Continual Learning of Generative Models With Limited Data: From Wasserstein-1 Barycenter to Adaptive Coalescence

Mehmet Dedeoglu, *Student Member, IEEE*, Sen Lin, *Member, IEEE*, Zhaofeng Zhang, *Student Member, IEEE*, and Junshan Zhang, *Fellow, IEEE*

***Abstract*— Learning generative models is challenging for a network edge node with limited data and computing power. Since tasks in similar environments share a model similarity, it is plausible to leverage pretrained generative models from other edge nodes. Appealing to optimal transport theory tailored toward Wasserstein-1 generative adversarial networks (WGANs), this study aims to develop a framework that systematically optimizes continual learning of generative models using local data at the edge node while exploiting adaptive coalescence of pretrained generative models. Specifically, by treating the knowledge transfer from other nodes as Wasserstein balls centered around their pretrained models, continual learning of generative models is cast as a constrained optimization problem, which is further reduced to a Wasserstein-1 barycenter problem. A two-stage approach is devised accordingly: 1) the barycenters among the pretrained models are computed offline, where displacement interpolation is used as the theoretic foundation for finding adaptive barycenters via a "recursive" WGAN configuration and 2) the barycenter computed offline is used as metamodel initialization for continual learning, and then, fast adaptation is carried out to find the generative model using the local samples at the target edge node. Finally, a weight ternarization method, based on joint optimization of weights and threshold for quantization, is developed to compress the generative model further. Extensive experimental studies corroborate the effectiveness of the proposed framework.**

***Index Terms*— Continual learning, generative adversarial networks (GANs), optimal transport theory, Wasserstein barycenters.**

I. Introduction

THE past few years have witnessed an explosive growth of the Artificial-Intelligence-of-Things (AIoT) devices at the network edge. On the grounds that the cloud has abundant computing resources, the conventional method for artificial intelligence (AI) at the network edge is that the cloud trains the AI models with the data uploaded from edge devices and then pushes the models back to the edge for on-device inference (e.g., Google Edge TPU). However, an emerging view is that this approach suffers from overwhelming communication overhead incurred by the data transmission from edge devices to the cloud, as well as potential privacy leakage. It is, therefore, of great interest to represent edge data using generative models because they require a smaller number of parameters than the data volume, and it is much more parsimonious compared to sending the edge data to the cloud; furthermore, they can also help to preserve data privacy [1], [2]. It is clear that continual learning fits naturally in edge applications consisting of multiple stages with data collection and learning model development for the edge. Taking a forward-looking view, this study focuses on the continual learning of generative models for edge intelligence (EI) to achieve efficient storage and accurate representation of edge data.

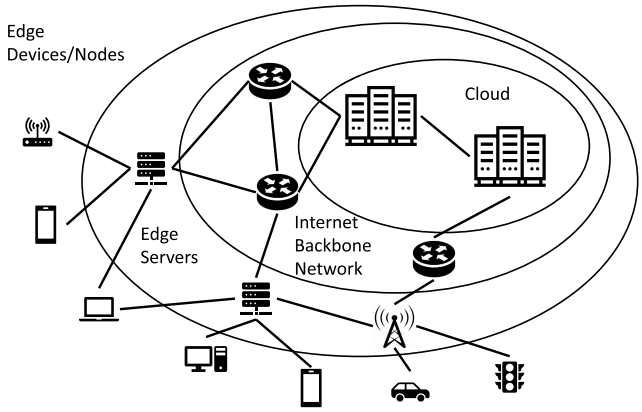


Fig. 1. Illustration of EI.

AI is an approach to build intelligent machines to perform tasks as humans do. In this regard, AI systems demonstrate behaviors associated with humans such as learning, problem-solving, representation, perception, and creativity. In parallel with AI, EI is a paradigm to build a set of *collaborative* intelligent machines at the edge to perform tasks as humans do [3]. Simply, EI aims to gain collective AI insights by leveraging limited resources at the network edge (see Fig. 1). Specifically, there are a variety of edge devices/nodes and

Manuscript received 4 September 2021; revised 17 March 2022, 21 July 2022, and 22 February 2023; accepted 24 February 2023. This work was supported in part by NSF Grant CNS-2203239, Grant CNS-2203412, and Grant CCSS-2121222. *(Corresponding author: Mehmet Dedeoglu.)*

Mehmet Dedeoglu and Zhaofeng Zhang are with the School of Electrical, Computer and Energy Engineering, Arizona State University, Tempe, AZ 85281 USA (e-mail: mdedeogl@asu.edu).

Sen Lin is with the AI-EDGE Institute, The Ohio State University, Columbus, OH 43210 USA.

Junshan Zhang is with the Department of Electrical and Computer Engineering, University of California at Davis, Davis, CA 95616 USA.

This article has supplementary material provided by the authors and color versions of one or more figures available at https://doi.org/10.1109/TNNLS.2023.3251096.

Digital Object Identifier 10.1109/TNNLS.2023.3251096

edge servers, ranging from self-driving cars to robots and from 5G base station servers to mobile phones. Edge applications involve computational tasks that require the processing of edge data to meet users' requests. In contrast to cloud computing, in which data centers away from edge nodes with access to huge datasets are responsible for processing users' requests, edge applications process limited data samples at the network edge to mitigate communication delay and bandwidth usage. For instance, a general consensus is that self-driving cars need to be aware of their surroundings and traffic to safely cruise through the traffic and to arrive at their destinations. To this end, self-driving cars can connect with other cars via vehicle-to-vehicle (V2V) and vehicle-to-everything (V2x) communications to learn about traffic, make predictions, and take actions, which would be infeasible to use conventional methods, such as cloud computing, due to large amounts of delays in communication between the car and the cloud. EI envisions developing solutions to these scalability issues and reducing communication overhead by carrying out the computational operations at the network edge.

In regards to scalability issues, deep generative models can parameterize high-dimensional data samples at edge nodes effectively, but it is often not feasible for a single edge node to train a deep generative model from scratch, which would otherwise require humongous training data and high computational power [4], [5]. Many edge AI applications (e.g., autonomous driving, smart robots, safety-critical health applications, and virtual reality), therefore, require EI and continual learning capability via fast adaptation with local data samples and utilizing the prior knowledge over other edge devices to adapt to dynamic application environments [6], [7].

A general consensus is that learning tasks across different edge nodes often share a model similarity. For instance, different robots may perform similar coordination behaviors according to environment changes. With this sight, we advocate that the pretrained generative models from other edge nodes are utilized to speed up the learning at a given edge node and seek to answer the following critical questions.

1) What is the right abstraction of knowledge from multiple pretrained models for continual learning?
2) How can an edge server leverage this knowledge for continual learning of a generative model?

The key to answering the first question lies in the efficient model fusion of multiple pretrained generative models. A common approach is the *ensemble method* [8], where the outputs of different models are aggregated to improve the prediction performance. However, this requires the edge server to maintain all the pretrained models and run each of them, which would outweigh the resources available at edge servers. Another way for model fusion is direct *weight averaging* [9]. Because the weights in neural networks are highly redundant and no one-to-one correspondence exists between the weights of two different neural networks, this method is known to yield poor performance even if the networks represent the same function of the input. As for the second question, transfer learning is a promising learning paradigm where an edge node incorporates the knowledge from the cloud or another node with its local training samples. [4], [5]. Recent work on transferring generative adversarial networks (GANs) [5] proposed several transfer configurations to leverage pretrained GANs to accelerate the learning process. However, since the transferred GAN is used only as initialization, transferring GANs suffers from catastrophic forgetting. Specifically, catastrophic forgetting may occur when a pretrained learning model is further trained using another dataset. Overtraining the model with new data causes the model to be overtuned to new data and forget the features learned from previous data [10]. To tackle these challenges, this work aims to develop a framework that explicitly optimizes the continual learning of generative models for the edge, based on the adaptive coalescence of pretrained generative models from other edge nodes, using optimal transport theory tailored toward GANs.

To mitigate the mode collapse problem due to the vanishing gradients, multiple GAN configurations have been proposed based on the Wasserstein-p metric W_p, including the Wasserstein-1 distance [11] and the Wasserstein-2 distance [12], [13]. Despite that Wasserstein-2 GANs are analytically tractable, the corresponding implementation often requires regularization and is often outperformed by the Wasserstein-1 GAN (W1GAN). With this insight, in this article, we focus on the W1GAN (WGAN refers to W1GAN throughout).

A. Basic Setting

Specifically, we consider a setting where a target edge node, denoted Node 0, learns a generative model for representing its underlying distribution. Training a WGAN is intimately related to finding a distribution minimizing the Wasserstein distance from the underlying distribution μ_0 [14]. In practice, an edge node has a limited number of samples with empirical distribution $\hat{\mu}_0$, which is distant from μ_0. A naive approach is to train a WGAN based on the limited local samples only, which can be captured via the optimization problem given by $\min_{\nu\in\mathcal{P}} W_1(\nu, \hat{\mu}_0)$, with $W_1(\cdot,\cdot)$ being the Wasserstein-1 distance between two distributions and $\mathcal{P}$ being the distribution space. The best possible outcome of solving this optimization problem can generate a distribution very close to $\hat{\mu}_0$, which, however, could still be far away from the true distribution μ_0. It is clear that training a WGAN simply based on the limited local samples at an edge node would not be able to obtain a generative model representing the true distribution μ_0.

As alluded to earlier, learning tasks across different edge nodes may share a similarity, e.g., limited local samples at Node 0 should be similar to the data samples from other edge nodes. To facilitate the continual learning at Node 0, pretrained generative models from other related edge nodes can be leveraged via knowledge transfer. Without loss of generality, we assume that there is a set $\mathcal{K}$ of K edge nodes with pretrained generative models. Since one of the most appealing benefits of WGANs is the ability to continuously estimate the Wasserstein distance during training [11], we assume that the knowledge transfer from Node k to Node 0 is in the form of a Wasserstein ball with radius η_k centered around its pretrained generative model μ_k at Node k for $k = 1, \ldots, K$.

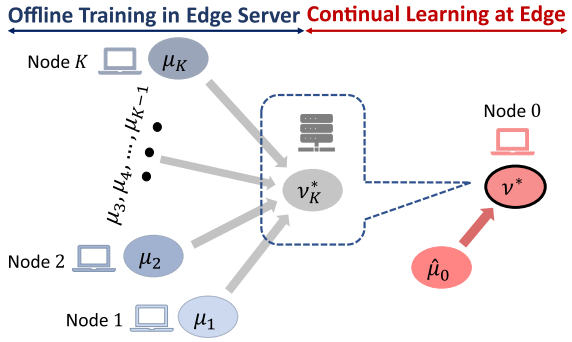


Fig. 2. Continual learning of generative models based on a coalescence of pretrained generative models $\{\mu_k, k = 1, \ldots, K\}$ and the local dataset at Node 0 (denoted by $\hat{\mu}_0$).

Intuitively, radius η_k represents the relevance (hence utility) of the knowledge transfer, and the smaller it is, the more informative the corresponding Wasserstein ball is. Building on this knowledge transfer model, we treat the continual learning problem at Node 0 as the coalescence of K generative models and empirical distribution $\hat{\mu}_0$, and cast it as the constrained optimization problem

$$\min_{\nu \in \mathcal{P}} W_1\big(\nu, \hat{\mu}_0\big), \ \text{s.t.} \ \ W_1(\nu, \mu_k) \leq \eta_k \quad \forall k \in \mathcal{K}. \tag{1}$$

Observe that the constraints in problem (1) dictate that the optimal coalesced generative model, denoted by ν^*, lies within the intersection of K Wasserstein balls (centered around $\{\mu_k\}$), exploiting the knowledge transfer systematically. It is worth noting that the optimization problem (1) can be extended to other distance functionals, e.g., the Jensen–Shannon divergence.

B. Main Contributions

The contributions of this work are summarized as follows.

1) We propose a systematic framework to enable continual learning of generative models via adaptive coalescence of pretrained generative models from other edge nodes and local samples at Node 0. In particular, by treating the knowledge transferred from each node as a Wasserstein ball centered around its local pretrained generative model, we cast the problem as a constrained optimization problem, which optimizes the continual learning of generative models.
2) Applying Lagrangian relaxation to (1), we reduce the optimization problem to finding a Wasserstein-1 barycenter of $K+1$ probability measures, among which K of them are pretrained generative models, and the last one is the empirical distribution (not a generative model though) corresponding to local data samples at Node 0. We propose a *barycentric fast adaptation approach* to efficiently solve the barycenter problem, where the barycenter ν_K^* for the K pretrained generative models is found recursively offline in edge servers, and then, the barycenter between the empirical distribution $\hat{\mu}_0$ of Node 0 and ν_K^* is solved via fast adaptation at Node 0 (see Fig. 2). A salient feature in this barycentric approach is that the generative replay, enabled by pretrained GANs, is used to annihilate catastrophic forgetting.
3) It is known that the Wasserstein-1 barycenter is notoriously difficult to analyze, partly because of the existence of infinitely many minimizers of the Monge problem. Appealing to the optimal transport theory, we use displacement interpolation as the theoretic foundation to devise recursive algorithms for finding adaptive barycenters, which ensures that the resulting barycenters lie in the baryregion.
4) From the implementation perspective, we introduce a "recursive" WGAN configuration, where a two-discriminator WGAN is used per recursive step to find adaptive barycenters sequentially. Then, the resulting barycenter in off-line training is treated as the metamodel initialization, and fast adaptation is carried out to find the generative model using the local samples at Node 0. A weight ternarization method, based on joint optimization of weights and threshold for quantization, is developed to compress the generative model and enable efficient edge learning. Extensive experiments corroborate the efficacy of the proposed framework for fast edge learning of generative models, and the W_1-based recursive WGAN configuration performs better than the W_2^2-based one.

C. Use Cases

To further illustrate the motivation behind the problem (1), we provide two applications in what follows.

1) Privacy is vital in predictive health applications of edge learning. In a pet disease prediction task, a user aims to learn her/his pet's disease by sending the pet's photographs to an edge server. Since photographs may contain private information, it is desired to eliminate unnecessary information for the detection of the disease from photos. A personalized WGAN model can generate unique photographs of the user's pet with only the necessary information and prevent the leakage of private information. However, a personalized WGAN model cannot be trained alone on the user's phone/laptop because it would require a lot of data samples and processing power. Instead, the user can leverage a barycenter WGAN model from the edge server, which is trained on healthy and unhealthy pet images offline, to train the user's personalized WGAN model, so as to represent the user's pet's health condition without revealing any private information. The training process can be done on the user's device as barycentric fast adaptation requires much less computational power. Alternative approaches, such as differential privacy, can provide similar privacy guarantees [15] but suffer from scalability issues in transferring photographs to an edge server. Training a barycenter or a personalized WGAN model via fast adaptation only requires the transfer of generative models and, hence, mitigates data transfer amount while preserving privacy.
2) Data storage at the edge is gaining popularity due to low delay access to data and ease of local data

management. Even though edge servers are more capable than user edge devices/nodes, they still have much less memory and computational power possessed by cloud data centers. To overcome these challenges, barycenter WGAN models can be used to represent datasets for applications requiring less precision. To exemplify, a single barycenter WGAN model could represent the information in hundreds of cat image databases instead of storing the same information in hundreds of WGAN models with minimum accuracy loss at the edge server. This accuracy loss can later be recovered quickly if the barycentric fast adaptation is applied to the barycenter WGAN model with the desired database.

The rest of this article is organized as follows. In Section II, related works about EI, optimal transport theory, and WGAN literature are discussed. In Section III, the adaptive coalescence of Wasserstein-1 generative models is mathematically modeled and analyzed, and a practical algorithm is proposed. The recursive barycentric fast adaptation algorithm is explained in detail in Section IV. In Section V, a variety of numerical experiments are performed and demonstrated. Section VI articulates the computational complexity of barycentric fast adaptation algorithm and how it ranks among its counterparts. Finally, the conclusions are discussed in Section VII.

II. Related Work

Optimal transport theory has recently been studied for deep learning applications (see [16], [17], and [18]). Agueh and Carlier [19] have developed an analytical solution to the Wasserstein barycenter problem. Aiming to numerically solve the Wasserstein barycenter problem, Cuturi and Doucet [20] proposed smoothing through entropy regularization for the discrete setting based on linear programming. Srivastava et al. [21] employed posterior sampling algorithms in studying Wasserstein barycenters, and Anderes et al. [22] characterized Wasserstein barycenters for the discrete setting (cf. [23]). Most recent studies leveraged optimal transport theory in domain adaptation [24], cross-domain alignment [25], reinforcement learning [26], and developed further approximations to the Wasserstein distance, e.g., Sinkhorn divergence [27]. Nguyen et al. [24] proposed the use of optimal transport theory to quantify the distance between embedded distributions of source and target data in the joint space. By minimizing the distance, TIDOT can mitigate both data and label shifts between different domains. Chen et al. [25] modeled the cross-domain alignment problem as a graph-matching problem and leveraged Wasserstein and Gromov–Wasserstein distances for node and edge matching, respectively. Chen et al. [26] employ optimal transport theory to solve the difficulties associated with the application of reinforcement learning in language generation. Li et al. [27] studied the Sinkhorn divergence and expanded its applicability to problems with complex data with a nonlinear structure. In particular, the authors extended the analytical results in the Euclidean space to the reproducing kernel Hilbert space.

GANs [28] have recently emerged as a powerful deep learning tool for obtaining generative models. Arjovsky et al. [11] have introduced the Wasserstein metric in GANs, which can help mitigate the vanishing gradient issue to avoid mode collapse. Using optimal transport theory, recent advances in Wasserstein GANs have shed light on understanding generative models. Leygonie et al. [12], Liu et al. [13], and Korotin et al. [35] proposed transport theory-based GAN configurations using quadratic Wasserstein-2 cost. In contrast, for the W1GAN, the discriminator may utilize one of the infinitely many transport maps from underlying empirical data distribution to the generative model [17], [18], and it remains open to decipher the relation between the model training and transport maps. Along a different line, a variety of techniques have been proposed for more robust training of GANs [4], [30], [31].

Pushing the AI frontier to the network edge for achieving EI has recently emerged as the marriage of AI and edge computing [3]. There are significant challenges since AI model training generally requires tremendous resources that greatly outweigh the capability of resource-limited edge nodes. To address this, various approaches have been proposed in the literature, including model compression [32], knowledge transfer learning [33], [34], hardware acceleration [35], and collaboration-based methods [36], [37]. Shafiee et al. [32] proposed an efficient deep neural network of size smaller than 2.4 MB to solve classification problems. However, the proposed neural network is designed to only work on small number of classes. Osia et al. [33] designed a hybrid architecture where local devices and cloud systems collaborate on running a deep neural network that has previously been fine-tuned on the cloud. To achieve privacy, the proposed technique extracts the minimum necessary information from the user to transfer to the cloud. Venkataramani et al. [35] proposed an energy-efficient powerful server architecture to train deep neural networks, which focused on improving core utilization, memory bandwidth, and reducing synchronization overheads to achieve its efficiency. Gao et al. [34] proposed an adaptive neural network architecture, in which network architecture adapts to a new task. Continual learning with an efficient architecture search technique learns which neurons to leverage for the new task. In regards to privacy, differential privacy can offer privacy guarantees on distributed learning systems that include data collection from users [15]. However, EI is not only concerned with privacy but also with scalability problems arising from the large-scale data at the edge. Therefore, this study adopts a GAN approach to overcome scalability and privacy concerns in EI applications.

Different from existing studies, this work focuses on the continual learning of generative models at the edge nodes. Rather than learning the new model from scratch, continual learning aims to design algorithms leveraging knowledge transfer from pretrained models to the new learning task [38], assuming that the training data of previous tasks are unavailable for the new task. Generative replay is gaining more attention where synthetic samples corresponding to earlier tasks are obtained with a generative model and replayed in model training for the new task to mitigate forgetting [39]. In this work, by learning generative models via the adaptive coalescence of pretrained generative models from other nodes,

the proposed "recursive" WGAN configuration facilitates fast edge learning in a continual manner, which can be viewed as an innovative integration of a few key ideas in continual learning, including the replay method [40], [41], which generates pseudosamples using generative models, and the regularization-based methods [42], which set the regularization for the model learning based on the learned knowledge from previous tasks [43].

III. Adaptive Coalescence of Wasserstein-1 Generative Models

Next, we first recast problem (1) as a Wasserstein barycenter problem. Then, we propose a two-stage recursive algorithm, characterize the geometric properties of geodesic curves therein, and use displacement interpolation as the foundation to devise recursive algorithms for finding adaptive barycenters.

A. Wasserstein-1 Barycenter Formulation via Lagrangian Relaxation

Observe that the Lagrangian for (1) is given as follows:

$$\mathcal{L}(\{\lambda_k\}, \nu) = W_1\big(\nu, \hat{\mu}_0\big) + \sum_{k=1}^{K} \lambda_k W_1(\nu, \mu_k) - \sum_{k=1}^{K} \lambda_k \eta_k \quad (2)$$

where $\{\lambda_k \geq 0\}_{1:K}$ are the Lagrange multipliers. Based on [44], problem (1) can be solved by using the following Lagrangian relaxation with $\lambda_k = (1/\eta_k), \forall k \in \mathcal{K}$, and $\lambda_0 = 1$:

$$\min_{\nu \in \mathcal{P}} \sum_{k=1}^{K} (1/\eta_k) W_1(\nu, \mu_k) + W_1\big(\nu, \hat{\mu}_0\big). \quad (3)$$

It is shown in [45] that the selection $\lambda_k = (1/\eta_k), \forall k \in \mathcal{K}$, ensures the same levels of robustness for (3) and (1). Intuitively, such a selection of $\{\lambda_k\}_{0:K}$ strikes a right balance, in the sense that larger weights are assigned to the knowledge transfer models (based on the pretrained generative models $\{\mu_k\}$) from the nodes with higher relevance, captured by smaller Wasserstein-1 ball radii. For given $\{\lambda_k \geq 0\}$, (3) turns out to be a Wasserstein-1 barycenter problem (cf. [22] and [26]), with the new complication that $\hat{\mu}_0$ is an empirical distribution corresponding to local samples at Node 0. Since $\hat{\mu}_0$ is not a generative model per se, its coalescence with other K general models is challenging.

B. Two-Stage Adaptive Coalescence Approach for Wasserstein-1 Barycenter Problem

Based on (3), we take a two-stage approach to enable efficient learning of the generative model at edge Node 0. The primary objective of Stage I is to find the barycenter for K pretrained generative models $\{\mu_1, \ldots, \mu_K\}$. It is clear that the ensemble method would not work well due to the required memory and computational resources. With this insight, we develop a recursive algorithm for the adaptive coalescence of pretrained generative models. In Stage II, the resulting barycenter solution in Stage I is treated as the model initialization and is further trained using the local samples at Node 0. We propose that the off-line training in Stage I is performed in the edge server, and the fast adaptation in Stage II is carried out at the edge server (in the same spirit as the model update of Google EDGE TPU), as outlined in the following.

Stage I (Find the Barycenter of K Pretrained Generative Models Across K Edge Nodes Offline): Mathematically, this entails the solution to the following problem:

$$\min_{\nu \in \mathcal{P}} \sum_{k=1}^{K} \frac{1}{\eta_k} W_1(\nu, \mu_k). \quad (4)$$

To reduce computational complexity, we propose the following recursive algorithm: take μ_1 as an initial point, i.e., $\nu_1^* = \mu_1$, and let ν_{k-1}^* denote the barycenter of $\{\mu_i\}_{1:k-1}$ obtained at iteration $k-1$ for $k = 2, \ldots, K$. Then, at each iteration k, a new barycenter ν_k^* is solved between the barycenter ν_{k-1}^* and the pretrained generative model μ_k.

Stage II (Fast Adaptation to Find the Barycenter Between ν_K^* and the Local Dataset at Node 0): Given the solution ν_K^* obtained in Stage I, we subsequently solve the following problem: $\min_{\nu \in \mathcal{P}} W_1(\nu, \hat{\mu}_0) + W_1(\nu, \nu_K^*)$. By taking ν_K^* as the model initialization, fast adaptation based on local samples is used to learn the generative model at Node 0.

C. From Displacement Interpolation to Adaptive Barycenters

As noted above, in practical implementation, the W1-GAN often outperforms Wasserstein-p GANs ($p > 1$). However, the Wasserstein-1 barycenter is notoriously difficult to analyze due to the nonuniqueness of the minimizer to the Monge problem [18]. Appealing to optimal transport theory, we next characterize the performance of the proposed two-stage recursive algorithm for finding the Wasserstein-1 barycenter of pretrained generative models $\{\mu_k, k = 1, \ldots, K\}$ and the local dataset at Node 0 by examining the existence of the barycenter and characterizing its geometric properties based on geodesic curves.

The seminal work [46] has established the existence of geodesic curves between any two distribution functions σ_0 and σ_1 in the p-Wasserstein space, $\mathcal{P}_p$, for $p \geq 1$. It is shown in [18] that there are infinitely many minimal geodesic curves between σ_0 and σ_1, when $p = 1$. This is best illustrated in N-dimensional Cartesian space, where the minimal geodesic curves between $\varsigma_0 \in \mathbb{R}^N$ and $\varsigma_1 \in \mathbb{R}^N$ can be parameterized as follows: $\varsigma_t = \varsigma_0 + s(t)(\varsigma_1 - \varsigma_0)$,, where $s(t)$ is an arbitrary function of t, indicating that there are infinitely many minimal geodesic curves between ς_0 and ς_1. This is in stark contrast to the case $p > 1$ where there is a unique geodesic between ς_0 and ς_1. In a similar fashion, there exist infinitely many transport maps, T_0^1, from σ_0 to σ_1 when $p = 1$. For convenience, let $C(\sigma_0, \sigma_1)$ denote an appropriate transport cost function quantifying the minimum cost to move a unit mass from σ_0 to σ_1. It had been shown in [18] that, when $p = 1$, two interpolated distribution functions on two distinct minimal curves may have a nonzero distance, i.e., $C(\hat{T}_0^1 \# \sigma_0, \tilde{T}_0^1 \# \sigma_0) \geq 0$, where # denotes the push-forward operator, thus yielding multiple minimizers to (4). For convenience, define $\mathcal{F} := \hat{\mu}_0 \cup \{\mu_k\}_{1:K}$.

Definition 1 (Baryregion): Let $g_t(\mu_k, \mu_\ell)_{0\le t\le 1}$ be any minimal geodesic curve between any pair $\mu_k, \mu_\ell \in \mathcal{F}$, and define the union $\mathcal{R} := \bigcup_{k=1}^{K} \bigcup_{\ell=k+1}^{K+1} g_t(\mu_k, \mu_\ell)_{0\le t\le 1}$. The baryregion $\mathcal{B}_\mathcal{R}$ is given by $\mathcal{B}_\mathcal{R} = \bigcup_{\sigma\in\mathcal{R}} \bigcup_{\varpi\in\mathcal{R}, \varpi\neq\sigma} g_t(\sigma, \varpi)_{0\le t\le 1}$.

Intuitively, $\mathcal{B}_\mathcal{R}$ encapsulates all possible interpolations through distinct geodesics between any two distributions in $\mathcal{F}$. Since each geodesic has finite length, $\mathcal{B}_\mathcal{R}$ defines a bounded set in $\mathcal{P}_1$. Next, we restate in Lemma 1 the renowned *displacement interpolation* result [46], which sets the foundation for each recursive step in finding a barycenter in our proposed two-stage algorithm. In particular, Lemma 1 leads to the fact that the barycenter ν^* resides in $\mathcal{B}_\mathcal{R}$.

Lemma 1 (Displacement Interpolation [47]): Let $C(\sigma_0, \sigma_1)$ denote the minimum transport cost between σ_0 and σ_1, and suppose that $C(\sigma_0, \sigma_1)$ is finite for $\sigma_0, \sigma_1 \in \mathcal{P}(\mathcal{X})$. Assume that $C(\sigma_s, \sigma_t)$, the minimum transport cost between σ_s and σ_t for any $0 \le s \le t \le 1$, is continuous. Then, the following holds true for any given continuous path $g_t(\sigma_0, \sigma_1)_{0\le t\le 1}$:

$$C\left(\sigma_{t_1}, \sigma_{t_2}\right) + C\left(\sigma_{t_2}, \sigma_{t_3}\right) = C\left(\sigma_{t_1}, \sigma_{t_3}\right),\ 0 \le t_1 \le t_2 \le t_3 \le 1.$$

In the adaptive coalescence algorithm, the kth recursion defines a baryregion, $\mathcal{B}_{\{\nu^*_{k_1}, \mu_k\}}$, consisting of geodesics between the barycenter ν^*_{k-1} found in the $(k-1)$th recursion and generative model μ_k. It is clear that $\mathcal{B}_{\{\nu^*_k, \mu_k\}} \subset \mathcal{B}_\mathcal{R}$. Viewing each recursive step in the above two-stage algorithm as adaptive displacement interpolation, we have the main result on the geodesics and the geometric properties regarding ν^* and $\{\nu^*_k\}_{1:K}$.

Proposition 1 Displacement Interpolation for Adaptive Barycenters: The adaptive barycenter, ν^*_k, obtained at the output of the kth recursive step in Stage I, is a displacement interpolation between ν^*_{k-1} and μ_k and resides inside $\mathcal{B}_\mathcal{R}$. Furthermore, the final barycenter ν^* resulting from Stage II of the recursive algorithm resides inside $\mathcal{B}_\mathcal{R}$.

Remark 1: It is worth noting that different orders for adaptive coalescence may lead to different final barycentric W1GAN models although the resulting ν^* resides in $\mathcal{B}_\mathcal{R}$ always. Had the quadratic Wasserstein cost W_2^2 been used, the final barycenter ν^* would be unique in $\mathcal{B}_\mathcal{R}$. However, the corresponding implementation using W_2^2 poses significant challenges [13], [29] and is often outperformed by W1GAN, which we will elaborate further in Section IV and Appendixes B and C.

IV. Recursive WGAN Configuration for Continual Learning

Based on the above theoretic results on adaptive coalescence via Wasserstein-1 barycenters, we next turn attention to the implementation of computing adaptive barycenters. Notably, assuming the knowledge of accurate empirical distribution models on discrete support, Cuturi and Doucet [20] introduce a powerful linear program (LP) to compute Wasserstein-p barycenters, but the computational complexity of this approach is excessive. In light of this, we propose a WGAN-based configuration for finding the Wasserstein-1 barycenter, which, in turn, enables fast learning of generative models based on

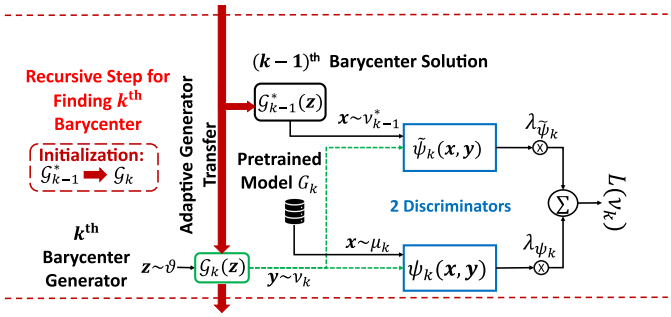


Fig. 3. Two-discriminator WGAN for efficient learning of the kth barycenter generator in off-line training, where x denotes the synthetic data generated from pretrained models [see Section IV-1 and (6)].

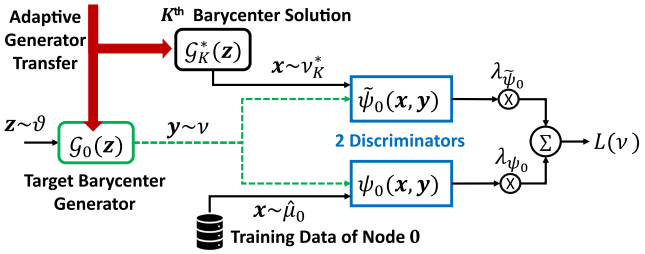


Fig. 4. Fast adaptation for learning a generative model at Node 0 [see Section IV-2 and (7)].

the coalescence of pretrained models. Specifically, (3) can be rewritten as

$$\min_{G}\max_{\{\varphi_k\}_{0:K}} \mathbb{E}_{x\sim\hat{\mu}_0}[\varphi_0(x)] - \mathbb{E}_{z\sim\vartheta}[\varphi_0(G(z))] + \sum_{k=1}^{K} \frac{1}{\eta_k}\left\{\mathbb{E}_{x\sim\mu_k}[\varphi_k(x)] - \mathbb{E}_{z\sim\vartheta}[\varphi_k(G(z))]\right\} \tag{5}$$

where G represents the generator and $\{\varphi_k\}_{0:K}$ are 1−Lipschitz functions for discriminator models, respectively (see Appendix D.C in the Supplementary Material for detailed derivation). Observe that the optimal generator DNN G^* facilitates the barycenter distribution ν^* at its output. We note that the multidiscriminator WGAN configuration has recently been developed [30], [48] by using a common latent space to train multiple discriminators, so as to improve stability. In stark contrast, in this work, distinct generative models from multiple nodes are exploited to train different discriminators, aiming to learn distinct transport plans among generative models.

A naive approach is to implement the above multidiscriminator WGAN in a one-shot manner where the generator and $K+1$ discriminators are trained simultaneously, which, however, would require overwhelming computation power and memory. To enable efficient training, we use the proposed two-stage algorithm and develop a *recursive* WGAN configuration (see Algorithms 1 and 2) to sequentially compute: 1) the barycenter ν^*_K for the off-line training in the edge servers, as shown in Fig. 3 and 2) the barycenter ν^* for the fast adaptation at the target edge node, as shown in Fig. 4. The analytical relation between *one-shot* and *recursive* barycenters has been studied for Wasserstein-2 distance, and sufficient conditions for their equivalence are presented in [49], which, would not suffice for the Wasserstein-1 distance, because of the existence of multiple Wasserstein-1 barycenters. Proposition 1 shows that any barycenter solution to a recursive algorithm resides inside a baryregion, which can be viewed as the counterpart for

Algorithm 1 Off-Line Training to Solve the Barycenter of K Pretrained Generative Models

Inputs: K pre-trained generator-discriminator pairs $\{(G_k, D_k)\}_{1:K}$ of corresponding source nodes $k \in \mathcal{K}$, noise prior $\vartheta(z)$, the batch size m, learning rate α

Set $\mathcal{G}_1^* \leftarrow G_1$, $\tilde{\psi}_1^* \leftarrow$ rand() or $\tilde{\psi}_1^* \leftarrow D_1$; **//Barycenter initialization**

for iteration $k = 2, \ldots, K$ **do**

  Set $\mathcal{G}_k \leftarrow \mathcal{G}_{k-1}^*$, $\tilde{\psi}_k \leftarrow$ rand(), $\psi_k \leftarrow$ rand() (or $\tilde{\psi}_k \leftarrow \psi \in \{\tilde{\psi}_{k-1}^*, \psi_{k-1}^*\}$, $\psi_k \leftarrow D_k$) and choose $\lambda_{\tilde{\psi}_k}$, λ_{ψ_k};

  while generator $\mathcal{G}_k$ has not converged **do**

    Sample batches of prior samples $\{z^{(i)}\}_{i=1}^m$, $\{z_{\tilde{\psi}_k}^{(i)}\}_{i=1}^m$, $\{z_{\psi_k}^{(i)}\}_{i=1}^m$ independently from prior $\vartheta(z)$;

    Generate synthetic data batches $\{x_{\tilde{\psi}_k}^{(i)}\}_{i=1}^m \sim \nu_{k-1}^*$ and $\{x_{\psi_k}^{(i)}\}_{i=1}^m \sim \mu_k$ by passing $\{z_{\tilde{\psi}_k}^{(i)}\}_{i=1}^m$ and $\{z_{\psi_k}^{(i)}\}_{i=1}^m$ through $\mathcal{G}_{k-1}^*$ and G_k, respectively;

    Compute gradients $g_{\tilde{\psi}_k}$ and g_{ψ_k}: $\{g_\omega \leftarrow \lambda_\omega \nabla_\omega \frac{1}{m} \sum_{i=1}^m [\omega(x_\omega^{(i)}) - \omega(\mathcal{G}_k(z^{(i)}))]\}_{\omega=\tilde{\psi}_k,\psi_k}$;

    Update both discriminators ψ_k and $\tilde{\psi}_k$: $\{\omega \leftarrow \omega + \alpha \cdot \text{Adam}(\omega, g_\omega)\}_{\omega=\psi_k,\tilde{\psi}_k}$;

    Compute gradient $g_{\mathcal{G}_k} \leftarrow -\nabla_{\mathcal{G}_k} \frac{1}{m} \sum_{i=1}^m [\lambda_{\psi_k} \psi_k(\mathcal{G}_k(z^{(i)})) + \lambda_{\tilde{\psi}_k} \tilde{\psi}_k(\mathcal{G}_k(z^{(i)}))]$;

    Update generator $\mathcal{G}_k$: $\mathcal{G}_k \leftarrow \mathcal{G}_k - \alpha \cdot \text{Adam}(\mathcal{G}_k, g_{\mathcal{G}_k})$;

  Assign $\mathcal{G}_k^* \leftarrow \mathcal{G}_k$

return generator $\mathcal{G}_K^*$ for barycenter ν_K^*, discriminators $\tilde{\psi}_K^*$, ψ_K^*.

the *one-shot* solution. We have also developed the bound on the gap between *one-shot* and *recursive* algorithms, which can be found in Appendix C. We next highlight a few important advantages of the "recursive" WGAN configuration for the barycentric fast adaptation algorithm.

1) Two-Discriminator WGAN Implementation per Recursive Step to Enable Efficient Training: At each recursive step k, we aim to find the barycenter ν_k^* between pretrained model μ_k and the barycenter ν_{k-1}^* from last round, which is achieved by training a two-discriminator WGAN as follows:

$$\min_{\mathcal{G}_k} \max_{\psi_k, \tilde{\psi}_k} \lambda_{\psi_k} \left\{ \mathbb{E}_{x \sim \mu_k}[\psi_k(x)] - \mathbb{E}_{z \sim \vartheta}[\psi_k(\mathcal{G}_k(z))] \right\} + \lambda_{\tilde{\psi}_k} \left\{ \mathbb{E}_{x \sim \nu_{k-1}^*}\left[\tilde{\psi}_k(x)\right] - \mathbb{E}_{z \sim \vartheta}\left[\tilde{\psi}_k(\mathcal{G}_k(z))\right] \right\} \quad (6)$$

where ψ and $\tilde{\psi}$ denote the corresponding discriminators for the pretrained model G_k and the barycenter model $\mathcal{G}_{k-1}^*$ from the previous recursive step, respectively (see Algorithm 1).

2) Model Initialization in Each Recursive Step: For the initialization of the generator $\mathcal{G}_k$, we use the trained generator $\mathcal{G}_{k-1}^*$ in the last step. $\mathcal{G}_{k-1}^*$ corresponds to the barycenter ν_{k-1}^*, and using it as the initialization the displacement interpolation would move along the geodesic curve from ν_{k-1}^* to μ_k [12]. Training GANs with such initializations would accelerate the convergence compared with training from scratch [5]. Finally, ν_K^* is adopted as initialization to enable fast adaptation at the target node. As the barycenter ν_K^* solved via off-line training, a new barycenter ν^* between local data (represented by $\hat{\mu}_0$) and ν_K^*, can be obtained by solving the problem

$$\min_{\mathcal{G}_0} \max_{\psi_0, \tilde{\psi}_0} \lambda_{\psi_0} \left\{ \mathbb{E}_{x \sim \hat{\mu}_0}[\psi_0(x)] - \mathbb{E}_{z \sim \vartheta}[\psi_0(\mathcal{G}_0(z))] \right\} + \lambda_{\tilde{\psi}_0} \left\{ \mathbb{E}_{x \sim \nu_K^*}\left[\tilde{\psi}_0(x)\right] - \mathbb{E}_{z \sim \vartheta}\left[\tilde{\psi}_0(\mathcal{G}_0(z))\right] \right\} \quad (7)$$

i.e., by training a two-discriminator WGAN, and fine-tuning the generator $\mathcal{G}_0$ from $\mathcal{G}_K^*$ would be notably *faster and more accurate* than learning the generative model from local data only (see Algorithm 2).

3) Fast Adaptation for Training Ternary WGAN at Node 0: As outlined in Algorithm 2, fast adaptation is used to find the barycenter between ν_K^* and the local dataset at Node 0. To further enhance edge learning, we adopt the weight ternarization method to compress the WGAN model during training. The weight ternarization method not only replaces computationally expensive multiplication operations with efficient addition/subtraction operations but also enables the sparsity in model parameters [50]. Specifically, the ternarization process is formulated as

$$w_l' = S_l \cdot \text{Tern}\left(w_l, \Delta_l^{\pm}\right) = S_l \cdot \begin{cases} +1, & w_l > \Delta_l^+ \\ 0, & \Delta_l^- \le w_l \le \Delta_l^+ \\ -1, & w_l < \Delta_l^- \end{cases} \quad (8)$$

where $\{w_l\}$ are the full precision weights for the lth layer, $\{w_l'\}$ are the weights after ternarization, $\{S_l\}$ is the layerwise weight scaling coefficient, and $\Delta_l^{\pm}$ are the layerwise thresholds. Since the fixed weight thresholds may lead to accuracy degradation, S_l is approximated as a differentiable closed-form function of $\Delta_l^{\pm}$ so that both weights and thresholds can be optimized

Algorithm 2 Fast Adaptation Algorithm to Solve for Learning the Generative Model at Node 0

Inputs: Final generator $\mathcal{G}_K^*$, final discriminators $\tilde{\psi}_K^*$, ψ_K^*, noise prior $\vartheta(z)$, the batch size m, learning rate α

Set $\mathcal{G}_0^* \leftarrow \mathcal{G}_K^*$, $\tilde{\psi}_0^* \leftarrow$ rand() or $\tilde{\psi}_0^* \leftarrow \psi \in \{\tilde{\psi}_K^*, \psi_K^*\}$;

while generator $\mathcal{G}_0$ has not converged **do**

  Sample batches of prior samples $\{z^{(i)}\}_{i=1}^m$ and $\{z_{\tilde{\psi}_0}^{(i)}\}_{i=1}^m$ independently from prior $\vartheta(z)$;

  Get real data batch $\{x_{\psi_0}^{(i)}\}_{i=1}^m \sim \hat{\mu}_0$ and generate synthetic data batch $\{x_{\tilde{\psi}_0}^{(i)}\}_{i=1}^m \sim \nu_K^*$ by passing $\{z_{\tilde{\psi}_0}^{(i)}\}_{i=1}^m$ through $\mathcal{G}_K^*$;

  Compute gradients $g_{\tilde{\psi}_0}$ and g_{ψ_0}: $\{g_\omega \leftarrow \lambda_\omega \nabla_\omega \frac{1}{m} \sum_{i=1}^m [\omega(x_\omega^{(i)}) - \omega(\mathcal{G}_0(z^{(i)}))]\}_{\omega=\tilde{\psi}_0,\psi_0}$;

  Update both discriminators ψ_0 and $\tilde{\psi}_0$: $\{\omega \leftarrow \omega + \alpha \cdot \text{Adam}(\omega, g_\omega)\}_{\omega=\psi_0,\tilde{\psi}_0}$;

  Compute gradient $g_{\mathcal{G}_0} \leftarrow -\nabla_{\mathcal{G}_0} \frac{1}{m} \sum_{i=1}^m [\lambda_{\psi_0} \psi_0(\mathcal{G}_0(z^{(i)})) + \lambda_{\tilde{\psi}_0} \tilde{\psi}_0(\mathcal{G}_0(z^{(i)}))]$;

  Update generator $\mathcal{G}_0$: $\mathcal{G}_0 \leftarrow \mathcal{G}_0 - \alpha \cdot \text{Adam}(\mathcal{G}_0, g_{\mathcal{G}_0})$;

Assign $\mathcal{G}_0^* \leftarrow \mathcal{G}_0$

return Generator $\mathcal{G}_0^*$ for barycenter ν_0^*.

simultaneously through backpropagation [51]. Let the generator and the discriminators of WGAN at Node 0 be denoted by $\mathcal{G}_0$, $\tilde{\psi}_0$, and ψ_0, which can be parameterized by the ternarized weights $\{\mathbf{w}'_{l_\mathcal{G}}\}_{l_\mathcal{G}=1}^{L_\mathcal{G}}$, $\{\mathbf{w}'_{l_{\tilde{\psi}}}\}_{l_{\tilde{\psi}}=1}^{L_{\tilde{\psi}}}$, and $\{\mathbf{w}'_{l_\psi}\}_{l_\psi=1}^{L_\psi}$, respectively. The barycenter ν^* at Node 0, captured by $\mathcal{G}_0^*$, can be obtained by training the ternary WGAN via iterative updates of both weights and thresholds, which takes three steps in each iteration: 1) calculating the scaling coefficients and the ternary weights for $\mathcal{G}_0$, $\tilde{\psi}_0$, and ψ_0; 2) calculating the loss function using the ternary weights via forward propagation; and 3) updating the full precision weights and the thresholds via backpropagation (see Algorithm 3 in supplementary materials under Appendix E.A).

4) Implementation Challenges in W_2^2-Based GAN: The practical success of W1GAN can be largely attributed to the elegant structural relation between Kantorovich potentials, i.e., $\varphi = -\psi$. Unfortunately, when the quadratic cost W_2^2 is used, Kantorovich potentials translate to $\varphi = \psi^*$, where * denotes the convex conjugate. Note that W_2 is a metric, but W_2^2 is not. When implementing the W_2^2-based GAN, both Kantorovich potentials are estimated by two distinct DNNs that must satisfy the convex conjugate constraint, which is practically challenging. Very recent studies [12], [13], [29] attempt to enforce the convex conjugate constraint between these DNNs through approximations or regularization under certain assumptions, but it remains not well understood. In this article, we have carried out experimental studies to compare the performance of W_1- and W_2^2-based recursive WGAN configurations, and our findings corroborate that W1GAN performs better (see Appendixes D.F and D.G in the Supplementary Material).

V. Experiments

1) Datasets, Models, and Evaluation: We extensively examine the performance of learning a generative model, using the barycentric fast adaptation algorithm, on a variety of widely adapted datasets in the GAN literature, including CIFAR10, CIFAR100, LSUN, and MNIST [52], [53], [54]. In experiments, we used various DCGAN-based architectures [55] depending on the dataset as different datasets vary in image size, feature diversity, and sample size, e.g., image samples in MNIST have less diversity compared to the rest of the datasets, while LSUN contains the largest number of samples with larger image sizes. Furthermore, we used the weight ternarization method [51] to jointly optimize weights and quantizers of the generative model at the target edge node, reducing the memory burden of generative models in memory-limited edge devices. Details on the characteristics of datasets and network architectures used in experiments are relegated to the Supplementary Material. The implementation details for all the results are provided in the Supplementary Material, and the implementation can be found in [56].

The lack of baseline experiment settings in EI inspired the fusion of experiment settings from GAN, transfer learning, and continual learning literature. To this end, multistage experiments are designed. First, individual WGAN models are trained on distinct subsets of a dataset (representing individual edge nodes) and evaluated using widely adopted scores from GAN literature [5]. In accordance with GAN literature, publicly available image datasets are selected, and each dataset is split into subsets of classes, which represents different tasks in a continual learning context [39]. The performance evaluation of different GAN architectures on image datasets is a well-studied field and provides a fair foundation for comparing the performance of the barycentric fast adaptation against other baselines. Therefore, the same, well-established DNN architectures are used across all techniques. In the second stage, the fast adaptation technique and its counterparts are operated on the trained WGAN models from the first stage to train a fusion WGAN for a different task as in transfer learning [5]. Finally, the second stage is repeated to constitute the forgetting of previous tasks in the fusion WGAN model and compare the performance of candidate techniques.

The Frechet–Inception distance (FID) score [57] is widely adopted for evaluating the performance of GAN models [5], [58] since it provides a quantitative assessment of the similarity of a dataset to another reference dataset. Therefore, we use the FID score to evaluate the performance evolution of the two-stage adaptive coalescence algorithm and all baseline algorithms during training. We here emphasize that a smaller FID score of a GAN indicates that it has a better performance. Note that, to avoid one-sided scores and make a fair comparison, other evaluation metrics, in addition to the FID score, are also leveraged to quantify the performance of all algorithms. A more comprehensive discussion of FID and other metrics is relegated to the Supplementary Material. In all experiments, we use the entire dataset as the reference dataset.

To demonstrate the improvements by using the proposed framework based on *barycentric fast adaptation*, we conduct extensive experiments and compare performance with three distinct baselines.

1) *Transferring GANs [5]:* A pretrained GAN model is used as initialization at Node 0 for training a new WGAN model by using local data samples.
2) *Ensemble Method:* The model initialization, obtained by using pretrained GANs at other edge nodes, is further trained using both local data from Node 0 and synthetic data samples.
3) *Edge-Only:* Only the local dataset at Node 0 is used in WGAN training. Due to the lack of sample diversity at the target edge node, the WGAN model trained using local data only is not expected to attain good performance. In stark contrast, the WGAN model is trained using the proposed two-stage adaptive coalescence algorithm, inherits the diversity from the pretrained models at other edge nodes, and results in better performance compared to its counterparts. Needless to say, if the entire dataset were available at Node 0, the best performance would be achieved.

2) Experiment Setup: We consider the following two scenarios.

1) *Overlapping Case:* The classes of the data samples at other edge nodes and at Node 0 overlap.
2) *Nonoverlapping Case:* The classes of the data samples at other edge nodes and at Node 0 are mutually exclusive.

In overlapping experiments, a dataset is equally split into two subdatasets, and subdatasets are used to pretrain two WGAN

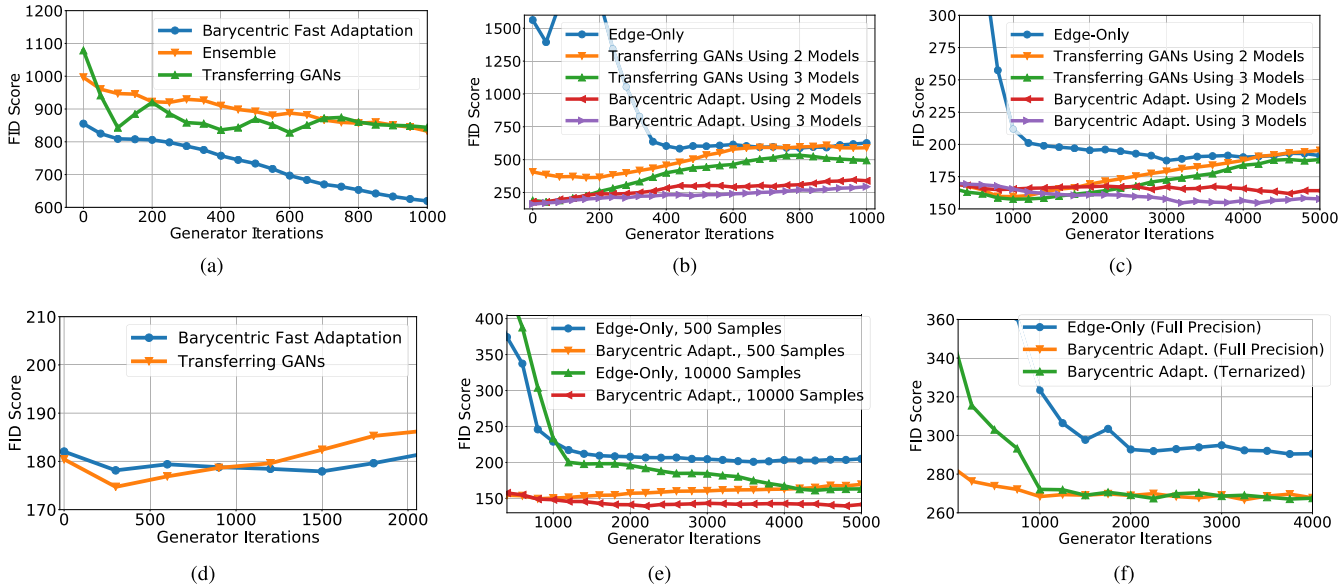


Fig. 5. Comparison of convergence of *barycentric fast adaptation* with various baselines. (a) MNIST: nonoverlapping case. (b) MNIST: overlapping case. (c) CIFAR10: overlapping case. (d) CIFAR100: overlapping case. (e) CIFAR10: overlapping case. (f) LSUN: overlapping case.

models independently. Subsequently, Algorithms 1 and 2 are used consecutively to find the barycenter and the final WGAN model at Node 0, respectively. The few data samples to be used in the fast adaptation stage are randomly selected from all classes in the dataset. For the *transferring GANs* method, the first pretrained model is further trained on the second subdataset using transfer learning to compute a fused model. In the final stage, few data samples from all classes in the dataset are used to train a WGAN model at Node 0 by leveraging the fused model via transfer learning again. In the *ensemble* method, the pretrained models are used to generate a synthetic dataset. The synthetic dataset is combined with the few data samples from all the classes, and the combined dataset is leveraged to train a final WGAN model at Node 0. Finally, the *edge-only* method only uses the few data samples from all the classes to train a WGAN model at Node 0.

In nonoverlapping experiments, randomly drawn samples from the first 40% of the classes in the dataset are allocated into the first node, and randomly drawn samples from the second 40% of the classes are allocated into the second node. The same steps as in the overlapping case are followed by using these two subdatasets until the final stage. In the final stage, a few data samples are randomly selected from the remaining 20% of the classes and are placed in Node 0.

3) Continual Learning Against Catastrophic Forgetting: We investigate the convergence and generated image quality of various training scenarios on CIFAR100 and MNIST datasets. As illustrated in Fig. 5, *barycentric fast adaptation* outperforms all baselines. *Transferring GANs* suffers from catastrophic forgetting because the continual learning is performed over local data samples at Node 0 only. On the contrary, the *barycentric fast adaptation* and the ensemble method leverage generative replay, which mitigates the negative effects of catastrophic forgetting. Furthermore, observe that the ensemble method suffers because of the limited data samples at Node 0, which are significantly outnumbered by synthetic data samples from pretrained GANs, and this imbalance degrades the applicability of the ensemble method for continual learning. On the other hand, the *barycentric fast adaptation* can obtain the barycenter between the local data samples at Node 0 and the barycenter model trained offline and, hence, can effectively leverage the abundance of data samples from edge nodes and accuracy of data samples at Node 0 for better continual learning.

4) Impact of Number of Pretrained Generative Models: To quantify the impact of cumulative model knowledge from pretrained generative models on the learning performance at the target node, we consider the scenario where ten classes in CIFAR10/MNIST are split into three subsets, e.g., the first pretrain model has classes $\{0, 1, 2\}$, the second pretrained model has classes $\{2, 3, 4\}$, and the third pretrained model has the remaining classes. One barycenter model is trained offline by using the first two pretrained models, and the second barycenter model is trained using all three pretrained models, respectively, based on which we evaluate the performance of *barycentric fast adaptation* with 1000 data samples at the target node. Fig. 5(a) and (c) showcases that the more model knowledge is accumulated in the barycenter computed offline, the higher image quality is achieved at Node 0. As expected, more model knowledge can help new edge nodes in training higher quality generative models. In both figures, the *barycentric fast adaptation* outperforms *transferring GANs*.

5) Impact of the Number of Data Samples at Node 0: Fig. 5(e) further illustrates the convergence across different numbers of data samples at the target node on the CIFAR10 dataset. As expected, the FID score gap between *barycentric fast adaptation* and *edge-only* method decreases as the number of data samples at the target node increases, simply because the empirical distribution becomes more "accurate." In particular, the significant gap of FID scores between *edge-only* and the *barycentric fast adaptation* approaches

in the initial stages indicates that the barycenter found via off-line training and adopted as the model initialization for fast adaptation is indeed close to the underlying model at the target node, hence enabling faster and more accurate edge learning than *edge-only*.

6) Impact of Wasserstein Ball Radii: The Wasserstein ball radius η_k for the pretrained model k represents the relevance (hence utility) of the knowledge transfer, which is intimately related to the capability to generalize beyond the pretrained generative models, and the smaller it is, the more informative the corresponding Wasserstein ball is. Hence, larger weights $\lambda_k = 1/\eta_k$ would be assigned to the nodes with higher relevance. We note that the weights are determined by the constraints and, thus, are fixed. Since we introduce the recursive WGAN configuration, the order of coalescence (each corresponding to a geodesic curve) may impact the final barycentric WGAN model and, hence, the performance of *barycentric fast adaptation*. To this end, we compute the coalescence of models of nodes with higher relevance at latter recursions to ensure that the final barycentric model is closer to the models of nodes with higher relevance.

7) Ternary WGAN-Based Barycentric Fast Adaptation: With the model initialization in the form of a full-precision barycenter model computed in off-line training, we next train a ternary WGAN with two discriminators for the target node to compress the generative model further. In particular, we use the same split of classes as the experiment illustrated in Fig. 5(e) and compare the image quality obtained by ternary WGAN-based fast adaptation against both full precision counterpart and *edge-only*. As seen in Fig. 5(f), the ternary WGAN-based *barycentric fast adaptation* results in negligible performance degradation compared to its full precision counterpart and is still much better compared to the *edge-only* approach.

8) Performance Evaluation Using Inception Score: In addition to the FID score, we also use IS, another widely used metric, to signify the robustness of the performance evaluation. Each of the three different numerical experiments is repeated five times, and the performance evaluation using FID and ISs is illustrated in Fig. 6. It is clear that both FID and IS evaluations corroborate the superior performance of the *barycentric fast adaptation* algorithm, as well as the small deviation from the mean performance. The worst and best case performances of the *barycentric fast adaptation* and its counterparts are illustrated in Table I. *best-mean, worst-mean, best*, and *worst* denote the best mean performance, the worst mean performance, the best performance in all five runs, and the worst performance in all five runs for the corresponding metrics, respectively. Table I further showcases the superior performance of *barycentric fast adaptation* in comparison to its counterparts, particularly when the number of available samples at Node 0 is limited.

An important observation herein is that both FID and IS quantify the quality and class diversity of the generated images. Specifically, the FID score leverages another large dataset (reference dataset) (the whole dataset in our experiments) to *relatively* compute the quality and class diversity of the generated images, whereas IS does not utilize a reference dataset, i.e., IS is *absolute*. A significant implication of this difference is that the FID score of a generated dataset can be different with respect to different reference datasets, while IS will be constant. Therefore, IS cannot quantify the effects of generator overfitting and the FID score does. In Fig. 6(c) and (f), only 20 data samples are used to train the WGAN generator models, and hence, the final WGAN models are prone to extreme overfitting. The WGAN models might generate the same images even for different values of (z), i.e., the image diversity within every class might be very low. In accordance with the generator overfitting phenomenon, we observe that the FID scores for all three methods are stationary after 2000 iterations in Fig. 6(f), whereas the IS curves continue to improve for *edge-only* and *transferring GANs* in Fig. 6(c). This indicates that generator overfitting occurs in the WGAN model trained with the *edge-only* and *transferring GANs* methods, whereas both the IS and FID scores for the *barycentric fast adaptation* method are stationary, indicating no generator overfitting.

9) Continual Learning Performance Across Dissimilar Data Samples: This experiment explores the performance of the *barycentric fast adaptation* when off-line barycenter is trained from pretrained WGAN models that are learned from two distinct datasets, e.g., CIFAR10 samples are placed in Node 1 and CIFAR100 samples are placed in Node 2. Specifically, samples from all ten classes in CIFAR10 are placed in Node 1, and the samples from the random 20 classes of CIFAR100 are placed in Node 2. In the *overlapping* scenario, Node 0 contained few samples from the same classes placed in Node 1 and Node 2. In the *nonoverlapping* scenario, Node 0 had few samples from 20 classes of CIFAR100, which are different from the classes in Node 2. We run experiments for *overlapping* and *nonoverlapping* scenarios for various number of samples at Node 0.

Fig. 7 demonstrates that *barycentric fast adaptation* outperforms the baselines for any sample size. The performance gap between *barycentric fast adaptation* and its counterparts increases significantly as fewer data samples are available at Node 0 because *barycentric fast adaptation* is better at retaining the previous knowledge from the edge and mitigating catastrophic forgetting. It is clear that the performance gap between *barycentric fast adaptation* and its counterparts is higher in this experiment because less model similarity of pretrained WGAN models amplifies the effects of catastrophic forgetting.

VI. Complexity Analysis

Computational cost is important for the success of EI techniques since edge nodes and edge servers are limited in computational power. The computational cost of DNN training, however, heavily depends on the DNN architecture and input size. To exemplify, a conventional 2-D-convolutional layer has a computational complexity of $\mathcal{O}({C_i}^2 {K_i}^2 H_i W_i)$ per input, where C_i, K_i, H_i, and W_i denote the number of channels, the kernel size, and the height and the width of the output for the ith layer, respectively. The generator and

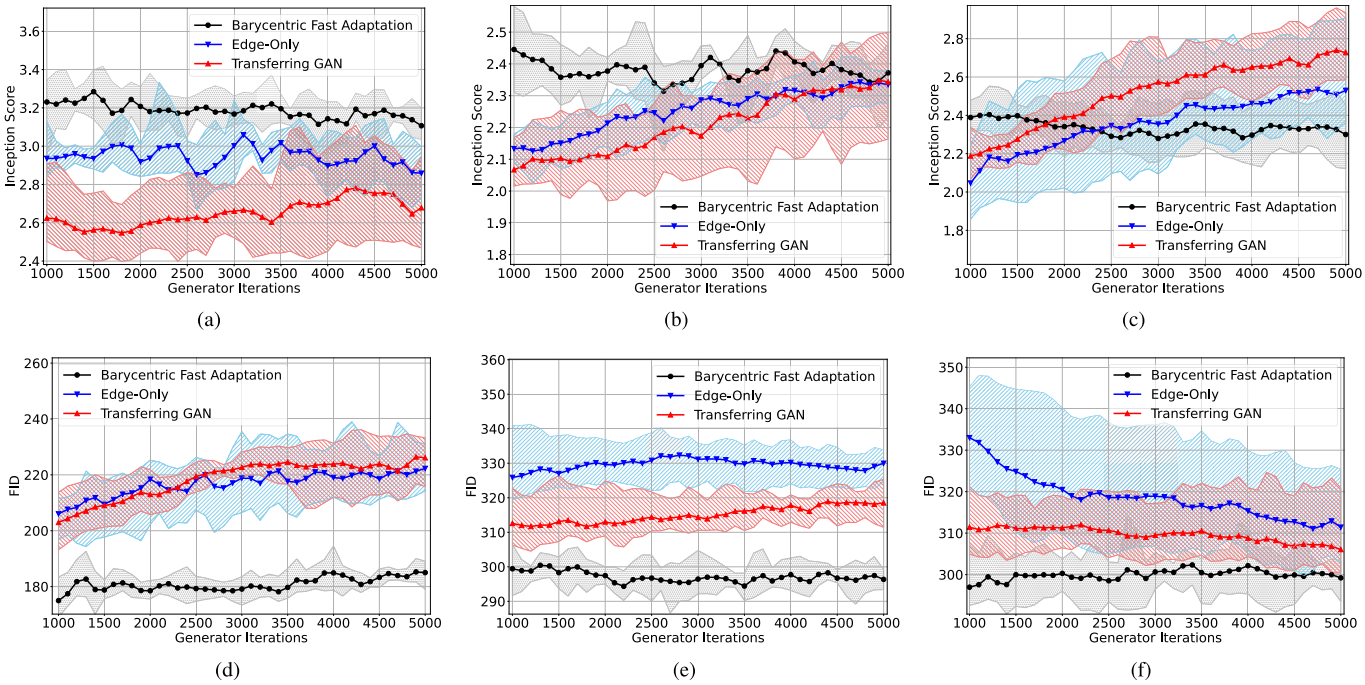


Fig. 6. Performance evaluation comparisons of various WGAN model training techniques using FID and inception scores (ISs) tested on CIFAR10. FID and ISs conclude similar performance results. Note that higher IS and lower FID scores indicate better performance. (a) IS for the overlapping case with 100 data samples at Node 0. (b) IS for the overlapping case with 50 data samples at Node 0. (c) IS score for the overlapping case with 20 data samples at Node 0. (d) FID score for the overlapping case with 100 data samples at Node 0. (e) FID score for the overlapping case with 50 data samples at Node 0. (f) FID score for the overlapping case with 20 data samples at Node 0.

TABLE I

PERFORMANCE COMPARISONS OF DIFFERENT WGAN MODEL TRAINING ALGORITHMS

	Metric\ Experiment	Inception Score				Frechet-Inception Distance			
		Best-Mean	Worst-Mean	Best	Worst	Best-Mean	Worst-Mean	Best	Worst
Fig. 6(a), 6(d)	Fast Adaptation	3.28 ± 0.13	3.11 ± 0.08	3.42	2.95	175 ± 7	185 ± 4	169	195
	Edge-Only	3.06 ± 0.11	2.85 ± 0.18	3.33	2.67	206 ± 7	222 ± 9	194	239
	Transferring GANs	2.78 ± 0.32	2.55 ± 0.15	3.11	2.40	203 ± 11	226 ± 8	193	236
Fig. 6(b), 6(e)	Fast Adaptation	2.45 ± 0.13	2.31 ± 0.13	2.58	2.12	294 ± 5	300 ± 4	287	306
	Edge-Only	2.34 ± 0.08	2.13 ± 0.09	2.42	2.03	326 ± 11	332 ± 7	319	341
	Transferring GANs	2.35 ± 0.16	2.07 ± 0.07	2.50	1.97	312 ± 8	319 ± 4	305	326
Fig. 6(c), 6(f)	Fast Adaptation	2.40 ± 0.12	2.28 ± 0.16	2.55	2.12	297 ± 7	302 ± 4	291	314
	Edge-Only	2.53 ± 0.27	2.05 ± 0.19	2.91	1.86	311 ± 13	333 ± 12	300	348
	Transferring GANs	2.74 ± 0.19	2.19 ± 0.13	2.96	2.07	306 ± 10	312 ± 7	300	325

the discriminator for CIFAR10/100 contain four convolutional layers with varying channel and activation sizes. The resulting computational complexity of forward and backward passes is then becomes $\mathcal{O}((C^\dagger)^2 K^2 H^\dagger W^\dagger)$ per sample, where $C^\dagger = \max_{i \in \mathcal{I}} C_i$, $H^\dagger = \max_{i \in \mathcal{I}} H_i$, $W^\dagger = \max_{i \in \mathcal{I}} W_i$, and $\mathcal{I}$ denotes the set of layers in both discriminator and generator. *Barycentric fast adaptation* and its counterparts use the same DNNs, but *barycentric fast adaptation* leverages two discriminators against the single discriminator in its counterparts.

Run-times and attained accuracy levels during WGAN training at Node 0 across different techniques are compared in Table II. Smaller run-time indicates lower computational cost, and "DNA" indicates that the corresponding algorithm did not achieve the image quality level. Results indicate a significant improvement over *edge-only* in computational complexity. While *barycentric fast adaptation* performs similarly to *transferring GANs* in lower quality image generation, *barycentric fast adaptation* can generate higher quality images with less training. We here notice that each training iteration for *edge-only* and *transferring GANs* takes less time compared to *barycentric fast adaptation* because the latter trains two discriminators. In particular, the WGAN models trained by the *transferring GANs* method suffer from catastrophic forgetting as training progresses. On the contrary, *barycentric fast adaptation* can train WGAN models capable of generating high-quality images

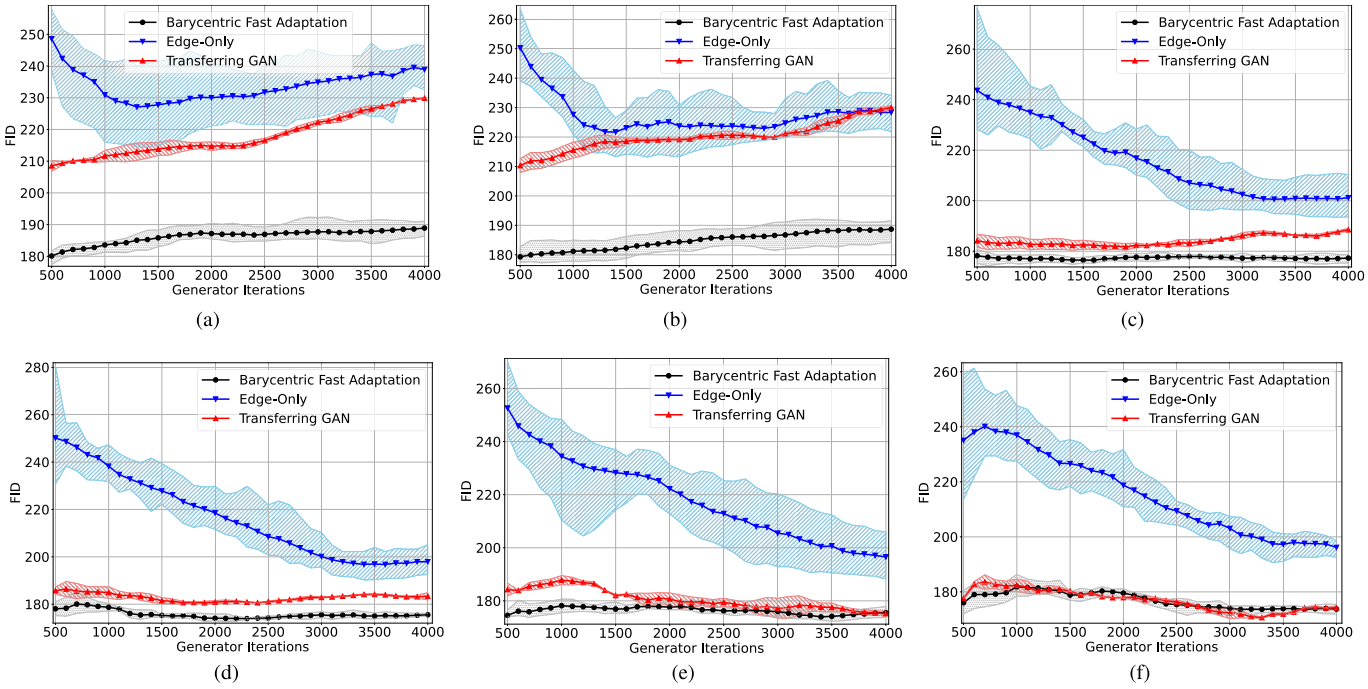


Fig. 7. Performance evaluation comparisons of various WGAN model training techniques using FID score tested on the mixture of datasets CIFAR10 and CIFAR100. Note that a lower FID score indicates better performance. (a) Nonoverlapping case: 100 samples at Node 0. (b) Overlapping case: 150 samples at Node 0. (c) Nonoverlapping case: 400 samples at Node 0. (d) Overlapping case: 600 samples at Node 0. (e) Nonoverlapping case: 2000 samples at Node 0. (f) Overlapping case: 3000 samples at Node 0.

TABLE II
COMPUTATIONAL COMPLEXITY COMPARISON OF BARYCENTRIC FAST ADAPTION AGAINST ITS COUNTERPARTS

	Database	MNIST			CIFAR10				CIFAR10 & CIFAR100			
	Method\Target FID	600	400	200	200	175	165	160	200	190	180	175
Iterations	Fast Adaptation	1	1	20	1	1	3000	4500	1	1	1	2200
	Transferring GANs	1	100	DNA	1	1	1000	DNA	1	1	1600	DNA
	Edge-Only	360	DNA	DNA	1200	DNA	DNA	DNA	2500	3400	DNA	DNA
Seconds	Fast Adaptation	1.0	1.0	20.4	1.5	1.5	4440.0	6660.0	1.5	1.5	1.5	3300.0
	Transferring GANs	0.5	53.5	DNA	0.8	0.8	755.0	DNA	0.8	0.8	1208.0	DNA
	Edge-Only	192.6	DNA	DNA	906.0	DNA	DNA	DNA	1887.5	2567.0	DNA	DNA

even at later stages of training since it overcomes catastrophic forgetting.

VII. CONCLUSION

In this work, we propose a systematic framework for continual learning of generative models via the adaptive coalescence of pretrained models from other edge nodes. Particularly, we cast the continual learning problem as a constrained optimization problem that can be reduced to a Wasserstein-1 barycenter problem. Appealing to the optimal transport theory, we characterize the geometric properties of geodesic curves therein and use displacement interpolation as the foundation to devise recursive algorithms for finding adaptive barycenters. Next, we take a two-stage approach to efficiently solve the barycenter problem, where the barycenter of the pretrained models is first computed offline in the edge servers via a "recursive" WGAN configuration based on displacement interpolation. Then, the resulting barycenter is treated as the metamodel initialization, and fast adaptation is used to find the generative model using the local samples at the target edge node. A weight ternarization method, based on joint optimization of weights and threshold for quantization, is developed to compress the edge generative model further. Extensive experimental studies corroborate the efficacy of the proposed framework.

APPENDIX A
PROOF OF PROPOSITION 1 FOR WASSERSTEIN-1 GAN

Proof: Let $\{\mu_k\}_{1:K}$ be any set of probability measures on a refined forming set and ν_k^* denote a continuous probability measure with no atoms, which minimizes the problem $\min_{\nu_k} W_1(\mu_k, \nu_k) + W_1(\nu_{k-1}^*, \nu_k)$ [17]. As Proposition 4 in supplementary materials under Appendix D-E states, there exist multiple refined forming sets, and the proceeding proof holds true for any refined forming set induced by the original set of probability distributions. The proceeding proof utilizes the geodesic property and the existence of a barycenter in Wasserstein-1 space, for which the details can be found in [17],

[47], and [59], respectively. Let the barycenter at iteration $k = 1$ be $\nu_1^* = \mu_1$, and suppose that $\alpha \notin \mathcal{B}_\mathcal{R}$ is a distribution satisfying

$$\min_{\nu_2} W_1(\mu_2, \nu_2) + W_1\left(\nu_1^*, \nu_2\right) = W_1(\mu_2, \alpha) + W_1(\mu_1, \alpha) \quad (9)$$

at recursion $k = 2$. Note that, if $\alpha \notin \mathcal{B}_\mathcal{R}$, α cannot reside on the geodesic curve $g_t(\mu_1, \mu_2)_{0\le t\le 1}$ since $g_t(\mu_1, \mu_2)_{0\le t\le 1} \in \mathcal{B}_\mathcal{R}$. By considering any distribution β_2, which resides on geodesic curve $g_t(\mu_1, \mu_2)$, we can also show that

$$\begin{aligned} W_1(\mu_1, \beta_2) + W_1(\mu_2, \beta_2) &= W_1(\mu_1, \beta_2) + W_1(\beta_2, \mu_2) \\ &= W_1(\mu_1, \mu_2) < W_1(\mu_1, \alpha) + W_1(\alpha, \mu_2) \\ &= \min_{\nu_2} W_1(\mu_2, \nu_2) + W_1\left(\nu_1^*, \nu_2\right) \end{aligned} \quad (10)$$

indicating that β attains a lower cost than the minimizer ν_2^*, which is a contradiction, indicating that ν_2^* must reside in $\mathcal{B}_\mathcal{R}$. Similarly, ν_3^* must also reside in $\mathcal{B}_\mathcal{R}$

$$\begin{aligned} W_1(\mu_3, \beta_3) + W_1\left(\nu_2^*, \beta_3\right) &= W_1(\mu_3, \beta_3) + W_1\left(\beta_3, \nu_2^*\right) \\ &= W_1\left(\mu_3, \nu_2^*\right) < W_1(\mu_3, \alpha) + W_1\left(\alpha, \nu_2^*\right). \end{aligned} \quad (11)$$

By induction, $\beta_k \in \mathcal{B}_\mathcal{R}$ attains a lower cost compared with $\alpha \notin \mathcal{B}_\mathcal{R}$ at the kth iteration

$$\begin{aligned} W_1(\mu_k, \beta_k) + W_1\left(\nu_{k-1}^*, \beta_k\right) &= W_1(\mu_k, \beta_k) + W_1\left(\beta_k, \nu_{k-1}^*\right) \\ = W_1\left(\mu_k, \nu_{k-1}^*\right) &< W_1(\mu_k, \alpha) + W_1\left(\alpha, \nu_{k-1}^*\right). \end{aligned} \quad (12)$$

Hence, $\nu_k^* = \beta_k \in \mathcal{B}_\mathcal{R}, \forall k$. Consequently, all barycenters at each iteration must reside in the baryregion $\mathcal{B}_\mathcal{R}$.

Similarly, we can show that, for stage II, the following holds:

$$\begin{aligned} W_1(\mu_0, \beta_K) + W_1\left(\nu_K^*, \beta_K\right) &= W_1(\mu_0, \beta_K) + W_1\left(\beta_K, \nu_K^*\right) \\ = W_1\left(\mu_0, \nu_K^*\right) &< W_1(\mu_0, \alpha) + W_1\left(\alpha, \nu_K^*\right). \end{aligned} \quad (13)$$

Therefore, ν^* resides in $\mathcal{B}_\mathcal{R}$, which completes the proof. □

Appendix B
Remark 1 for Quadratic Wasserstein-2 Cost Function

For ease of exposition, we restate the seminal result by [19].

Proposition 2: [19] The barycenter (in W_2^2 sense) of $\{(\mu_k, \lambda_k)\}_k$, i.e., $\arg\min_\nu \sum_k^K \lambda_k W_2^2(\mu_k, \nu)$, constitutes a unique solution ν^* if $\{\mu_k\}_{1:K}$ vanishes on small sets. The unique solution is characterized as $\nu^* = \nabla\phi_k \sharp \mu_k$, where ϕ_k is a convex potential defined in terms of the Kantorovich potentials [[19], eq. (3.5)].

Corollary 1: If $K = 2$, the set of barycenters ν_t^* is characterized as

$$\begin{aligned} \nu_t^* &= \arg\min_\nu t W_2^2(\mu_1, \nu) + (1-t) W_2^2(\mu_2, \nu) \\ &= \left(t\mathrm{id} + (1-t)\nabla\phi^*\right)\sharp\mu_1 \end{aligned} \quad (14)$$

where id is the identity map and $\nabla\phi^*$ is the conjugate Brenier's map between μ_0 and μ_1.

Corollary 1 implies that ν_t^* is the geodesic curve between μ_1 and μ_2. Hence, the solution to the W_2^2 barycenter problem with $K = 2$ and the fixed weight pair $(t, 1-t)$ always resides on the geodesic curve between μ_1 and μ_2 [19], [46].

Proof: (Displacement Interpolation for Adaptive Barycenters with W_2^2 Cost Function): Let $\{\mu_k\}_{1:K}$ be a set of probability measures on a refined forming set, which vanish on small sets, and ν_k^* denotes a continuous probability measure with no atoms, which minimizes the problem $\min_{\nu_k} W_1(\mu_k, \nu_k) + W_1(\nu_{k-1}^*, \nu_k)$ [17]. The following proof builds upon Corollary 1. We consider the following barycenter sequence generated by the recursive W2GAN configuration:

$$\begin{aligned} \mathcal{S} = \Big\{ &\nu_1^* = \mu_1, \nu_2^* = \arg\min_\nu t W_2^2(\mu_2, \nu) + (1-t) W_2^2\left(\nu_1^*, \nu\right), \\ &\ldots, \nu_K^* = \arg\min_\nu t W_2^2(\mu_K, \nu) + (1-t) W_2^2\left(\nu_{K-1}^*, \nu\right) \Big\}. \end{aligned} \quad (15)$$

For ease of exposition, we assign μ_1 as the unique barycenter in the first recursion, i.e., $\nu_1^* = \mu_1 \in \mathcal{B}_\mathcal{R}$. In iteration 2, the new barycenter can be stated in terms of the previous barycenter as

$$\begin{aligned} \nu_2^*(t, 1-t) &= \arg\min_\nu t W_2^2(\mu_2, \nu) + (1-t) W_2^2\left(\nu_1^*, \nu\right) \\ &= \left((1-t)\mathrm{id} + t\nabla\phi^*\right)\sharp\nu_1^*. \end{aligned} \quad (16)$$

We note that $\nu_2^*(t, 1-t) = ((1-t)\mathrm{id} + t\nabla\phi^*)\sharp\nu_1^*$ defines a geodesic between μ_2 and ν_1^*, and hence, $\nu_2^*(t, 1-t) \in \mathcal{B}_\mathcal{R}$ by definition. Furthermore, the barycenter, $\nu_2^*(t, 1-t)$, is unique by Proposition 2 and Corollary 1. By induction, the kth barycenter is expressed as

$$\begin{aligned} \nu_k^*(t, 1-t) &= \arg\min_\nu t W_2^2(\mu_k, \nu) + (1-t) W_2^2\left(\nu_{k-1}^*, \nu\right) \\ &= \left((1-t)\mathrm{id} + t\nabla\phi^*\right)\sharp\nu_{k-1}^*. \end{aligned} \quad (17)$$

As before, $\nu_k^*(t, 1-t)((1-t)\mathrm{id} + t\nabla\phi^*)\sharp\nu_{k-1}^* \in \mathcal{B}_\mathcal{R}$ and $\nu_k^*(t, 1-t)$ is the unique barycenter for a fixed $(t, 1-t)$ pair, and hence, we conclude that all the barycenters in the sequence $\mathcal{S}$ reside inside the baryregion $\mathcal{B}_\mathcal{R}$. Similarly, for the fast adaptation stage, the final barycenter ν^* can be shown to reside on the geodesic $((1-t)\mathrm{id} + t\nabla\phi^*)\sharp\nu_K^*$ between ν_K^* and $\hat{\mu}_0$

$$\begin{aligned} \nu^*(t, 1-t) &= \arg\min_\nu t W_2^2\left(\hat{\mu}_0, \nu\right) + (1-t) W_2^2\left(\nu_K^*, \nu\right) \\ &= \left((1-t)\mathrm{id} + t\nabla\phi^*\right)\sharp\nu_K^* \end{aligned} \quad (18)$$

and, hence, $\nu^* \in \mathcal{B}_\mathcal{R}$, which completes the proof. □

Appendix C
Bounds on the Gap Between One-Shot Barycentric and Recursive Barycentric Algorithms

The W_1 barycenter problem is analytically challenging because the geodesic curve between two distributions is not unique, which may lead to multiple barycenters at every recursion (see [21, Example 7.4]). Proposition 1 shows that every barycenter resides inside the baryregion $\mathcal{B}_\mathcal{R}$ for any $\{\lambda_k\}$. It follows that $\mathcal{B}_\mathcal{R}$ also provides a bound on the gap between the *one-shot* and *recursive* Wasserstein barycenter problems under the W_1 cost function. For the W_2 cost function, it is shown in [49] that the approximation gap can be driven to be 0. Proposition 3 demonstrates how to achieve this by using the proposed recursive algorithm.

Proposition 3: Let $\{\lambda_1, \lambda_2, \ldots, \lambda_K\}$ and $\{\lambda_{\mu_1}, \lambda_{\mu_2}, \ldots, \lambda_{\mu_K}\}$ denote the weights of the distributions $\{\mu_1, \mu_2, \ldots, \mu_K\}$ in *one-shot* and *recursive* W_2 barycenter problems, respectively, and let $\{\lambda_{\nu_1^*}, \lambda_{\nu_2^*}, \ldots, \lambda_{\nu_K^*}\}$ denote the weights of the barycenters in the *recursive* W_2 barycenter problem. The solutions of *one-shot* and *recursive* W_2 barycenter problems are the same if $\lambda_{\nu_k^*} = \sum_{\ell=1}^{k} \lambda_\ell / \sum_{\ell=1}^{k+1} \lambda_\ell$ and $\lambda_{\mu_k} = \lambda_k / \sum_{\ell=1}^{k+1} \lambda_\ell$, $\forall k \in \mathcal{K}$.

Proof: Without loss of generality, we assume that $\sum_{i=1}^{K} \lambda_i = 1$. Then, from Corollary 1, we have that

$$\begin{aligned}
K=1 \rightarrow \quad & \nu_1^* = \mu_1; \quad \lambda_{\nu_1^*} = \lambda_{\mu_1} = \lambda_1 \\
K=2 \rightarrow \quad & \nu_1^* = \mu_1, \nu_2^* = \left(\lambda_{\nu_1^*} \text{i.d.} + \lambda_{\mu_k} T_{\mu_2}^{\nu_1^*}\right) \sharp \nu_1^* \\
& \lambda_{\nu_1^*} = \lambda_1 / \lambda_1 + \lambda_2, \lambda_{\mu_2} = \lambda_2 / \lambda_1 + \lambda_2 \\
K=3 \rightarrow \quad & \nu_1^* = \mu_1, \nu_2^* = T_{\nu_2^*}^{\nu_1^*} \sharp \nu_1^* = \left(\lambda_{\nu_1^*} \text{i.d.} + \lambda_{\mu_k} T_{\mu_2}^{\nu_1^*}\right) \sharp \nu_1^* \\
& \nu_3^* = \left(\lambda_{\nu_2^*} T_{\nu_2^*}^{\nu_1^*} + \lambda_{\mu_3} T_{\mu_3}^{\nu_1^*}\right) \sharp \nu_1^* \\
& \quad = \left(\lambda_{\nu_2^*} \lambda_{\nu_1^*} \text{i.d.} + \lambda_{\nu_2^*} \lambda_{\mu_2} T_{\mu_2}^{\nu_1^*} + \lambda_{\mu_3} T_{\mu_3}^{\nu_1^*}\right) \sharp \nu_1^* \\
& \lambda_{\nu_1^*} = \lambda_1 / \lambda_1 + \lambda_2, \lambda_{\mu_2} = \lambda_2 / \lambda_1 + \lambda_2 \\
& \lambda_{\nu_2^*} = \lambda_1 + \lambda_2 / \lambda_1 + \lambda_2 + \lambda_3, \lambda_{\mu_3} = \lambda_3 / \lambda_1 + \lambda_2 + \lambda_3. \qquad (19)
\end{aligned}$$

By induction, we have

$$\begin{aligned}
K=k \rightarrow \quad & \nu_i^* = \left(\lambda_{\nu_{i-1}^*} T_{\nu_{i-1}^*}^{\nu_1^*} + \lambda_{\mu_i} T_{\mu_i}^{\nu_1^*}\right) \sharp \nu_1^* \quad \forall i = 1, \ldots, k \\
& \lambda_{\nu_i^*} = \sum_{j=1}^{i} \lambda_j / \sum_{j=1}^{i+1} \lambda_j \\
& \lambda_{\mu_{i+1}} = \lambda_{i+1} / \sum_{j=1}^{i+1} \lambda_j \forall i = 1, \ldots, k-1. \qquad (20)
\end{aligned}$$

Then, for $K = k+1$, we can show

$$\begin{aligned}
K = k+1 \rightarrow \nu_{i+1}^* &= \left(\lambda_{\nu_i^*} T_{\nu_i^*}^{\nu_1^*} + \lambda_{\mu_{i+1}} T_{\mu_{i+1}}^{\nu_1^*}\right) \sharp \nu_1^* \\
&= \left(\lambda_{\nu_i^*} \left(\lambda_{\nu_{i-1}^*} T_{\nu_{i-1}^*}^{\nu_1^*} + \lambda_{\mu_i} T_{\mu_i}^{\nu_1^*}\right) + \lambda_{\mu_{i+1}} T_{\mu_{i+1}}^{\nu_1^*}\right) \sharp \nu_1^* \\
&= \left(\lambda_{\nu_i^*} \lambda_{\nu_{i-1}^*} T_{\nu_{i-1}^*}^{\nu_1^*} + \lambda_{\nu_i^*} \lambda_{\mu_i} T_{\mu_i}^{\nu_1^*} + \lambda_{\mu_{i+1}} T_{\mu_{i+1}}^{\nu_1^*}\right) \sharp \nu_1^* \quad \forall i = 1, \ldots, k \\
\lambda_{\nu_i^*} = \sum_{j=1}^{i} \lambda_j / \sum_{j=1}^{i+1} \lambda_j, \lambda_{\mu_{i+1}} &= \lambda_{i+1} / \sum_{j=1}^{i+1} \lambda_j \quad \forall i = 1, \ldots, k \qquad (21)
\end{aligned}$$

which completes the proof. □

REFERENCES

[1] Y. Jiang et al., "Model pruning enables efficient federated learning on edge devices," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Apr. 25, 2022, doi: 10.1109/TNNLS.2022.3166101.

[2] A. Z. Tan, H. Yu, L. Cui, and Q. Yang, "Towards personalized federated learning," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Mar. 28, 2022, doi: 10.1109/TNNLS.2022.3160699.

[3] 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," *Proc. IEEE*, vol. 107, no. 8, pp. 1738–1762, Aug. 2019.

[4] R. Yonetani, T. Takahashi, A. Hashimoto, and Y. Ushiku, "Decentralized learning of generative adversarial networks from non-iid data," 2019, *arXiv:1905.09684*.

[5] Y. Wang, C. Wu, L. Herranz, J. van de Weijer, A. Gonzalez-Garcia, and B. Raducanu, "Transferring GANs: Generating images from limited data," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 218–234.

[6] G. M. Park, S. M. Yoo, and J. H. Kim, "Convolutional neural network with developmental memory for continual learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 6, pp. 2691–2705, Jun. 2020.

[7] H. Li, P. Barnaghi, S. Enshaeifar, and F. Ganz, "Continual learning using Bayesian neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 9, pp. 4243–4252, Sep. 2020.

[8] L. Breiman, "Bagging predictors," *Mach. Learn.*, vol. 24, no. 2, pp. 123–140, 1996.

[9] J. Smith and M. Gashler, "An investigation of how neural networks learn from the experiences of peers through periodic weight averaging," in *Proc. 16th IEEE Int. Conf. Mach. Learn. Appl. (ICMLA)*, Dec. 2017, pp. 731–736.

[10] R. Kemker, M. McClure, A. Abitino, T. Hayes, and C. Kanan, "Measuring catastrophic forgetting in neural networks," in *Proc. AAAI Conf. Artif. Intell.*, 2018, vol. 32, no. 1, pp. 1–9.

[11] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein GAN," 2017, *arXiv:1701.07875*.

[12] J. Leygonie, J. She, A. Almahairi, S. Rajeswar, and A. Courville, "Adversarial computation of optimal transport maps," 2019, *arXiv:1906.09691*.

[13] H. Liu, X. Gu, and D. Samaras, "Wasserstein GAN with quadratic transport cost," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 4832–4841.

[14] S. Arora, R. Ge, Y. Liang, T. Ma, and Y. Zhang, "Generalization and equilibrium in generative adversarial nets (GANs)," in *Proc. 34th Int. Conf. Mach. Learn.*, vol. 70, 2017, pp. 224–232.

[15] J. Lou and Y.-M. Cheung, "An uplink communication-efficient approach to featurewise distributed sparse optimization with differential privacy," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 10, pp. 4529–4543, Oct. 2021.

[16] Y. Brenier, "Polar factorization and monotone rearrangement of vector-valued functions," *Commun. Pure Appl. Math.*, vol. 44, no. 4, pp. 375–417, Jun. 1991.

[17] L. Ambrosio, N. Gigli and G. Savaré, *Gradient Flows in Metric Spaces and in the Space of Probability Measures* (Lectures in Mathematics ETH Zürich). Basel, Switzerland: Birkhänser, 2008.

[18] C. Villani, *Optimal Transport: Old and New*, vol. 338. Berlin, Germany: Springer-Verlag.

[19] M. Agueh and G. Carlier, "Barycenters in the Wasserstein space," *SIAM J. Math. Anal.*, vol. 43, no. 2, pp. 904–924, 2011.

[20] M. Cuturi and A. Doucet, "Fast computation of Wasserstein barycenters," in *Proc. Int. Conf. Mach. Learn.*, 2014, pp. 685–693.

[21] S. Srivastava, V. Cevher, Q. Dinh, and D. Dunson, "WASP: Scalable Bayes via barycenters of subset posteriors," in *Proc. 18th Int. Conf. Artif. Intell. Statist.*, 2015, pp. 912–920.

[22] E. Anderes, S. Borgwardt, and J. Miller, "Discrete Wasserstein barycenters: Optimal transport for discrete data," *Math. Methods Oper. Res.*, vol. 84, no. 2, pp. 389–409, Oct. 2016.

[23] M. Staib, S. Claici, J. M. Solomon, and S. Jegelka, "Parallel streaming Wasserstein barycenters," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 2647–2658.

[24] T. Nguyen et al., "TIDOT: A teacher imitation learning approach for domain adaptation with optimal transport," in *Proc. IJCAI*, 2021, pp. 2862–2868.

[25] L. Chen, Z. Gan, Y. Cheng, L. Li, L. Carin and J. Liu, "Graph optimal transport for cross-domain alignment," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 1542–1553.

[26] L. Chen et al., "Sequence generation with optimal-transport-enhanced reinforcement learning," in *Proc. AAAI Conf. Artif. Intell.*, vol. 34, Apr. 2020, pp. 7512–7520.

[27] Q. Li, Z. Wang, G. Li, J. Pang, and G. Xu, "Hilbert sinkhorn divergence for optimal transport," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 3834–3843.

[28] I. Goodfellow et al., "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2672–2680.

[29] A. Korotin, V. Egiazarian, A. Asadulaev, A. Safin, and E. Burnaev, "Wasserstein-2 generative networks," in *Proc. Int. Conf. Learn. Represent.*, 2021, pp. 1–30.

[30] I. Durugkar, I. Gemp, and S. Mahadevan, "Generative multi-adversarial networks," 2016, *arXiv:1611.01673*.

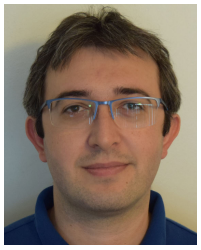
[31] D. Simon and A. Aberdam, "Barycenters of natural images constrained Wasserstein barycenters for image morphing," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2020, pp. 7910–7919.

[32] M. Javad Shafiee, F. Li, B. Chwyl, and A. Wong, "SquishedNets: Squishing SqueezeNet further for edge device scenarios via deep evolutionary synthesis," 2017, *arXiv:1711.07459*.

[33] S. A. Osia et al., "A hybrid deep learning architecture for privacy-preserving mobile analytics," *IEEE Internet Things J.*, vol. 7, no. 5, pp. 4505–4518, May 2020.

[34] Q. Gao, Z. Luo, D. Klabjan, and F. Zhang, "Efficient architecture search for continual learning," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Mar. 2, 2022, doi: 10.1109/TNNLS.2022.3151511.

[35] S. Venkataramani et al., "ScaleDeep: A scalable compute architecture for learning and evaluating deep networks," in *Proc. 44th Annu. Int. Symp. Comput. Archit.*, 2017, pp. 13–26.
[36] S. Lin, G. Yang, and J. Zhang, "Real-time edge intelligence in the making: A collaborative learning framework via federated meta-learning," 2020, *arXiv:2001.03229*.
[37] Z. Zhang, S. Lin, M. Dedeoglu, K. Ding, and J. Zhang, "Data-driven distributionally robust optimization for edge intelligence," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Jul. 2020, pp. 2619–2628.
[38] S. Thrun, "A lifelong learning perspective for mobile robot control," in *Intelligent Robots and Systems* Amsterdam, The Netherlands: Elsevier, 1995, pp. 201–214.
[39] X. Su, S. Guo, T. Tan, and F. Chen, "Generative memory for lifelong learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 6, pp. 1884–1898, Jun. 2020.
[40] C. Wu et al., "Memory replay GANs: Learning to generate new categories without forgetting," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 5962–5972.
[41] H. Shin, J. K. Lee, J. Kim, and J. Kim, "Continual learning with deep generative replay," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 2990–2999.
[42] S.-W. Lee, J.-H. Kim, J. Jun, J.-W. Ha, and B.-T. Zhang, "Overcoming catastrophic forgetting by incremental moment matching," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 4652–4662.
[43] M. De Lange et al., "A continual learning survey: Defying forgetting in classification tasks," 2019, *arXiv:1909.08383*.
[44] R. Volpi, H. Namkoong, O. Sener, J. C. Duchi, V. Murino, and S. Savarese, "Generalizing to unseen domains via adversarial data augmentation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 5334–5344.
[45] A. Sinha, H. Namkoong, R. Volpi, and J. Duchi, "Certifying some distributional robustness with principled adversarial training," 2017, *arXiv:1710.10571*.
[46] R. J. McCann, "A convexity principle for interacting gases," *Adv. Math.*, vol. 128, no. 1, pp. 153–179, Jun. 1997.
[47] C. Villani, *Topics in Optimal Transportation*, no. 58. Providence, RI, USA: American Mathematical Society, 2003.
[48] C. Hardy, E. Le Merrer, and B. Sericola, "MD-GAN: Multi-discriminator generative adversarial networks for distributed datasets," in *Proc. IEEE Int. Parallel Distrib. Process. Symp. (IPDPS)*, May 2019, pp. 866–877.
[49] E. Boissard et al., "Distribution's template estimate with Wasserstein metrics," *Bernoulli*, vol. 21, no. 2, pp. 740–759, 2015.
[50] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding," 2015, *arXiv:1510.00149*.
[51] Z. He and D. Fan, "Simultaneously optimizing weight and quantizer of ternary neural network using truncated Gaussian approximation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 11438–11446.
[52] L. Deng, "The MNIST database of handwritten digit images for machine learning research," *IEEE Signal Process. Mag.*, vol. 29, no. 6, pp. 141–142, Nov. 2012.
[53] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Tech. Rep. TR-2009, 2009.
[54] F. Yu, A. Seff, Y. Zhang, S. Song, T. Funkhouser, and J. Xiao, "LSUN: Construction of a large-scale image dataset using deep learning with humans in the loop," 2015, *arXiv:1506.03365*.
[55] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," 2015, *arXiv:1511.06434*.
[56] M. Dedeoglu. *Continual-Learning-of-Generative-Models-With-Limited-Data*. Figshare. Accessed: Feb. 18, 2023. [Online]. Available: https://figshare.com/articles/software/Continual-Learning-of-Generative-Models-with-Limited-Data/22121375
[57] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "GANs trained by a two time-scale update rule converge to a local nash equilibrium," in *Proc. Adv. Neural Inf. Process. Syst.*, I. Guyon et al., Eds. Red Hook, NY, USA: Curran Associates, 2017, pp. 6626–6637.
[58] P. Grnarova et al., "A domain agnostic measure for monitoring and evaluating GANs," in *Proc. Adv. Neural Inf. Process. Syst.* Red Hook, NY, USA: Curran Associates, 2019, pp. 12092–12102.
[59] T. Le Gouic and J.-M. Loubes, "Existence and consistency of Wasserstein barycenters," *Probab. Theory Rel. Fields*, vol. 168, nos. 3–4, pp. 901–917, Aug. 2017.



Mehmet Dedeoglu (Student Member, IEEE) received the B.Sc. and M.Sc. degrees in electrical and electronics engineering from Bilkent University, Ankara, Turkey, in 2012 and 2014, respectively, and the Ph.D. degree in electrical, computer, and energy engineering from Arizona State University, Tempe, AZ, USA, in 2021.

His current research interests lie in the nexus of generative adversarial networks, deep learning, optimization, signal processing, and algorithms.

Dr. Dedeoglu received the Best Student Paper Award at the 22nd IEEE Signal Processing and Communications Applications (SIU) Conference in 2014.

Sen Lin (Member, IEEE) received the Ph.D. degree in electrical engineering from Arizona State University, Tempe, AZ, USA, in 2021.

He is currently a Post-Doctoral Scholar with the NSF AI-EDGE Institute, The Ohio State University, Columbus, OH, USA. His research interests broadly fall in the intersection of machine learning and wireless networking. Currently, his research focuses on developing algorithms and theories in continual learning, metalearning, reinforcement learning, adversarial machine learning, and bilevel optimization, with applications in multiple domains, e.g., edge computing, security, and network control.

Dr. Lin's papers received several awards, including the International Symposium on Modelling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOPT)'18 Best Student Paper Award and the Spotlight Presentation in International Conference on Learning Representations (ICLR) 2022.

Zhaofeng Zhang (Student Member, IEEE) received the B.Eng. degree in electrical engineering from the Huazhong University of Science and Technology, Wuhan, China, in 2015, and the M.S. degree in electrical engineering from Arizona State University, Tempe, AZ, USA, in 2017, where he is currently pursuing the Ph.D. degree with the School of Electrical, Computer, and Energy Engineering.

His research interests include edge computing, statistical machine learning, deep learning, and optimization.

Junshan Zhang (Fellow, IEEE) received the Ph.D. degree from the School of Electrical and Computer Engineering (ECE), Purdue University, West Lafayette, IN, USA, in 2000.

He was in the Faculty of the School of Electrical, Computer and Energy Engineering (ECEE), Arizona State University, Tempe, AZ, USA, from 2000 to 2021. He co-founded Smartiply Inc., Basking Ridge, NJ, USA, a fog computing startup company delivering boosted network connectivity and embedded artificial intelligence. He is currently a Professor with the Department of ECE, University of California at Davis, Davis, CA, USA. His research interests fall in the general field of information networks and data science, including edge intelligence, reinforcement learning, continual learning, network optimization and control, and game theory, with applications in connected and automated vehicles, 5G and beyond, wireless networks, IoT data privacy/security, and smart grids.

Prof. Zhang received the ONR Young Investigator Award in 2005 and the NSF CAREER Award in 2003. His papers won several awards, including the Best Student Paper Award at International Symposium on Modelling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOPT) 2018, the Kenneth C. Sevcik Outstanding Student Paper Award of ACM SIGMETRICS/IFIP Performance 2016, the Best Paper Runner-Up Award of IEEE International Conference on Computer Communications (INFOCOM) 2009 and IEEE INFOCOM 2014, and the Best Paper Award at IEEE International Conference on Communications (ICC) 2008 and ICC 2017. He has served as the Editor-in-Chief of IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS from 2019 to 2022.