spaces $\mathcal{X}$ and $\mathcal{Y}$, respectively. For example, if $\mathcal{X}'$ and $\mathcal{Y}'$ are measurable subsets of $\mathcal{X}$ and $\mathcal{Y}$, $\mu(\mathcal{X}')$ and $\nu(\mathcal{Y}')$ indicate the measures of how much sand is located inside $\mathcal{X}'$ and $\mathcal{Y}'$, respectively. In addition, we model a measurable and nonnegative cost function $c(x, y)$, which quantifies the costs of transporting one unit of mass (i.e., sand) from location x to location y.

1) Discretization: When μ and ν are only accessible through discrete samples, the corresponding empirical probability measures can be written, respectively, as

$$\mathbf{a} = \sum_{i=1}^{m} \mu_i \delta(\mathbf{x}_i), \quad \mathbf{b} = \sum_{j=1}^{n} \nu_j \delta(\mathbf{y}_j) \tag{1}$$

where $\delta(\cdot)$ denotes the Dirac function, μ_i and ν_j are probability masses associated to the ith and jth samples, where the weights μ_i and ν_j live in the probability simplex, namely, $\sum_{i=1}^{m} \mu_i = 1$ and $\sum_{i=1}^{n} \nu_j = 1$, respectively.

Given the empirical probability measures $\mathbf{a}$ and $\mathbf{b}$, we now denote by $\mathcal{V}$ the set of probabilistic couplings (or transport polytopes) as

$$\mathcal{V}(\mathbf{a}, \mathbf{b}) = \left\{\mathbf{T} \in \mathbb{R}_{+}^{m \times n} | \mathbf{T}\mathbf{1}_n = \mathbf{a}, \mathbf{T}^T\mathbf{1}_m = \mathbf{b}\right\} \quad \forall \mathbf{a} \in \mathbb{R}^m, \mathbf{b} \in \mathbb{R}^n \tag{2}$$

where $\mathbf{1}_d$ is a d-dimensional vector of ones, $\mathcal{U}(\mathbf{a}, \mathbf{b})$ contains all nonnegative m-by-n matrices with row and column sums $\mathbf{a}$ and $\mathbf{b}$, respectively. In addition, $\mathcal{V}(\mathbf{a}, \mathbf{b})$ has a probabilistic interpretation: for any two multinomial random variables A and B taking values in $\{1, \ldots, n\}$ and $\{1, \ldots, m\}$, each with the probability distribution $\mathbf{a}$ and $\mathbf{b}$, respectively, the set $\mathcal{V}(\mathbf{a}, \mathbf{b})$ contains all possible joint probabilities of (A, B). Indeed, any $\mathbf{T} \in \mathcal{V}(\mathbf{a}, \mathbf{b})$ can be identified as a joint probability matrix for (A, B) such that $\mathbf{T}^{[i,j]} = \Pr[A = \mathbf{x}_i, B = \mathbf{y}_j]$, where $\mathbf{T}^{[i,j]}$ is the (i, j)th entry of $\mathbf{T}$.

For the discretization, we denote by $\mathbf{C}$ is the cost function matrix, whose (i, j)th entry $\mathbf{C}^{[i,j]}$ is the ground cost of moving the probability mass $\mathbf{x}_i$ to $\mathbf{y}_j$. Given the transport cost matrix $\mathbf{C}$, the cost of mapping $\mathbf{a}$ to $\mathbf{b}$ using a transport matrix (or

joint probability) $\mathbf{T}$ is computed by $\langle \mathbf{T}, \mathbf{C} \rangle$. The Kantorovitch problem between $\mathbf{a}$ and $\mathbf{b}$ given the transport cost matrix $\mathbf{C}$ can then be expressed as

$$\mathcal{P}_0: \quad \min_{\mathbf{T} \in \mathcal{V}(\mathbf{a}, \mathbf{b})} \langle \mathbf{T}, \mathbf{C} \rangle \tag{3}$$

where $\langle \cdot, \cdot \rangle$ stands for the Frobenius dot product, i.e., $\langle \mathbf{A}, \mathbf{B} \rangle \triangleq \text{tr}(\mathbf{A}^T \mathbf{B})$ for any two matrices $\mathbf{A}$ and $\mathbf{B}$ with the same size.

2) Wasserstein Distance and Barycenter: When the ground cost is a metric, the optimal value of OT problems is also a metric. To quantitatively characterize the transport costs, we conventionally define the *Wasserstein distance* of order p between $\mathbf{a}$ and $\mathbf{b}$ as

$$W_p(\mathbf{a}, \mathbf{b}) = \left(\min_{\mathbf{T} \in \mathcal{V}(\mathbf{a}, \mathbf{b})} \langle \mathbf{T}, \mathbf{C} \rangle \right)^{\frac{1}{p}} \tag{4}$$

where $\mathbf{C}^{[i,j]} = (d_{\mathcal{S}}(\mathbf{x}_i, \mathbf{y}_j))^p$ denotes a metric measuring the distance between $\mathbf{x}_i$ and $\mathbf{y}_j$, and $d_{\mathcal{S}}(\cdot)$ represents distance over the space $\mathcal{S}$. Note that the Wasserstein distance is also known as the EMD [58] in computer vision communities, and it defines a metric over the space of integrable squared probability measures.

In literature, it is typical to look for the "mean" or "barycenter" of several data points. The notion of barycenters can be extended to the use of the Wasserstein distance. Formally, we have the barycenter of multiple probability measures

$$\mathbf{a}^* = \arg \min_{\mathbf{a} \in \mathbb{R}^m} \sum_{k \in \mathcal{K}} \rho_k W_p(\mathbf{a}, \mathbf{a}_k), \quad \sum_{k=1}^{K} \rho_k = 1 \tag{5}$$

where $\mathbf{a}$ lies in the probability simplex composed of probability vectors in $\mathbb{R}^m$. Although (5) is in essence an LP, its scale forbids the use of generic solvers for medium-scale problems.

3) Regularization: In practice, pursuing an optimal solution to OT problems is computationally prohibitive. To address this issue, it is typical to introduce an entropic regularization penalty to the original OT problems. In particular, let us denote by $\lambda > 0$ a regularization parameter and by $\Omega(\mathbf{T})$ the corresponding regularization term given a transport matrix $\mathbf{T}$. Then, we have the discrete regularized OT problem as

$$O_\lambda(\mathbf{a}, \mathbf{b}) \coloneqq \min_{\mathbf{T} \in \mathcal{V}(\mathbf{a}, \mathbf{b})} \langle \mathbf{T}, \mathbf{C} \rangle - \lambda \Omega(\mathbf{T}) \tag{6}$$

where the regularization term [37] is defined as

$$\begin{aligned} \Omega(\mathbf{T}) &= -\langle \mathbf{T}, \log(\mathbf{T}) - \mathbf{1}_{m \times n} \rangle \\ &= -\sum_{i=1}^{m} \sum_{j=1}^{n} \mathbf{T}^{[i,j]} \left(\log\left(\mathbf{T}^{[i,j]} \right) - 1 \right) \end{aligned} \tag{7}$$

where $\log(\mathbf{T})$ represents the entry-wise logarithms of $\mathbf{T}$.

IV. Problem Formulation

In the following, we leverage discrete OT techniques for the MA in the considered AIoT system. In particular, we formulate the transport cost minimization problem $\mathcal{P}_1$ and show that its optimal solution has a closed-form expression. Then, we present the equivalent minimization problem $\mathcal{P}_2$ that preserves the optimality of $\mathcal{P}_1$, and we will further use $\mathcal{P}_2$ to guide our algorithm design and analysis in Section V.

Recall that the Wasserstein distance, barycenter and regularization have been defined in (4), (5), and (6), respectively. Now, we are ready to formulate the cost minimization for a given layer among K NNs in the AIoT system as

$$\mathcal{P}_1: \min_{\mathbf{a} \in \mathbb{R}^m} \sum_{k \in \mathcal{K}} \rho_k \left(\min_{\mathbf{T} \in \mathcal{V}(\mathbf{a}_k, \mathbf{a})} \langle \mathbf{T}, \mathbf{C} \rangle - \lambda \Omega(\mathbf{T}) \right)^{\frac{1}{p}}.$$

Note that $\mathcal{P}_1$ differs from (5) in that multiple probability masses are transported to a single destination, while the latter operates in the opposite way.

Lemma 1: The optimal solution $\mathbf{T}^*$ to $\mathcal{P}_1$ has the form

$$\mathbf{T}^* = \text{diag}(\mathbf{u}) \mathbf{K} \text{diag}(\mathbf{v}) \tag{8}$$

where $\mathbf{u} \in \mathbb{R}^m$ and $\mathbf{v} \in \mathbb{R}^n$ represent two scaling variables, and $\mathbf{K} \in \mathbb{R}^{m \times n}$ with $\mathbf{K}^{[i,j]} = e^{-\mathbf{C}^{[i,j]}/\lambda}$.

Proof: By introducing two dual variables $\mathbf{f} \in \mathbb{R}^m$ and $\mathbf{g} \in \mathbb{R}^n$ for each marginal constraint, the Lagrangian of $\mathcal{P}_1$ can be expressed as

$$\begin{aligned} L_k(\mathbf{T}, \mathbf{f}, \mathbf{g}) = \langle \mathbf{T}, \mathbf{C} \rangle - \lambda \Omega(\mathbf{T}) \\ - \langle \mathbf{f}, \mathbf{T} \mathbf{1}_n - \mathbf{a}_k \rangle - \left\langle \mathbf{g}, \mathbf{T}^T \mathbf{1}_m - \mathbf{a} \right\rangle. \end{aligned} \tag{9}$$

After partially differentiating (9) with respect to $\mathbf{T}^{[i,j]}$, we have

$$\frac{\partial L_k(\mathbf{T}, \mathbf{f}, \mathbf{g})}{\partial \mathbf{T}^{[i,j]}} = \mathbf{C}^{[i,j]} + \lambda \log\left(\mathbf{T}^{[i,j]} \right) - \mathbf{f}^{[i]} - \mathbf{g}^{[j]} \tag{10}$$

based on which we see that the (i, j)th entry of the OT matrix $\mathbf{T}^*$ is given by

$$\mathbf{T}^*_{ij} = e^{\mathbf{f}^{[i]}/\lambda} \mathbf{K}^{[i,j]} e^{\mathbf{g}^{[j]}/\lambda} = \mathbf{u}^{[i]} \mathbf{K}^{[i,j]} \mathbf{v}^{[j]} \tag{11}$$

where $\mathbf{f}^{[i]}$ refers to the ith entry of $\mathbf{f}$, and $\mathbf{g}^{[j]}$ refers to the jth entry of $\mathbf{g}$, and the last equality results from the settings of

$$\mathbf{u} \coloneqq e^{\mathbf{f}/\lambda} \quad \text{and} \quad \mathbf{v} \coloneqq e^{\mathbf{g}/\lambda}. \tag{12}$$

Finally, orchestrating (11) in matrix form gives rise to (8). ■

In practice, computing the OT matrix $\mathbf{T}^*$ may encounter numerical overflows if the regularization parameter λ is relatively small as compared to the entries of the transport cost matrix $\mathbf{C}$. Thanks to Lemma 1, the issue of numerical overflows can be alleviated by carrying out the computations in the log domain. The relevance of this approach is clearer by considering the equivalent form $\mathcal{P}_2$, in which these log-domain computations arise naturally

$$\mathcal{P}_2: \quad \min_{\mathbf{a} \in \mathbb{R}^m} \sum_{k \in \mathcal{K}} \rho_k \left(\min_{\mathbf{f} \in \mathbb{R}^m, \mathbf{g} \in \mathbb{R}^n} \psi(\mathbf{f}, \mathbf{g}) \right)^{\frac{1}{p}}$$

where

$$\begin{aligned} \psi(\mathbf{f}, \mathbf{g}) = \lambda \mathbf{1}^T \text{diag}\left(e^{\mathbf{f}/\lambda} \right) \mathbf{K} \text{diag}\left(e^{\mathbf{g}/\lambda} \right) \mathbf{1} \\ - \langle \mathbf{f}, \mathbf{a} \rangle - \langle \mathbf{g}, \mathbf{b} \rangle. \end{aligned} \tag{13}$$

Theorem 1: $\mathcal{P}_2$ is equivalent to $\mathcal{P}_1$.

Proof: Recall that the optimal solution to $\mathcal{P}_1$ can be factorized in matrix form as $\mathbf{T}^* = \text{diag}(\mathbf{u}) \mathbf{K} \text{diag}(\mathbf{v})$ according to Lemma 1. Due to the probabilistic couplings $\mathcal{V}(\mathbf{a}, \mathbf{b})$, it is

clear that the dual variables $\mathbf{u}$ and $\mathbf{v}$ must satisfy the following nonlinear equations:

$$\operatorname{diag}\left(e^{\mathbf{f}/\lambda}\right)\mathbf{K}\operatorname{diag}\left(e^{\mathbf{g}/\lambda}\right)\mathbf{1}_n = \mathbf{a} \tag{14}$$

$$\operatorname{diag}\left(e^{\mathbf{g}/\lambda}\right)\mathbf{K}^T\operatorname{diag}\left(e^{\mathbf{f}/\lambda}\right)\mathbf{1}_m = \mathbf{b}. \tag{15}$$

In addition, it is straightforward to see that

$$\begin{aligned}\log\left(\mathbf{T}^*\right) &= \log(\operatorname{diag}(\mathbf{u})\mathbf{K}\operatorname{diag}(\mathbf{v})) \\ &= \begin{bmatrix} \log\left(\mathbf{u}^{[1]}\mathbf{K}^{[1,1]}\mathbf{v}^{[1]}\right) & \log\left(\mathbf{u}^{[1]}\mathbf{K}^{[1,2]}\mathbf{v}^{[2]}\right) & \cdots \\ \log\left(\mathbf{u}^{[2]}\mathbf{K}^{[2,1]}\mathbf{v}^{[1]}\right) & \log\left(\mathbf{u}^{[2]}\mathbf{K}^{[2,2]}\mathbf{v}^{[2]}\right) & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix} \\ &= \frac{1}{\lambda}\left(\mathbf{f}\mathbf{1}_n^T - \mathbf{C} + \mathbf{1}_m\mathbf{g}^T\right) \end{aligned} \tag{16}$$

where the last equality results from (12). Again, according to Lemma 1, we see that

$$\begin{aligned}\Omega\left(\mathbf{T}^*\right) &= -\left\langle \mathbf{T}^*, \log\left(\mathbf{T}^*\right) - \mathbf{1}_{m\times n}\right\rangle \\ &= -\frac{1}{\lambda}\Big\langle \operatorname{diag}\left(e^{\mathbf{f}/\lambda}\right)\mathbf{K}\operatorname{diag}\left(e^{\mathbf{g}/\lambda}\right) \\ &\qquad \mathbf{f}\mathbf{1}_n^T - \mathbf{C} + \mathbf{1}_m\mathbf{g}^T - \lambda\mathbf{1}_{m\times n}\Big\rangle \\ &= \frac{1}{\lambda}\langle \mathbf{f}, \mathbf{a}\rangle - \frac{1}{\lambda}\left\langle e^{\mathbf{f}/\lambda}, \mathbf{K}^\circ e^{\mathbf{g}/\lambda}\right\rangle \\ &\quad + \frac{1}{\lambda}\langle \mathbf{g}, \mathbf{b}\rangle - \left\langle e^{\mathbf{f}/\lambda}, \mathbf{K}e^{\mathbf{g}/\lambda}\right\rangle \end{aligned} \tag{17}$$

where the second equality results from (16), the last one follows according to (14) and (15), and

$$\mathbf{K}^\circ = \begin{bmatrix} \mathbf{C}^{[1,1]}e^{-\mathbf{C}^{[1,1]}/\lambda} & \mathbf{C}^{[1,2]}e^{-\mathbf{C}^{[1,2]}/\lambda} & \cdots \\ \mathbf{C}^{[2,1]}e^{-\mathbf{C}^{[2,1]}/\lambda} & \mathbf{C}^{[2,2]}e^{-\mathbf{C}^{[2,2]}/\lambda} & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix}. \tag{18}$$

By the fact that

$$\left\langle e^{\mathbf{f}/\lambda}, e^{\mathbf{g}/\lambda}\mathbf{K}^\circ\right\rangle = \left\langle \mathbf{T}^*, \mathbf{C}\right\rangle \tag{19}$$

combining (17) and (19) yields the Lagrangian

$$L_k\left(\mathbf{T}^*, \mathbf{f}, \mathbf{g}\right) = \langle \mathbf{f}, \mathbf{a}\rangle + \langle \mathbf{g}, \mathbf{b}\rangle - \lambda\left\langle e^{\mathbf{f}/\lambda}, \mathbf{K}e^{\mathbf{g}/\lambda}\right\rangle. \tag{20}$$

By the fact that maximizing the Lagrangian serves as a lower bound of the original minimization problem, therefore we have

$$\begin{aligned}\max_{\mathbf{f}\in\mathbb{R}^n, \mathbf{g}\in\mathbb{R}^m} &\langle \mathbf{f}, \mathbf{a}\rangle + \langle \mathbf{g}, \mathbf{b}\rangle - \lambda\left\langle e^{\mathbf{f}/\lambda}, \mathbf{K}e^{\mathbf{g}/\lambda}\right\rangle \\ &= \min_{\mathbf{f}\in\mathbb{R}^n, \mathbf{g}\in\mathbb{R}^m} \psi(\mathbf{f}, \mathbf{g}) \end{aligned} \tag{21}$$

which implies the equivalence between $\mathcal{P}_1$ and $\mathcal{P}_2$. ■

V. Algorithm Design and Analysis

In this section, we apply the concepts of Wasserstein distance and barycenter to design the CODE algorithm, and then present how to employ the Sinkhorn projection technique [59] to design the ϵ-OT subroutine. Then, we present the convergence analysis for the integrated solution, where the ϵ-OT subroutine is invoked iteratively as a subroutine of the CODE algorithm.

Algorithm 1 CODE Algorithm

Input: $\mathcal{K}$, $\{\mathcal{N}_{k,l}\}$, L, ϵ, λ.
Output: $\left\{\overline{\mathbf{W}}_{\bar{k},l}\right\}$.

```
Sort the NNs of K into K°;                                   ▷ [A1]
Set the first NN of K° as the non target NN k^NT;
for q = 2 to |K| do                                          ▷ [A7]
    Select the q-th NN of K° as the target NN k^T;           ▷ [A2]
    Calculate the initial transport matrix T_{k^NT,1} based on (22);
    for l = 2 to L do                                        ▷ [A3]
        Calculate the pre-processed model weights Ŵ_{k^NT,l} and
        transport cost matrices C^{[i,j]}_{k^NT,k^T,l} based on (23) and (24),
        respectively;                                        ▷ [A4]
        Calculate the post-processed model weights W̃_{k^NT,l} based
        on (25);                                             ▷ [A5]
        Get the model weights W̄_{k̄,l} of the synthetic NN k̄ based
        on (26);                                             ▷ [A6]
        Set the synthetic NN k̄ as the non-target k^NT.
```

A. Coupled Model Averaging Algorithm

We provide the detailed descriptions of the CODE algorithm (see Algorithm 1) as follows.

1) [**A1**] *Initial Selection of a Nontarget NN:* In the beginning, with all of the locally trained models on hand, we sort the NNs in ascending or descending order according to their test accuracies. Let $\mathcal{K}^\circ$ be the sorted list of NNs. In each pair of NNs, we have a nontarget NN and a target NN, where the former gets aligned to the latter. Here, we select the first NN from $\mathcal{K}^\circ$ as the nontarget NN k^{NT}.
2) [**A2**] *Selection of a Target NN:* Given the selected nontarget NN k^{NT}, we need to select a target NN so as to perform an MA. Suppose that we are at the qth iteration. Then, we select the qth NN from $\mathcal{K}^\circ$ as the target NN k^T. Then, we use a uniform distribution to initialize the transport matrix of the input layer for the nontarget NN k^{NT} through

$$\mathbf{T}_{k^{\mathrm{NT}},1} \leftarrow \operatorname{diag}\left(\frac{1}{N_{k^{\mathrm{T}},1}}\mathbf{1}_{N_{k^{\mathrm{T}},1}}\right). \tag{22}$$

3) [**A3**] *Layer-by-Layer Model Alignment:* To keep the layer structure of the target and the nontarget NNs intact, we consider aligning the model weights of a pair of NNs in a layer-by-layer fashion as shown in Figs. 3 and 4. For example, the model weights of the lth layer in the nontarget NN get aligned with respect to the target NN, and the same procedure continues for the $(l+1)$th layer. This procedure begins with the second layer (since the model weights of the first layer are initialized according to [**A2**]). Note that even the number of neurons of a given layer differs among these two NNs, the alignment of model weights can still be handled (to be described in [**A5**]).
4) [**A4**] *Construction of Transport Cost Matrices:* Before invoking Algorithm 2, it is necessary to define the transport cost matrix. To this end, we first preprocess the

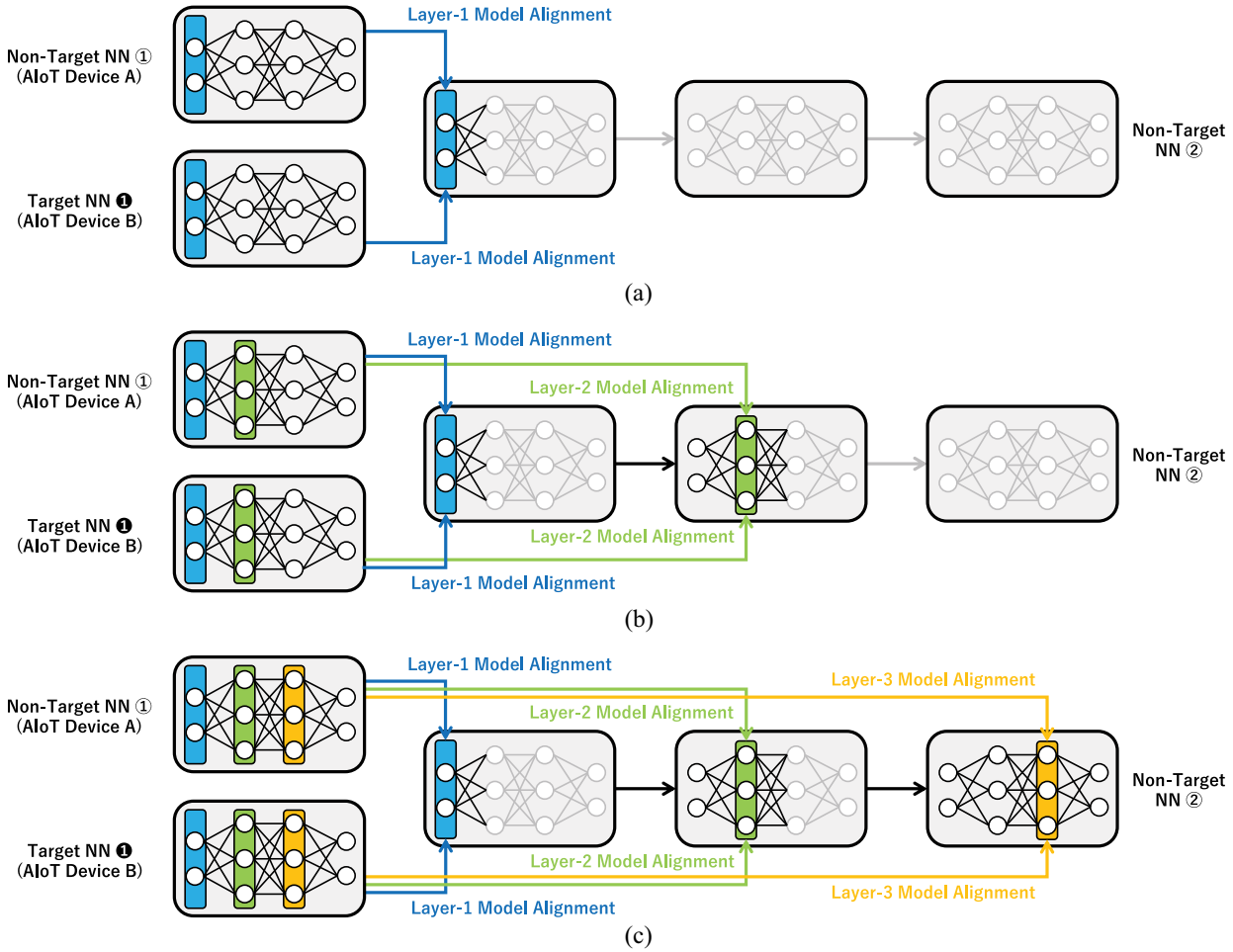


Fig. 3. Procedures of the layer-by-layer model alignment between a nontarget NN and a target NN. (a) Layer-1 model alignment. (b) Layer-2 model alignment. (c) Layer-3 model alignment.

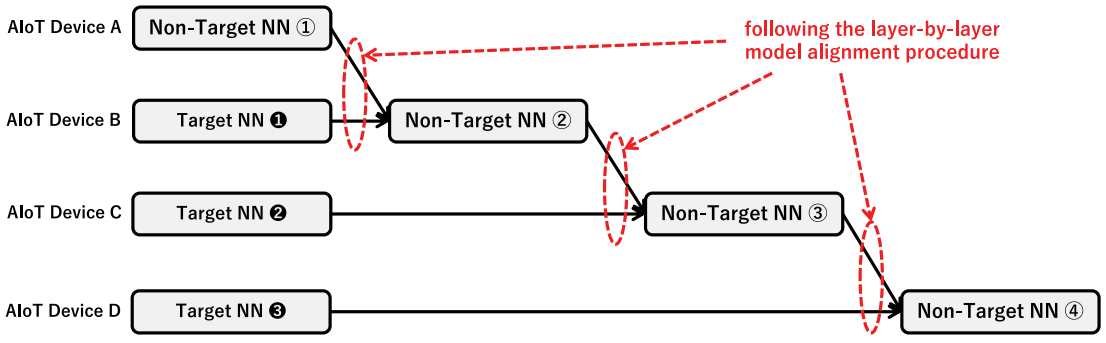


Fig. 4. Evolving roles of the target and nontarget NNs.

model weights of NN k in the lth layer through

$$\widehat{\mathbf{W}}_{k^{\mathrm{NT}},l} \leftarrow \mathbf{W}_{k^{\mathrm{NT}},l}\mathbf{T}_{k^{\mathrm{NT}},l}\mathrm{diag}\left(\frac{1}{N_{k^{\mathrm{T}},l}}\mathbf{1}_{N_{k^{\mathrm{T}},l}}\right) \quad \forall l \in \{1,\ldots,L\} \tag{23}$$

where the model weights of the nontarget NN k^{NT} are weighted by $\mathbf{T}_{k^{\mathrm{NT}},l}$ and then properly normalized by a diagonal matrix. Then, we are now ready to construct the transport cost matrix of the lth layer from nontarget NN k to the target NN $\hat{k}$ through

$$\mathbf{C}^{[i,j]}_{k^{\mathrm{NT}},k^{\mathrm{T}},l} \leftarrow \left\|\widehat{\mathbf{W}}^{[i]}_{k^{\mathrm{NT}},l} - \widehat{\mathbf{W}}^{[j]}_{k^{\mathrm{T}},l}\right\|_2 \quad \forall i \in \mathcal{N}_{k^{\mathrm{NT}},l}, j \in \mathcal{N}_{k^{\mathrm{T}},l}, l \in \{1,\ldots,L\} \tag{24}$$

where $\widehat{\mathbf{W}}^{[i]}_{k^{\mathrm{NT}},l}$ and $\widehat{\mathbf{W}}^{[j]}_{k^{T},l}$ denote the ith and the jth entries of the model weights of NN k^{NT} and k^{T} in the lth layer, respectively. Note that the 2-norm used in (24) corresponds to the setting of $d_{\mathcal{S}}$ to $\|\cdot\|_2$ in (4).

5) [**A5**] *Adjustment of Model Weights:* Here, we run Algorithm 2 to get an ϵ-approximate transport matrix

Algorithm 2 ϵ-OT Subroutine

Input: ϵ, $\mathbf{C}$, λ.
Output: $\widehat{\mathbf{T}}$.

```
Initialize the iteration index t ← 0;
Initialize the auxiliary variables f_(0), g_(0) ← 0;                ▷ [B1]
while the transport matrix T̂_(t) satisfies (27) do                  ▷ [B2]
    if t is even then                                               ▷ [B3]
        Update the auxiliary variable f_(t) according to (28);
        Set the auxiliary variable g_(t+1) ← g_(t);
    else                                                            ▷ [B3]
        Update the auxiliary variable g_(t) according to (29);
        Set the auxiliary variable f_(t+1) ← f_(t);
    Construct the transport matrix T̂_(t+1) according
    to (30);                                                        ▷ [B4]
    Increase the iteration index t ← t + 1;
Set the ϵ-approximate transport matrix T̂ ← T̂_(t).
```

$\mathbf{T}_{k,l}$ between the probability measures of the pair of NNs k^{NT} and k^T. Then, we get the adjusted model weights by using $\mathbf{T}_{k,l}$ to align the model weights of NN k^{NT} with respect to the target NN k^T through

$$\widetilde{\mathbf{W}}_{k^{\mathrm{NT}},l} \leftarrow \mathrm{diag}\left(\frac{1}{N_{k^{\mathrm{T}},l+1}}\mathbf{1}_{N_{k^{\mathrm{T}},l+1}}\right)\mathbf{T}^T_{k^{\mathrm{NT}},l+1}\widehat{\mathbf{W}}_{k^{\mathrm{NT}},l} \quad \forall l \in \{1,\ldots,L-1\}. \tag{25}$$

6) [**A6**] *Averaging of Model Weights:* Next, we average the model weights between the pair of NNs k^{NT} and k^T to obtain those of the synthetic NN $\bar{k}$ through

$$\overline{\mathbf{W}}_{\bar{k},l} = \frac{1}{2}\left(\widetilde{\mathbf{W}}_{k^{\mathrm{NT}},l} + \mathbf{W}_{k^{\mathrm{T}},l}\right) \quad \forall l \in \{1,\ldots,L\}. \tag{26}$$

Note that the synthetic NN $\bar{k}$ essentially has the same structure (e.g., the number of layers and neurons) as the target NN k^T.

7) [**A7**] *Repeating Procedure:* If there are any NNs remain un-selected, we assign the synthetic NN $\bar{k}$ as the new nontarget NN k^{NT}, and then go back to [**A2**]. Otherwise, the algorithm is terminated, and the final synthetic NN $\bar{k}$ becomes the globally averaged model.

B. ϵ-OT Subroutine

In the following, we first provide the definition of ϵ-approximate transport matrices. Then, we give the detailed descriptions of the ϵ-OT subroutine (see Algorithm 2), which produces an ϵ-approximate transport matrix for the optimization problem $\mathcal{P}_2$ and meanwhile serves as a subroutine of the CODE algorithm (i.e., line 9 of Algorithm 1).

Definition 1: Given a transport cost matrix $\mathbf{C}$ and the probability measures $\mathbf{a}$ and $\mathbf{b}$, the transport matrix $\bar{\mathbf{T}} \in \mathbb{R}_+^{m\times n}$ is called an ϵ-approximate transport matrix if $\bar{\mathbf{T}}\mathbf{1}_n = \mathbf{a}$ and $\bar{\mathbf{T}}^T\mathbf{1}_m = \mathbf{b}$, and the following inequality holds true:

$$\left\langle \bar{\mathbf{T}}, \mathbf{C}\right\rangle \le \left\langle \mathbf{T}^*, \mathbf{C}\right\rangle + \epsilon$$

where $\mathbf{T}^*$ is an OT matrix for $\mathcal{P}_0$.

With this definition in mind, our aim is to design an algorithm that produces ϵ-approximate transport matrix and has a convergence bound parameterized by $1/\epsilon$. In the following, we elaborate on the design of the ϵ-OT subroutine.

1) [**B1**] *Auxiliary Variable Initialization:* According to $\mathcal{P}_2$, there are two auxiliary variables $\mathbf{f}$ and $\mathbf{g}$ need to be determined. Here, the ϵ-OT subroutine tends to iteratively update those auxiliary variables, we denote by $\mathbf{f}_{(t)}$ and $\mathbf{g}_{(t)}$ their values at the tth iteration, and we initialize them to zeros.
2) [**B2**] *Convergence Criterion:* The ϵ-OT subroutine aims to iteratively update the auxiliary variables $\mathbf{g}_{(t)}$ until the constraints are met given the predetermined accuracy ϵ. In particular, we have the stopping criterion expressed as

$$\left\|\widehat{\mathbf{T}}_{(t)}\mathbf{1} - \mathbf{a}\right\|_1 + \left\|\widehat{\mathbf{T}}^T_{(t)}\mathbf{1} - \mathbf{b}\right\|_1 \le \epsilon \tag{27}$$

where $\widehat{\mathbf{T}}_{(t)}$ refers to the transport matrix at the end of tth iteration. Once (27) can be satisfied, an ϵ-approximate transport matrix will then be output.
3) [**B3**] *Bidirectional Update:* Since we have two auxiliary variables to be updated, we consider updating them in turn. When the iteration index t is even, we update the auxiliary variable $\mathbf{f}_{(t)}$ according to

$$\mathbf{f}_{(t+1)} \leftarrow \mathbf{f}_{(t)} + \log(\mathbf{a}) - \log\left(\widehat{\mathbf{T}}_{(t)}\mathbf{1}\right). \tag{28}$$

Similarly, we update the auxiliary variable $\mathbf{g}_{(t)}$ according to

$$\mathbf{g}_{(t+1)} \leftarrow \mathbf{g}_{(t)} + \log(\mathbf{b}) - \log\left(\widehat{\mathbf{T}}^T_{(t)}\mathbf{1}\right). \tag{29}$$

4) [**B4**] *Transport Matrix Construction:* When $\mathbf{f}_{(t)}$ or $\mathbf{g}_{(t)}$ gets updated, we can construct the transport matrix through

$$\widehat{\mathbf{T}}_{(t+1)} \leftarrow \mathrm{diag}\left(e^{\mathbf{f}_{(t+1)}/\lambda}\right)\mathbf{K}\mathrm{diag}\left(e^{\mathbf{g}_{(t+1)}/\lambda}\right)\mathbf{1}. \tag{30}$$

The transport matrix will be iteratively updated, and its converged one will fall within the specified convergence parameter ϵ.

Remark 3: In the proposed solution, the CODE needs to be operated over feed-forward NNs [e.g., multilayer perceptron (MLP)]. In addition, the CODE works in a layer-by-layer fashion, and therefore the number of layers needs to be the same for all NNs. However, the number of neurons of the same layer across NNs can be different.

Remark 4: The computing costs induced by the CODE algorithm are different from the FedAvg algorithm as follows.

1) From the perspective of a PS, both one-shot and multishot versions of the FedAvg algorithm have the least energy consumption as they simply need weighted averaging operations. On the other hand, the proposed solution needs to run the ϵ-OT subroutines, and hence it incurs higher computing costs.
2) From the perspective of an AIoT device, the proposed solution has identical computing costs with respect to the one-shot FedAvg algorithm, and lower computing costs with respect to the multishot FedAvg algorithm. In spite of the elevated computing costs at the PS, the

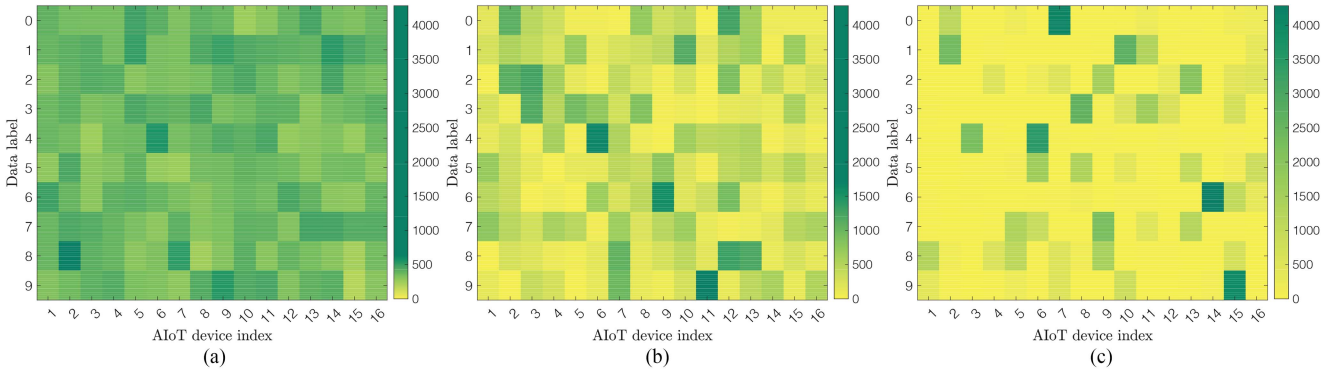


Fig. 5. Data heterogeneity among the AIoT devices subject to various parameters of the Dirichlet distribution. (a) $\alpha = 10^{20}$. (b) $\alpha = 1$. (c) $\alpha = 0.1$.

proposed solution is advantageous in alleviating the burden of computing for AIoT devices (especially resource- and/or energy-constrained ones).

C. Convergence Analysis

To analyze the convergence of the proposed solution, we have to estimate the number of iterations required for reaching the stopping criteria. In particular, we first formally define the differential transport costs, and then quantify how they can be upper bounded according to the changes of dual variables.

Definition 2: Given any two variables $\mathbf{f}$ and $\mathbf{g}$, we define the differential transport costs as

$$\begin{aligned}\widetilde{\psi}(\mathbf{f},\mathbf{g}) &= \psi(\mathbf{f},\mathbf{g}) - \psi\left(\mathbf{f}^*,\mathbf{g}^*\right) \\ &= \langle \mathbf{1}, \mathbf{B}(\mathbf{f},\mathbf{g})\mathbf{1}\rangle - \left\langle \mathbf{1}, \mathbf{B}\left(\mathbf{f}^*,\mathbf{g}^*\right)\mathbf{1}\right\rangle \\ &\quad + \left\langle \mathbf{f}^* - \mathbf{f}, \mathbf{a}\right\rangle + \left\langle \mathbf{g}^* - \mathbf{g}, \mathbf{b}\right\rangle. \end{aligned} \tag{31}$$

Lemma 2: At the end of each iteration t, the differential transport costs induced by Algorithm 2 satisfies

$$\begin{aligned}&\widetilde{\psi}\left(\mathbf{f}_{(t)},\mathbf{g}_{(t)}\right) \\ &\le R\left(\left\|\mathbf{B}\left(\mathbf{f}_{(t)},\mathbf{g}_{(t)}\right)\mathbf{1} - \mathbf{a}\right\|_1 + \left\|\mathbf{B}\left(\mathbf{f}_{(t)},\mathbf{g}_{(t)}\right)^T\mathbf{1} - \mathbf{b}\right\|_1\right)\end{aligned} \tag{32}$$

where $R = -\ln(e^{-\|\mathbf{C}\|_\infty/\lambda}\min(\check{\mathbf{a}},\check{\mathbf{b}}))$, $\check{\mathbf{a}} = \min_{1\le i\le m}\mathbf{a}^{[i]}$, and $\check{\mathbf{b}} = \min_{1\le j\le n}\mathbf{b}^{[j]}$.

Proof: See Appendix A. ■

Given the upper bounds derived in Lemma 2, we are now ready to assess the convergence of the proposed solution.

Theorem 2: The convergence of the integrated solution of Algorithms 1 and 2 is given by $\mathcal{O}(|\mathcal{K}|LR/\epsilon)$.

Proof: See Appendix B. ■

VI. Performance Evaluation

We consider an AIoT system composed of 16 AIoT devices and a PS. For all AIoT devices, we use the same configuration to set up their individual NNs. All AIoT devices perform deep learning tasks for the standard image classification using the MNIST dataset [60], which consists of 60 000 training images and 10 000 testing images. In addition, we use the MLP [61] (denoted as MLP) to construct a fully connected NN with three hidden layers, which consists of 784 input neurons, ten output neurons, and the hidden layers have 400, 200, and 100 neurons, respectively. The local training is conducted over 10 epochs, and the batch size is 64. On the other hand, we set the OT regularization parameter to 0.01 and the OT convergence parameter to 10^{-7}. The order of Wasserstein distance is set to 2. All of the default parameter settings used throughout all simulation results are summarized in Table II unless stated otherwise.

TABLE II
Default Parameter Settings

Parameter	Value
The number of AIoT devices, $\lvert\mathcal{K}\rvert$	16
Data usage, $\mathcal{D}_k$	MNIST dataset [60]
The heterogeneity of data distribution	Dirichlet distribution [62]
The configuration of an NN	MLP [61]
The number of layers of each NN, L	5 (with 3 hidden layers)
The number of neurons of each NN, $N_{k,l}$	784, 400, 200, 100, 10 for $l = 1, 2, \ldots, 5$, resp.
Optimizer for training	SGD, Adam
The number of epochs	10
Learning rate	(SGD) 0.1
	(Adam) 0.001
Batch size	64
OT regularization parameter, λ	0.01
OT convergence parameter, ϵ	10^{-7}
The order of Wasserstein distance, p	2

To demonstrate the impacts of data heterogeneity, the whole dataset is partitioned among the AIoT devices according to the Dirichlet distribution [62], denoted as Dir(α) for all $\alpha \ge 0$, where α is the parameter determining the degree of data heterogeneity. The smaller the parameter α, the more pronounced the heterogeneity of the data distributions across AIoT devices, and vice versa. In Fig. 5, we consider one IID setting (i.e., $\alpha = 10^{20}$) and two NIID settings (i.e., $\alpha = 1$, 0.1). In Fig. 5(a), we observe that the training data is almost uniformly allocated to each AIoT device. When the data distributions across AIoT devices go heterogeneous, we see from Fig. 5(b), the training data is unevenly allocated to the AIoT devices. Once if the data heterogeneity becomes more pronounced as Fig. 5(c), we can see that each AIoT device holds massive training data of some data labels but very few of others.

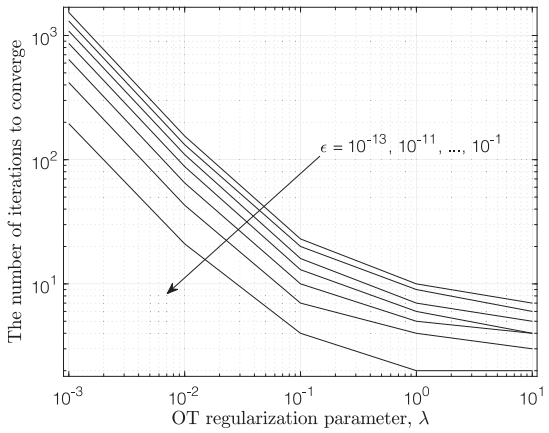


Fig. 6. Impacts of the OT convergence parameter.

To evaluate the performance of our proposed solution, we consider the following MA mechanisms.

1) CODE: The integrated solution of the CODE algorithm and the ϵ-OT subroutine, where the locally trained models are sorted in ascending order of their prediction accuracies. Then, the MA will be performed by the CODE algorithm, where the NNs are sorted in ascending order of their individual test accuracies.
2) LMF: Following the OT-based layer-wise model fusion [54], the model weights are averaged across NNs first given a layer, and the same procedure proceeds until the last layer. Note that this mechanism also requires to solve OT inherently, and hence we apply the ϵ-OT subroutine here for a fair comparison.
3) FedAvg-Δ: The model weights are averaged across NNs for all layers according to the FedAvg algorithm [10], where Δ denotes the number of shots (i.e., communication rounds) elapsed. Note that FedAvg-1 is in essence a Vanilla averaging.

For ease of demonstration, CODE, LMF, and FedAvg-1 are illustrated in nongray colors, while FedAvg-10, FedAvg-30, and FedAvg-50 are drawn in gray colors.

Figs. 6 and 7 illustrate the impacts of OT regularization and convergence parameters, respectively. For this, we restrict our focuses to executing the ϵ-OT subroutine, instead of the whole integrated solution. In Fig. 6, we see that the number of iterations required for the algorithm to converge decreases with the OT convergence parameter. The greater the OT convergence parameter, the smaller the elapsed time for iteratively updating the variables. In addition, when the OT regularization parameter increases, the required number of iterations reduces due to the tolerance to more blurred OT solution. On the other hand, Fig. 7 depicts the output transport matrices corresponding to different OT regularization parameters. The greater the OT regularization parameter, the less concentrated the entry values (i.e., the less the zero-valued entries). This is because the OT regularization parameter tends to make the transport matrix smoother, and hence increasing its value results in less sparse transport matrices.

Fig. 8 shows the impacts of number of AIoT devices in terms of test accuracies. From Fig. 8(a)–(c), we see that CODE significantly outperforms FedAvg-1. This is because FedAvg-1 tends to be under fitting and thus far away from reaching a converged global model, and such a situation deteriorates when the degree of NIID-ness increases. In addition, CODE is comparable to LMF under the IID setting, and it tends to be superior when the degree of NIID-ness increases, thanks to its efficient CODE. When comparing Fig. 8(a)–(c) with Fig. 8(d)–(f), the test accuracies of all mechanisms are elevated thanks the rapid convergence of Adam with respect to SGD. On the other hand, FedAvg-10, FedAvg-30, and FedAvg-50 can gradually converge to global models that yield better test accuracies than CODE when the number of AIoT devices is not excessive, due to the benefits of performing MA over multiple communication rounds. Nevertheless, CODE is advantageous in that it can achieve a satisfactory test accuracy in merely one communication round, which can effectively reduce the computing and communication costs for AIoT devices.

Fig. 9 illustrates how the number of epochs in local training affects the test accuracies. From Fig. 9(a)–(c), we observe that the test accuracy of CODE increases with the number of epochs, whereas FedAvg-1 completely diverges in the presence of tens of AIoT devices regardless of the degree of NIID-ness. Similarly, CODE is comparable to LMF under the IID setting, and it outperforms when the degree of NIID-ness increases. When comparing Fig. 9(a)–(c) with Fig. 9(d)–(f), the test accuracies of all mechanisms (except FedAvg-1) improve with the number of epochs, where SGD grows more than Adam as the former requires more local training to reach the same level of the latter. On the other hand, FedAvg-10, FedAvg-30 and FedAvg-50 can reach good test accuracies with few epochs due to multiple communication rounds elapsed, and CODE is comparable to them under the IID setting with a sufficient number of epochs. The performance gap in between inevitably enlarges when the degree of NIID-ness increases, since CODE has to finish MA based on the locally trained models that look very different in merely one communication round. Nevertheless, CODE performs much better than LMF under the NIID settings.

To demonstrate how our proposed solution is applicable to advanced NNs, we implement a CNN (identical to VGG-11 [63] except that only one fully connected layer is used) for ease of demonstration, which is denoted as CNN for brevity. To this end, we have to make the following configurations specifically for the MNIST dataset:

1) The MNIST dataset is merely constituted by grayscale images, whereas CNN generically takes RGB images as its input. Therefore, we have to map one-channel data (i.e., grayscale images) to three-channel one (i.e., RGB images) by triplicating the gray values. For example, if the gray value is 10, the corresponding RGB value will be (10, 10, 10).
2) Since each image of the MNIST dataset is 28×28, which is incompatible with the input format of CNN, say 224×224. Therefore, we resize the images of the MNIST dataset by means of zero-padding to ensure the input compatibility.

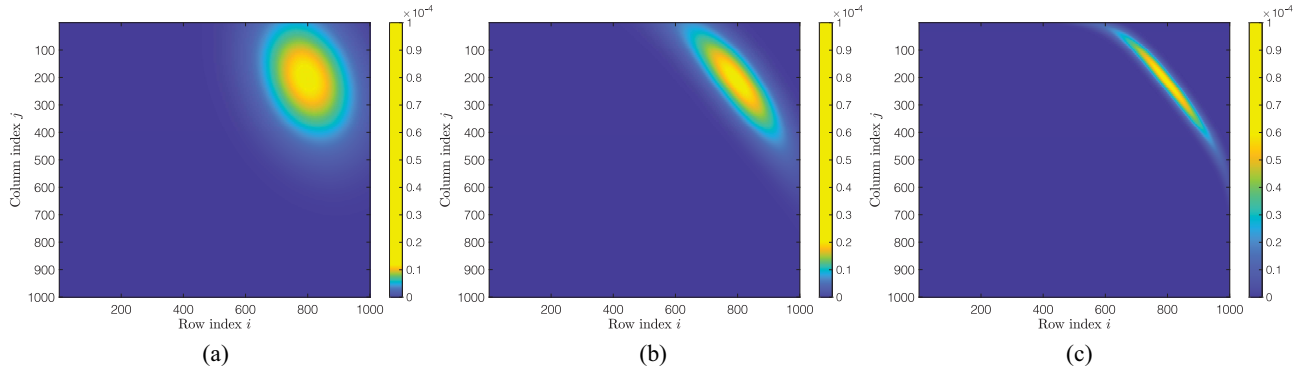


Fig. 7. Impacts of the OT regularization parameter. (a) $\lambda = 0.1$. (b) $\lambda = 0.01$. (c) $\lambda = 0.001$.

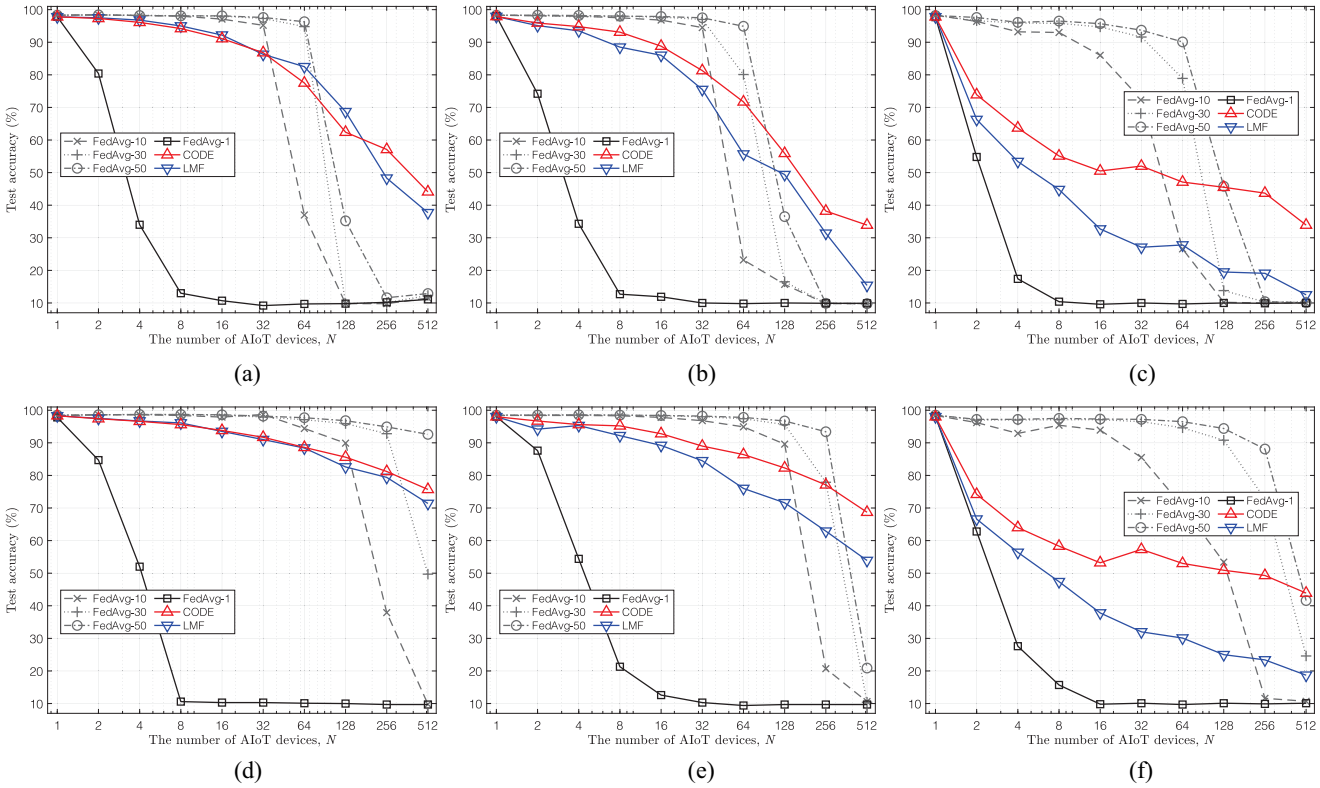


Fig. 8. Impacts of the numbers of AIoT devices under different optimizers and various data heterogeneity settings. (a) SGD, $\alpha = 10^{20}$. (b) SGD, $\alpha = 1$. (c) SGD, $\alpha = 0.1$. (d) Adam, $\alpha = 10^{20}$. (e) Adam, $\alpha = 1$. (f) Adam, $\alpha = 0.1$.

Fig. 10 shows how different types of NN architectures affect the test accuracies, where we use the Adam optimizer with the learning rate of 0.0001. In Fig. 10(a)–(c), we observe that MLP works better than CNN in both CODE and LMF under the one-shot setting. This is because MLP has a much simpler NN architecture than CNN, and hence the former converges soon while the latter tends to be underfitting. In addition, when extending the one-shot setting to the few-shot ones, we see that CNN outperforms MLP since it can extract features and converge better thanks to more local training opportunities. Notably, CODE aided with CNN can achieve promising performance in the presence of data heterogeneity as long as a few shots are given.

Remark 5: For the datasets with merely few object classes (e.g., the considered MNIST dataset), simple NNs (e.g., MLP) may work better under the one-shot setting, while more advanced NNs (e.g., CNN) are preferable to boost learning performance if a few more shots are allowable, especially when the data heterogeneity across AIoT devices increases.

VII. Conclusion

FL enabling AIoT devices to collaboratively train ML models on the network edge has been widely regarded a promising solution for realizing distributed ML. In this article, we investigated the design of AIoT systems, where AIoT devices perform local training without exposing their own data and a PS is responsible for averaging the locally trained models. To assess the cost effectiveness of AIoT systems, we leveraged OT to formulate the transport costs of averaging the model weights of moving couples of AIoT devices. Then, we

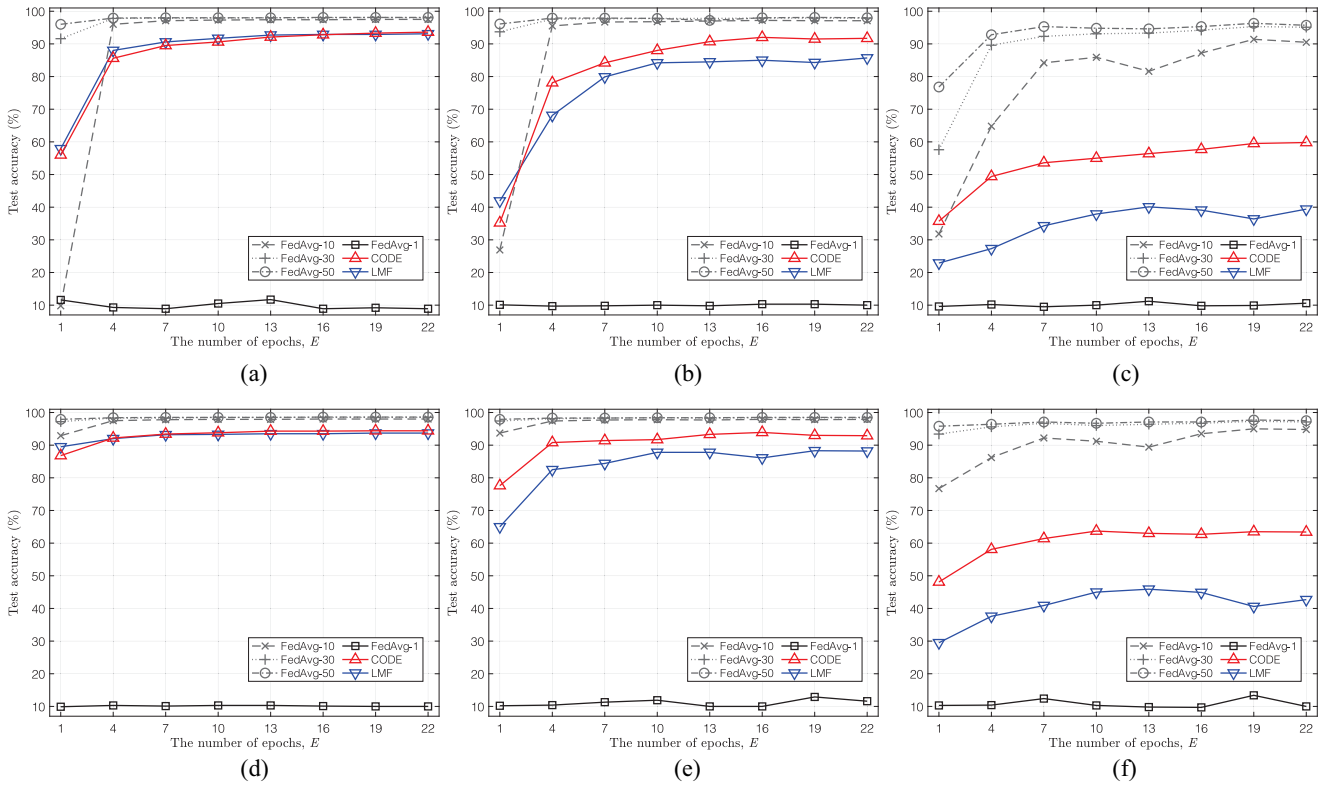


Fig. 9. Impacts of the numbers of epochs under different optimizers and various data heterogeneity settings. (a) SGD, $\alpha = 10^{20}$. (b) SGD. $\alpha = 1$. (c) SGD. $\alpha = 0.1$. (d) Adam, $\alpha = 10^{20}$. (e) Adam, $\alpha = 1$. (f) Adam, $\alpha = 0.1$.

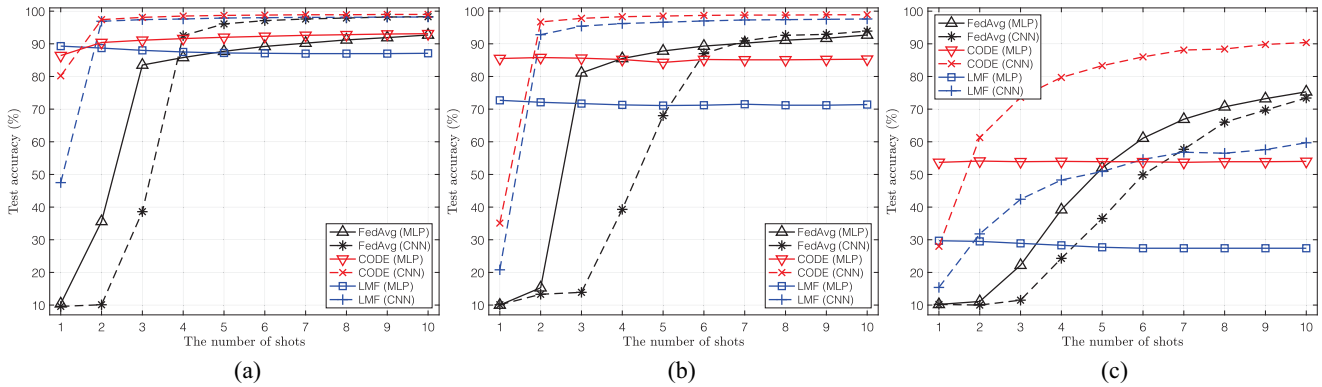


Fig. 10. Impacts of the types of NNs under various data heterogeneity settings. (a) $\alpha = 10^{20}$. (b) $\alpha = 1$. (c) $\alpha = 0.1$.

proposed the CODE algorithm to average locally trained models progressively, wherein the ϵ-OT subroutine is designed to yield ϵ-approximate transport matrices. Our simulation results showed that the proposed solution outperforms other MA mechanisms under various parameter settings, and it benefits much from the locally trained models if their own test accuracies are further elevated.

APPENDIX A

Let us denote the function $\Lambda\big(\mathbf{f}^\circ, \mathbf{g}^\circ | \mathbf{f}_{(t)}, \mathbf{g}_{(t)}\big)$ for $k \geq 1$ and arbitrary $(\mathbf{f}^\circ, \mathbf{g}^\circ)$ as

$$\Lambda\big(\mathbf{f}^\circ, \mathbf{g}^\circ | \mathbf{f}_{(t)}, \mathbf{g}_{(t)}\big) = \big\langle \mathbf{1}, \mathbf{B}\big(\mathbf{f}^\circ, \mathbf{g}^\circ\big)\mathbf{1}\big\rangle - \big\langle \mathbf{f}^\circ, \mathbf{B}\big(\mathbf{f}_{(t)}, \mathbf{g}_{(t)}\big)\mathbf{1}\big\rangle - \Big\langle \mathbf{g}^\circ, \mathbf{B}\big(\mathbf{f}_{(t)}, \mathbf{g}_{(t)}\big)^T\mathbf{1}\Big\rangle \tag{33}$$

which is a convex function of $(\mathbf{f}, \mathbf{g})$. According to Definition 2 and (33), we have

$$\begin{aligned}\widetilde{\psi}\big(\mathbf{f}_{(t)}, \mathbf{g}_{(t)}\big) &= \Lambda\big(\mathbf{f}_{(t)}, \mathbf{g}_{(t)} | \mathbf{f}_{(t)}, \mathbf{g}_{(t)}\big) - \Lambda\big(\mathbf{f}^*, \mathbf{g}^* | \mathbf{f}_{(t)}, \mathbf{g}_{(t)}\big) \\ &\quad + \big\langle \mathbf{f}_{(t)} - \mathbf{f}^*, \mathbf{B}\big(\mathbf{f}_{(t)}, \mathbf{g}_{(t)}\big)\mathbf{1} - \mathbf{a}\big\rangle \\ &\quad + \Big\langle \mathbf{g}_{(t)} - \mathbf{g}^*, \mathbf{B}\big(\mathbf{f}_{(t)}, \mathbf{g}_{(t)}\big)^T\mathbf{1} - \mathbf{b}\Big\rangle \\ &\leq \big\langle \mathbf{f}_{(t)} - \mathbf{f}^*, \mathbf{B}\big(\mathbf{f}_{(t)}, \mathbf{g}_{(t)}\big)\mathbf{1} - \mathbf{a}\big\rangle \\ &\quad + \Big\langle \mathbf{g}_{(t)} - \mathbf{g}^*, \mathbf{B}\big(\mathbf{f}_{(t)}, \mathbf{g}_{(t)}\big)^T\mathbf{1} - \mathbf{b}\Big\rangle \end{aligned} \tag{34}$$

where the inequality holds because the gradient of (33) vanishes at $\big(\mathbf{f}_{(t)}, \mathbf{g}_{(t)}\big)$. In addition, at the end of each iteration, it is clear that $\big\langle \mathbf{1}, \mathbf{B}\big(\mathbf{f}_{(t)}, \mathbf{g}_{(t)}\big)\mathbf{1} - \mathbf{a}\big\rangle = 0$, we have

$$\begin{aligned}
&\left\langle \mathbf{f}_{(t)} - \mathbf{f}^*, \mathbf{B}\left(\mathbf{f}_{(t)}, \mathbf{g}_{(t)}\right)\mathbf{1} - \mathbf{a} \right\rangle \\
&= \left\langle \mathbf{f}_{(t)} - \mathbf{f}^* - \left(\max_i \mathbf{f}_{(t)}^{[i]} - \min_i \mathbf{f}_{(t)}^{[i]}\right)\mathbf{1}, \mathbf{B}\left(\mathbf{f}_{(t)}, \mathbf{g}_{(t)}\right)\mathbf{1} - \mathbf{a} \right\rangle \\
&\leq \left\| \mathbf{f}_{(t)} - \frac{\max_i \mathbf{f}_{(t)}^{[i]} - \min_i \mathbf{f}_{(t)}^{[i]}}{2}\mathbf{1} \right\|_\infty \left\| \mathbf{B}\left(\mathbf{f}_{(t)}, \mathbf{g}_{(t)}\right)\mathbf{1} - \mathbf{a} \right\|_1 \\
&+ \left\| \mathbf{f}^* - \frac{\max_i \mathbf{f}_{(t)}^{[i]} - \min_i \mathbf{f}_{(t)}^{[i]}}{2}\mathbf{1} \right\|_\infty \left\| \mathbf{B}\left(\mathbf{f}_{(t)}, \mathbf{g}_{(t)}\right)\mathbf{1} - \mathbf{a} \right\|_1 \\
&\leq R \left\| \mathbf{B}\left(\mathbf{f}_{(t)}, \mathbf{g}_{(t)}\right)\mathbf{1} - \mathbf{a} \right\|_1 \qquad (35)
\end{aligned}$$

where the first inequality follows by the Hölder's inequality. Using the same procedure, it is straightforward to show that

$$\begin{aligned}
&\left\langle \mathbf{g}_{(t)} - \mathbf{g}^*, \mathbf{B}\left(\mathbf{f}_{(t)}, \mathbf{g}_{(t)}\right)^T\mathbf{1} - \mathbf{b} \right\rangle \\
&\leq R \left\| \mathbf{B}\left(\mathbf{f}_{(t)}, \mathbf{g}_{(t)}\right)^T\mathbf{1} - \mathbf{b} \right\|_1. \qquad (36)
\end{aligned}$$

Combining (35) and (36) completes the proof of this lemma.

Appendix B

Consider that $k \geq 1$ is even, where $\mathbf{g}_{(t)} = \mathbf{g}_{(t+1)}$. Then, we see that

$$\begin{aligned}
&\widetilde{\psi}\left(\mathbf{f}_{(t)}, \mathbf{g}_{(t)}\right) - \widetilde{\psi}\left(\mathbf{f}_{(t+1)}, \mathbf{g}_{(t+1)}\right) \\
&= \psi\left(\mathbf{f}_{(t)}, \mathbf{g}_{(t)}\right) - \psi\left(\mathbf{f}_{(t+1)}, \mathbf{g}_{(t+1)}\right) \\
&= \left\langle \mathbf{1}, \mathbf{B}\left(\mathbf{f}_{(t)}, \mathbf{g}_{(t)}\right)\mathbf{1} \right\rangle - \left\langle \mathbf{1}, \mathbf{B}\left(\mathbf{f}_{(t+1)}, \mathbf{g}_{(t+1)}\right)\mathbf{1} \right\rangle \\
&\quad + \left\langle \mathbf{f}_{(t+1)} - \mathbf{f}_{(t)}, \mathbf{a} \right\rangle + \left\langle \mathbf{g}_{(t+1)} - \mathbf{g}_{(t)}, \mathbf{b} \right\rangle \\
&= \left\langle \mathbf{a}, \mathbf{f}_{(t+1)} - \mathbf{f}_{(t)} \right\rangle \\
&= \left\langle \mathbf{a}, \ln(\mathbf{a}) - \ln\left(\mathbf{B}\left(\mathbf{f}_{(t)}, \mathbf{g}_{(t)}\right)\mathbf{1}\right) \right\rangle \qquad (37)
\end{aligned}$$

where the last inequality follows according to the algorithm construction. In fact, (37) can be linked to the well-known Kullback–Leibler divergence, which defines the measure between two probability distributions $\mathbf{p}$ and $\mathbf{q}$ as

$$\mathsf{KL}(\mathbf{p}\|\mathbf{q}) := \sum_{i=1}^{m}\sum_{j=1}^{n} \mathbf{p}^{[i,j]} \log\left(\frac{\mathbf{p}^{[i,j]}}{\mathbf{q}^{[i,j]}}\right). \qquad (38)$$

Therefore, we obtain

$$\begin{aligned}
&\left\langle \mathbf{a}, \ln(\mathbf{a}) - \ln\left(\mathbf{B}\left(\mathbf{f}_{(t)}, \mathbf{g}_{(t)}\right)\mathbf{1}\right) \right\rangle \\
&= \mathsf{KL}\left(\mathbf{a}\|\mathbf{B}\left(\mathbf{f}_{(t)}, \mathbf{g}_{(t)}\right)\mathbf{1}\right) \\
&\geq \frac{1}{2}\left\| \mathbf{B}\left(\mathbf{f}_{(t)}, \mathbf{g}_{(t)}\right)\mathbf{1} - \mathbf{a} \right\|_1^2 \qquad (39)
\end{aligned}$$

where the inequality follows by the Pinsker's inequality. In addition, according to Lemma 2 and the stopping criterion of Algorithm 2, we see that

$$\left\| \mathbf{B}\left(\mathbf{f}_{(t)}, \mathbf{g}_{(t)}\right)\mathbf{1} - \mathbf{a} \right\|_1 \geq \max\left(\frac{\widetilde{\psi}\left(\mathbf{f}_{(t)}, \mathbf{g}_{(t)}\right)}{R}, \epsilon\right). \qquad (40)$$

Combining (37) and (40), we get

$$\begin{aligned}
&\widetilde{\psi}\left(\mathbf{f}_{(t)}, \mathbf{g}_{(t)}\right) - \widetilde{\psi}\left(\mathbf{f}_{(t+1)}, \mathbf{g}_{(t+1)}\right) \\
&\geq \frac{1}{2}\max\left(\frac{\widetilde{\psi}^2\left(\mathbf{f}_{(t)}, \mathbf{g}_{(t)}\right)}{R^2}, \epsilon^2\right). \qquad (41)
\end{aligned}$$

On the other hand, according to [64], the following inequalities hold:

$$\begin{aligned}
\frac{\widetilde{\psi}\left(\mathbf{f}_{(t+1)}, \mathbf{g}_{(t+1)}\right)}{2R^2} &\leq \frac{\widetilde{\psi}\left(\mathbf{f}_{(t)}, \mathbf{g}_{(t)}\right)}{2R^2} - \left(\frac{\widetilde{\psi}\left(\mathbf{f}_{(t)}, \mathbf{g}_{(t)}\right)}{2R^2}\right)^2 \\
&\leq \left(t + \frac{2R^2}{\widetilde{\psi}\left(\mathbf{f}_{(1)}, \mathbf{g}_{(1)}\right)}\right)^{-1} \qquad (42)
\end{aligned}$$

which implies

$$t \leq \frac{2R^2}{\widetilde{\psi}\left(\mathbf{f}_{(t)}, \mathbf{g}_{(t)}\right)} - \frac{2R^2}{\widetilde{\psi}\left(\mathbf{f}_{(1)}, \mathbf{g}_{(1)}\right)}. \qquad (43)$$

In addition, the following inequality holds naturally:

$$\widetilde{\psi}\left(\mathbf{f}_{(t+\tau)}, \mathbf{g}_{(t+\tau)}\right) \leq \widetilde{\psi}\left(\mathbf{f}_{(t)}, \mathbf{g}_{(t)}\right) - \frac{\tau\epsilon^2}{2} \qquad (44)$$

from which we obtain

$$t \leq 1 + \frac{2}{\epsilon^2}\left(\widetilde{\psi}\left(\mathbf{f}_{(1)}, \mathbf{g}_{(1)}\right) - \widetilde{\psi}\left(\mathbf{f}_{(t)}, \mathbf{g}_{(t)}\right)\right). \qquad (45)$$

By the fact that $\widetilde{\psi}\left(\mathbf{f}_{(t)}, \mathbf{g}_{(t)}\right) \leq \widetilde{\psi}\left(\mathbf{f}_{(1)}, \mathbf{g}_{(1)}\right)$, (43) and (45) can be jointly interpreted as

$$t \leq \min_{\zeta \in (0, \widetilde{\psi}\left(\mathbf{f}_{(1)}, \mathbf{g}_{(1)}\right)]} G(\zeta) \qquad (46)$$

where the function $G(\zeta)$ is expressed as

$$G(\zeta) = 2 + \frac{2R^2}{\zeta} - \frac{2R^2}{\widetilde{\psi}\left(\mathbf{f}_{(1)}, \mathbf{g}_{(1)}\right)} + \frac{2\zeta}{\epsilon^2}. \qquad (47)$$

Now, the derivative of $G(\zeta)$ gives rise to

$$\frac{\partial G(\zeta)}{\partial \zeta} = \frac{-2R^2}{\zeta^2} + \frac{2}{\epsilon^2}. \qquad (48)$$

Therefore, (46) can be further expressed as

$$t \leq \begin{cases} 2 + \frac{4R}{\epsilon} - \frac{2R^2}{\widetilde{\psi}\left(\mathbf{f}_{(1)}, \mathbf{g}_{(1)}\right)}, & \text{if } \widetilde{\psi}\left(\mathbf{f}_{(1)}, \mathbf{g}_{(1)}\right) > R\epsilon \\ 2 + \frac{2\widetilde{\psi}\left(\mathbf{f}_{(1)}, \mathbf{g}_{(1)}\right)}{\epsilon^2}, & \text{otherwise} \end{cases} \qquad (49)$$

from which we get the convergence bound as $t = \mathcal{O}(R/\epsilon)$. Since Algorithm 1 invokes Algorithm 2 for $|\mathcal{K}| - 1$ NNs, each of which has L layers, we finally arrive at the convergence bound of $\mathcal{O}(|\mathcal{K}|LR/\epsilon)$.

References

[1] P. Voigt and A. von dem Bussche, *The EU General Data Protection Regulation (GDPR): A Practical Guide*. Cham, Switzerland: Springer, 2017, [Online]. Available: https://link.springer.com/book/10.1007/978-3-319-57959-7

[2] W. Y. B. Lim et al., "Federated learning in mobile edge networks: A comprehensive survey," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 3, pp. 2031–2063, 3rd Quart., 2020.

[3] S. Niknam, H. S. Dhillon, and J. H. Reed, "Federated learning for wireless communications: Motivation, opportunities, and challenges," *IEEE Commun. Mag.*, vol. 58, no. 6, pp. 46–51, Jun. 2020.

[4] L. U. Khan et al., "Federated learning for edge networks: Resource optimization and incentive mechanism," *IEEE Commun. Mag.*, vol. 58, no. 10, pp. 88–93, Oct. 2020.

[5] T.-C. Chiu, Y.-Y. Shih, A.-C. Pang, C.-S. Wang, W. Weng, and C.-T. Chou, "Semisupervised distributed learning with non-IID data for AIoT service platform," *IEEE Internet Things J.*, vol. 7, no. 10, pp. 9266–9277, Oct. 2020.

[6] J. Zhang and D. Tao, "Empowering things with intelligence: A survey of the progress, challenges, and opportunities in artificial intelligence of things," *IEEE Internet Things J.*, vol. 8, no. 10, pp. 7789–7817, May 2021.

[7] Z. Chang, S. Liu, X. Xiong, Z. Cai, and G. Tu, "A survey of recent advances in edge-computing-powered artificial intelligence of things," *IEEE Internet Things J.*, vol. 8, no. 18, pp. 13849–13875, Sep. 2021.

[8] L. Lin et al., "Understanding the impact on convolutional neural networks with different model scales in AIoT domain," *J. Parallel Distrib. Comput.*, vol. 170, pp. 1–12, Dec. 2022.

[9] "Cisco global cloud index: Forecast and methodology, 2016–2021." 2023. [Online]. Available: https://www.cisco.com/c/en/us/solutions/collateral/service-provider/global-cloud-index-gci/white-paper-c11-738085.html

[10] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. AISTATS*, Apr. 2017, pp. 1273–1282.

[11] S. Wang et al., "Adaptive federated learning in resource constrained edge computing systems," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 6, pp. 1205–1221, Jun. 2019.

[12] D. C. Nguyen et al., "Federated learning meets blockchain in edge computing: Opportunities and challenges," *IEEE Internet Things J.*, vol. 8, no. 16, pp. 12806–12825, Aug. 2021.

[13] P. Roy, S. Sarker, M. A. Razzaque, M. Mamun-or-Rashid, M. M. Hassan, and G. Fortino, "Distributed task allocation in mobile device cloud exploiting federated learning and subjective logic," *J. Syst. Archit.*, vol. 113, Feb. 2021, Art. no. 101972.

[14] R. C. Geyer, T. Klein, and M. Nabi, "Differentially private federated learning: A client level perspective," 2017, *arXiv:1712.07557*.

[15] S. Truex et al., "A hybrid approach to privacy-preserving federated learning," in *Proc. AISec*, Nov. 2019, pp. 1–11.

[16] K. Wei et al., "Federated learning with differential privacy: Algorithms and performance analysis," *IEEE Trans. Inf. Forensics Security*, vol. 15, pp. 3454–3469, 2020.

[17] A. N. Bhagoji, S. Chakraborty, P. Mittal, and S. Calo, "Analyzing federated learning through an adversarial lens," in *Proc. ICML*, Jun. 2019, pp. 634–643.

[18] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How to backdoor federated learning," in *Proc. AISTATS*, Aug. 2020, pp. 2938–2948.

[19] M. Fang, X. Cao, J. Jia, and N. Gong, "Local model poisoning attacks to Byzantine-robust federated learning," in *Proc. USENIX Security*, Aug. 2020, pp. 1605–1622.

[20] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated learning with non-IID data," 2018, *arXiv:1806.00582*.

[21] F. Sattler, S. Wiedemann, K.-R. Müller, and W. Samek, "Robust and communication-efficient federated learning from non-i.i.d. data," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 9, pp. 3400–3413, Sep. 2020.

[22] H. Wang, Z. Kaplan, D. Niu, and B. Li, "Optimizing federated learning on non-IID data with reinforcement learning," in *Proc. IEEE INFOCOM*, Jul. 2020, pp. 1698–1707.

[23] Y. Jiang, J. Konečný, K. Rush, and S. Kannan, "Improving federated learning personalization via model agnostic meta learning," 2019, *arXiv:1909.12488*.

[24] Y. Mansour, M. Mohri, J. Ro, and A. T. Suresh, "Three approaches for personalization with applications to federated learning," 2020, *arXiv:2002.10619*.

[25] C. T. Dinh, N. H. Tran, and T. D. Nguyen, "Personalized federated learning with Moreau envelopes," in *Proc. NIPS*, Dec. 2020, pp. 21394–21405.

[26] N. Shlezinger, M. Chen, Y. C. Eldar, H. V. Poor, and S. Cui, "Federated learning with quantization constraints," in *Proc. IEEE ICASSP*, May 2020, pp. 8851–8855.

[27] A. Reisizadeh, A. Mokhtari, H. Hassani, A. Jadbabaie, and R. Pedarsani, "FedPAQ: A communication-efficient federated learning method with periodic averaging and quantization," in *Proc. AISTATS*, Aug. 2020, pp. 2021–2031.

[28] F. Haddadpour, M. M. Kamani, A. Mokhtari, and M. Mahdavi, "Federated learning with compression: Unified analysis and sharp guarantees," in *Proc. AISTATS*, Apr. 2021, pp. 2350–2358.

[29] A. Andoni, P. Indyk, and R. Krauthgamer, "Earth mover distance over high-dimensional spaces," in *Proc. ACM-SIAM SODA*, vol. 8, Jan. 2008, pp. 343–352.

[30] C. Villani, *Optimal Transport: Old and New*. Heidelberg, Germany: Springer, 2009.

[31] T. Salimans, H. Zhang, A. Radford, and D. Metaxas, "Improving GANs using optimal transport," 2018, *arXiv:1803.05573*.

[32] K. Kandasamy, W. Neiswanger, J. Schneider, B. Poczos, and E. P. Xing, "Neural architecture search with Bayesian optimization and optimal transport," in *Proc. NIPS*, vol. 31, Dec. 2018, pp. 2020–2029.

[33] L. Chizat and F. Bach, "On the global convergence of gradient descent for over-parameterized models using optimal transport," in *Proc. NIPS*, vol. 31, Dec. 2018, pp. 3040–3050.

[34] N. Courty, R. Flamary, D. Tuia, and A. Rakotomamonjy, "Optimal transport for domain adaptation," *IEEE Trans. Pattern Anal. Mach. Intell*, vol. 39, no. 9, pp. 1853–1865, Sep. 2017.

[35] N. Courty, R. Flamary, A. Habrard, and A. Rakotomamonjy, "Joint distribution optimal transportation for domain adaptation," in *Proc. NIPS*, vol. 30, 2017, pp. 3733–3742.

[36] B. B. Damodaran, B. Kellenberger, R. Flamary, D. Tuia, and N. Courty, "DeepJDOT: Deep joint distribution optimal transport for unsupervised domain adaptation," in *Proc. ECCV*, Sep. 2018, pp. 447–463.

[37] M. Cuturi, "Sinkhorn distances: Lightspeed computation of optimal transport," in *Proc. NIPS*, Dec. 2013, pp. 2292–2300.

[38] A. Genevay, G. Peyre, and M. Cuturi, "Learning generative models with Sinkhorn divergences," in *Proc. AISTATS*, vol. 84, Apr. 2018, pp. 1608–1617.

[39] J. Feydy, T. Séjourné, F.-X. Vialard, S.-I. Amari, A. Trouve, and G. Peyré, "Interpolating between optimal transport and MMD using Sinkhorn divergences," in *Proc. AISTATS*, vol. 89, Apr. 2019, pp. 2681–2690.

[40] M. Mozaffari, W. Saad, M. Bennis, and M. Debbah, "Wireless communication using unmanned aerial vehicles (UAVs): Optimal transport theory for hover time optimization," *IEEE Trans. Wireless Commun.*, vol. 16, no. 12, pp. 8052–8066, Dec. 2017.

[41] M. Mozaffari, A. T. Z. Kasgari, W. Saad, M. Bennis, and M. Debbah, "Beyond 5G with UAVs: Foundations of a 3D wireless cellular network," *IEEE Trans. Wireless Commun.*, vol. 18, no. 1, pp. 357–372, Jan. 2019.

[42] D. Wang, J. Tian, H. Zhang, and D. Wu, "Task offloading and trajectory scheduling for UAV-enabled MEC networks: An optimal transport theory perspective," *IEEE Wireless Commun. Lett.*, vol. 11, no. 1, pp. 150–154, Jan. 2022.

[43] L. Wang, H. Zhang, S. Guo, and D. Yuan, "Deployment and association of multiple UAVs in UAV-assisted cellular networks with the knowledge of statistical user position," *IEEE Trans. Wireless Commun.*, vol. 21, no. 8, pp. 6553–6567, Aug. 2022.

[44] N. Guha, A. Talwalkar, and V. Smith, "One-shot federated learning," 2019, *arXiv:1902.11175*.

[45] M. Shin, C. Hwang, J. Kim, J. Park, M. Bennis, and S.-L. Kim, "XOR mixup: Privacy-preserving data augmentation for one-shot federated learning," 2020, *arXiv:2006.05148*.

[46] Y. Zhou, G. Pu, X. Ma, X. Li, and D. Wu, "Distilled one-shot federated learning," 2020, *arXiv:2009.07999*.

[47] Q. Li, B. He, and D. Song, "Practical one-shot federated learning for cross-silo setting," 2020, *arXiv:2010.01017*.

[48] J. Zhang et al., "A practical data-free approach to one-shot federated learning with heterogeneity," 2021, *arXiv:2112.12371*.

[49] R. Song et al., "Federated learning via decentralized dataset distillation in resource-constrained edge environments," 2022, *arXiv:2208.11311*.

[50] M. Kamp et al., "Efficient decentralized deep learning by dynamic model averaging," in *Proc. ECML PKDD*, Sep. 2018, pp. 393–409.

[51] H. Yu, S. Yang, and S. Zhu, "Parallel restarted SGD with faster convergence and less communication: Demystifying why model averaging works for deep learning," in *Proc. AAAI*, vol. 33, Jul. 2019, pp. 5693–5700.

[52] T. Lin, L. Kong, S. U. Stich, and M. Jaggi, "Ensemble distillation for robust model fusion in federated learning," in *Proc. NIPS*, vol. 33, 2020, pp. 2351–2363.

[53] H. Wang, M. Yurochkin, Y. Sun, D. Papailiopoulos, and Y. Khazaeni, "Federated learning with matched averaging," in *Proc. ICLR*, Apr./May 2020, pp. 1–16.

[54] S. P. Singh and M. Jaggi, "Model fusion via optimal transport," in *Proc. NIPS*, vol. 33, Dec. 2020, pp. 22045–22055.

[55] N. H. Tran, W. Bao, A. Zomaya, M. N. H. Nguyen, and C. S. Hong, "Federated learning over wireless networks: Optimization model design and analysis," in *Proc. IEEE INFOCOM*, 2019, pp. 1387–1395.

[56] 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.

[57] M. Chen, Z. Yang, W. Saad, C. Yin, H. V. Poor, and S. Cui, "A joint learning and communications framework for federated learning over wireless networks," *IEEE Trans. Wireless Commun.*, vol. 20, no. 1, pp. 269–283, Jan. 2021.
[58] Y. Rubner, C. Tomasi, and L. Guibas, "A metric for distributions with applications to image databases," in *Proc. ICCV*, Jan. 1998, pp. 59–66.
[59] J. Altschuler, J. Niles-Weed, and P. Rigollet, "Near-linear time approximation algorithms for optimal transport via Sinkhorn iteration," in *Proc. NIPS*, vol. 30, Dec. 2017, pp. 1–18.
[60] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.
[61] M. Gardner and S. Dorling, "Artificial neural networks (the multilayer perceptron)—A review of applications in the atmospheric sciences," *Atmos. Environ.*, vol. 32, no. 14, pp. 2627–2636, 1998.
[62] M. Yurochkin, M. Agarwal, S. Ghosh, K. Greenewald, N. Hoang, and Y. Khazaeni, "Bayesian nonparametric federated learning of neural networks," in *Proc. ICML*, vol. 97, Jun. 2019, pp. 7252–7261.
[63] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2015, *arXiv:1409.1556*.
[64] Y. Nesterov, *Lectures on Convex Optimization*. Cham, Switzerland: Springer Nat., 2018, [Online]. Available: https://link.springer.com/book/10.1007/978-3-319-91578-4

Yi-Han Chiang (Member, IEEE) received the Ph.D. degree from the Graduate Institute of Communication Engineering, National Taiwan University, New Taipei, Taiwan, in 2017.

He was a Postdoctoral Researcher with the Information Systems Architecture Science Research Division, National Institute of Informatics, Tokyo, Japan, in 2018 and 2019. He is currently an Assistant Professor with the Department of Electrical and Electronic Systems Engineering, Osaka Metropolitan University, Osaka, Japan. His research interests include federated learning over wireless networks, age of information, mobile edge computing, and energy efficiency of cellular systems.

Koudai Terai received the B.S. degree from the School of Electrical and Electronic Engineering, Osaka Prefecture University, Sakai, Japan, in 2022. He is currently pursuing the M.S. degree with the Department of Electrical and Electronic Systems Engineering, Osaka Metropolitan University, Osaka, Japan.

His research interests include federated learning over wireless networks and mobile edge computing.

Tsung-Wei Chiang received the Ph.D. degree from the Graduate Institute of Communication Engineering, National Taiwan University, New Taipei, Taiwan, in 2016.

Following his Postdoctoral carrier, he served as a Postdoctoral Fellow and a Researcher with National Taiwan University, in 2018. In 2019, he joined Microsoft Research in Redmond, Redmond, WA, USA, as a Visiting Scholar and a Research Scientist. There, he advanced his knowledge in the fields of data science and deep learning. He currently contributes as a Communications Research Scientist with MediaTek Inc., Hsinchu, Taiwan, where he focuses on next-generation advanced wireless communication technologies. His current research interests encompass a broad spectrum, including wireless communications, information geometry, machine learning, deep learning, statistical signal processing, detection theory, estimation theory, complex networks, and high-performance computing.

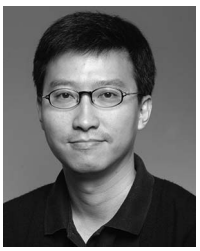
Hai Lin (Senior Member, IEEE) received the B.E. degree from Shanghai Jiaotong University, Shanghai, China, in 1993, the M.E. degree from the University of the Ryukyus, Nishihara, Japan, in 2000, and the Dr.Eng. degree from Osaka Prefecture University, Sakai, Japan, in 2005.

In 2000, he joined the Graduate School of Engineering, Osaka Prefecture University (renamed Osaka Metropolitan University), in 2022, where he is currently a Professor. His research interests include signal processing for communications, wireless communications, and statistical signal processing.

Dr. Lin served several times as a Technical Program Co-Chair for the Signal Processing for Communications Symposium and the Wireless Communications Symposium of the IEEE International Conference on Communications and IEEE Global Communications Conference. He was the Chair of the Signal Processing and Communications Electronics Technical Committee, IEEE Communications Society, from 2015 to 2016. Currently, he is the Chair of the IEEE Communications Society Kansai Chapter. He was an Associate Editor for the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS. He is currently serving as an Associate Editor for the IEEE TRANSACTIONS ON COMMUNICATIONS and the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY.

Yusheng Ji (Fellow, IEEE) received the B.E., M.E., and D.E. degrees in electrical engineering from the University of Tokyo, Tokyo, Japan, in 1984, 1986, and 1989, respectively.

She joined the National Center for Science Information Systems, Japan, in 1990. She is currently a Professor and the Director of the Information Systems Architecture Science Research Division, National Institute of Informatics, Tokyo, Japan, and a Professor with the Graduate University for Advanced Studies, Hayama, Japan. Her research interests include resource management in wireless networks and mobile computing.



John C. S. Lui (Fellow, IEEE) received the Ph.D. degree in computer science from the University of California at Los Angeles, Los Angeles, CA, USA, in 1991.

He is currently the Choh-Ming Li Chair Professor with the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong. His current research interests include quantum Internet, machine learning, online learning theories, future Internet architectures and protocols, network economics, and network/system security.

Dr. Lui has received various departmental teaching awards and the CUHK Vice Chancellor's Exemplary Teaching Award. He is an Elected Member of the IFIP WG 7.3, a Fellow of ACM, a Senior Research Fellow of the Croucher Foundation, and the RGC Senior Research Fellow.