OPTIMAL TRANSPORT-BASED DOMAIN ALIGNMENT AS A PREPROCESSING STEP FOR FEDERATED LEARNING

Luiz Manella Pereira and M. Hadi Amini, Senior Member, IEEE

Knight Foundation School of Computing and Information Sciences, Florida International University
Security, Optimization, and Learning for InterDependent networks laboratory (solid lab)
www.solidlab.network

ABSTRACT

Federated learning (FL) is a subfield of machine learning that avoids sharing local data with a central server, which can enhance privacy and scalability. The inability to consolidate data leads to a unique problem called dataset imbalance, where agents in a network do not have equal representation of the labels one is trying to learn to predict. In FL, fusing locally-trained models with unbalanced datasets may deteriorate the performance of global model aggregation, and reduce the quality of updated local models and the accuracy of the distributed agents' decisions. In this work, we introduce an Optimal Transport-based preprocessing algorithm that aligns the datasets by minimizing the distributional discrepancy of data along the edge devices. We accomplish this by leveraging Wasserstein barycenters when computing channel-wise averages. These barycenters are collected in a trusted central server where they collectively generate a target RGB space. By projecting our dataset towards this target space, we minimize the distributional discrepancy on a global level, which facilitates the learning process due to a minimization of variance across the samples. We demonstrate the capabilities of the proposed approach over the CIFAR-10 dataset, where we show its capability of reaching higher degrees of generalization in fewer communication rounds.

Index Terms— Optimal Transport, Federated Learning, Domain Alignment, Image Preprocessing

1. INTRODUCTION

Federated Learning (FL), a subdomain of Machine Learning (ML), addresses the challenges of decentralized or distributed model training. It offers a compelling framework for scenarios in which data cannot be centrally aggregated due to privacy constraints, thereby promoting compliance with data protection regulations and enhancing scalability [1]. Beyond its foundational role in privacy-preserving learning, FL also facilitates model personalization—adapting learning outcomes to individual users across the network—an increasingly relevant objective given the heterogeneity of user behavior and datasets. A comprehensive overview of the challenges and practical implementations of personalized federated learning is presented in [2].

Despite its broad applicability, particularly in contexts with stringent data privacy constraints, FL introduces a set of constraints that must be carefully addressed to ensure robust and efficient model training. These constraints include limited communication bandwidth, restricted computation at edge devices, privacy preservation requirements, and data heterogeneity and imbalance. Dataset imbalance in FL emerges when edge devices possess non-uniform class distributions, disparate dataset sizes, or varying data quality [3, 4]. In this work, we propose a preprocessing framework that addresses this imbalance challenge in a model- and algorithm-agnostic manner. Our method aligns and transforms local datasets into a shared representation space that captures statistical information from all participating agents in the network.

2. RELATED WORKS

The issue of dataset imbalance, also known as the domain-distribution discrepancy problem or the multiple-source domain problem, has been a core challenge in federated learning. In this section, we will review various approaches to address this challenge. Our approach is, to our knowledge, the first to perform zero-shot distribution alignment in FL and only the second to incorporate optimal transport (OT). Unlike prior work, our method eliminates the need for iterative

training-time computation by applying OT as a preprocessing step, which reduces overhead and enhances scalability. Although it may be interesting to think of various preprocessing techniques to compare against, to our knowledge, none exist. In our research, we have only seen local alignments and processing techniques such as feature normalization (e.g., min-max scaling) or standardization (e.g., Z-score scaling). While these techniques are useful to help train models by scaling feature values to desirable ranges, they do not align samples across agents. The lack of literature further supports the importance of our idea.

The most relatable work we have found is that of Farnia et al. [5], where the authors introduce an OT-based alignment step during the process of training the model. While their method learns the space they project to and the projection map dynamically during training, our approach performs alignment before training, avoiding iterative optimization. The authors begin by extending the standard OT task between two distributions to a multi-marginal OT problem. They then use these results to create a min-max optimization problem, which leads to their algorithm called *FedOT*, which is a dynamic approach to learning the data-transformation map. In this work, we instead present a zero-shot alignment that leverages [6] to create the alignment map, and in turn, lower computation cost.

In [7], Wang et al. comprehensively surveyed the field of domain adaptation. They describe three primary techniques used to solve this problem: data manipulation, representation learning, and learning strategy. While domain adaptation and generalization have been broadly studied, only a few methods directly address distribution alignment in FL. These include transfer learning approaches [8], representation aggregation [9], and adversarial training [10]. However, these either require task-specific models or exhibit inferior performance compared to OT-based methods like L2A-OT [11]. First, [8] focuses on personalization through transfer learning. The pseudo domain generalization occurs in the transfer step, as the *FedHealth* algorithm aligns the lower levels of the convolutional neural network (CNN) aggregated in the cloud with those locally stored, and personalized, in each agent's device. On the other hand, in [9], Wu and Gong tackle the learning strategy approach of domain generalization (DG) and unsupervised domain adaptation (UDA). In their algorithm, called Collaborative OPtimization and Aggregation (COPA), each agent in the network contains a local dataset, a feature extractor, and a classifier. The local feature extractors are collected to create a global feature extractor that is domain-invariant. The local classifiers are ensembled to create the global classifier. These global models are used to update the local models until convergence. Lastly, [10] subscribes to the "learning strategy" approach to solving domain generalization. They accomplish this through their algorithm *FedADG*, which uses an adversarial component to measure and then align different source distributions to a globally known, shared distribution. While *FedADG* improves on some test cases, the authors mention their approach to generating invariant features yields results that are typically worse than an OT-based approach called L2A-OT [11].

Thus far, we have covered papers that are directly related to our work. We now focus on other research that indirectly tackles, or at least provides, a direction to work with imbalanced datasets. We first turn to clustering in federated learning. In [12], Sattler et al. propose their main algorithm, clustered federated learning (CFL). In an imbalanced dataset paradigm, the agents' data-generating distributions are different. To overcome this challenge, CFL clusters the incoming gradients of the agents through a recursive cosine similarity-based bipartitioning and then updates the model's parameters once for each cluster by averaging the gradients of the clients in each cluster respectively. Moreover, in [13], Hsu et al. create a custom dataset and analyze the degree of imbalance based on geo-location tagging and distributional statistics. To tackle the class distribution shift problem in FL, they began with, *FedAvg*, short for federated averaging [1], and updated it to include variance mitigation and a resampling strategy via adding weights to each agent. Their algorithms, respectively called *FedVC* and *FedIR*, computed the weights as a ratio of the target joint distribution of images and class labels and a sampled joint distribution of a specific client's image-label pair. The authors propose using Bayes theorem to reformulate the weight function as a ratio of the global distribution of class y and the agent's local distribution of class y. The weights are used to scale the updates of the global model sitting in the trusted server and lead to better accuracy.

Lastly, personalization is an interesting direction because it has been both used as a solution to the aforementioned problem and as an application to improve local, customized solutions for FL. Fallah et al. present *MAML* an algorithm that trains an "initial shared model" which is updated using a few gradient steps to yield a good, personalized model [14]. For more algorithms regarding personalization, we refer readers to [2]. n summary, while prior work has explored various strategies—dynamic OT, adversarial alignment, clustering, and personalization—none provide a model-agnostic, zero-shot distribution alignment mechanism. Our method fills this gap with a computationally efficient, one-time preprocessing step that enhances the stability of training.

3. PRELIMINARIES

3.1. Federated Learning

The goal of federated learning (FL) is to train a single (global) model to make accurate predictions across all agents in a network. FL problems contain constraints, including computational power at the edge, communication bandwidth, and data heterogeneity. Privacy is one of the key motivations for using FL which prevents the amalgamation of data in a central

server. To overcome this problem, FL research introduced novel techniques that are variations of parameter averaging or gradient averaging. A more comprehensive look at the challenges of FL and associated solutions can be found in [3]. Independently of the approach, at its core, FL looks for a solution to

$$\min_w f(w), \quad f(w) = \sum_{a=1}^{A} p_a f_a(w) \tag{1}$$

where A is the number of devices in our network, $p_a \geq 0$, and $\sum_a p_a = 1$. Each agent has a local objective, $f_a(w)$, which is averaged together as the global objective $f(w)$. For more details into equation 1, we refer readers to [3].

3.2. Optimal Transport

Optimal Transport (OT), founded by Gaspard Monge, is a mathematical framework that addresses the problem of finding the most cost-efficient way to transport mass from a source distribution to a target distribution [15]. OT has found widespread applications in various fields, including economics, computer vision, machine learning, and statistics. At its core, OT seeks to quantify the discrepancy between two probability distributions by defining a distance metric that considers both the magnitudes of the masses being transported and the distances over which they are moved. Unlike traditional distance metrics, such as Euclidean distance, which focus solely on point-to-point comparisons, OT provides a more geometrically nuanced measure that captures the structural similarities between distributions.

Kantorovich's formulation of OT introduces the concept of mass-splitting, which allows mass from the domain to be broken up and mapped to different locations in the codomain. Mathematically, it reads:

$$L_{\mathbf{C}}(\mathbf{a}, \mathbf{b}) = \min_{P \in U(a,b)} \langle \mathbf{C}, \mathbf{P} \rangle = \sum_{i,j} \mathbf{C}_{i,j} \mathbf{P}_{i,j}, \tag{2}$$

given the set of *admissible couplings*

$$U(\mathbf{a}, \mathbf{b}) = \left\{ P \in R_+^{n \times m} : \mathbf{P} \mathbb{1}_m = \mathbf{a}, \mathbf{P}^T \mathbb{1}_n = \mathbf{b} \right\}$$

Intuitively, one is looking for a permutation matrix $\mathbf{P}$ that determines how to distribute mass in a cost-minimizing fashion given the transportation cost $\mathbf{C}$.

OT presents a novel approach to comparing two probability distributions. With additional constraints, OT yields a metric, or a distance function, called the *Wasserstein metric*, or *Earth Mover's Distance*. Suppose that for some $p \geq 1$ and $\mathbf{C} = \mathbf{D}^p$ where $\mathbf{D} \in \mathbb{R}^{n \times n}$ is a distance on $[\![n]\!]$, the following hold: 1- $\mathbf{D}$ is symmetric, 2- diag($\mathbf{D}$) = 0, and 3- $\forall (i,j,k) \in [\![n]\!]^3, \mathbf{D}_{i,k} \leq \mathbf{D}_{i,j} + \mathbf{D}_{j,k}$. Then the p-Wasserstein distance is

$$W_p(\mathbf{a}, \mathbf{b}) = L_{\mathbf{D}^p}(\mathbf{a}, \mathbf{b})^{1/p}. \tag{3}$$

Equipped with a distance function, we can now define an averaging function, called a Wasserstein barycenter (WB):

$$\min_{\mathbf{a} \in \Sigma_n} \sum_{s=1}^{S} \lambda_s W_p^p(\mathbf{a}, \mathbf{b}_s), \tag{4}$$

where λ_s is a real-valued weight (usually a uniform distribution such that each input probability vector is given an equal amount of importance [16]) and Σ_n is a probability simplex with n bins. Intuitively, given a set of input probability vectors $\mathbf{b}_s$, we are looking for a probability vector $\mathbf{a}$ that minimizes the weighted sum of the p-Wasserstein distance between $\mathbf{a}$ and each $\mathbf{b}$.

There are different methods to solve the optimal transport problem and to compute Wasserstein barycenters. The current state-of-the-art methods rely on entropic regularization. In [17], Marco Cuturi introduces a solution to quickly solve the entropy regularized OT problem, $W_{reg}(\mathbf{a}, \mathbf{b})$, using Sinkhorn's algorithm, which now reads: For $\lambda > 0$,

$$d_M^\lambda(a, b) = \langle P^\lambda, C \rangle, \tag{5}$$

where $P^\lambda = \mathrm{argmin}_{P \in U(a,b)} \langle P, C \rangle - \frac{1}{\lambda} h(P)$. Equipped with a computational solver, Benamou et al. leveraged iterative Bregman projections to compute the entropic regularized Wasserstein barycenter, which can be seen in [18], leading to a faster and more general solution. The entropic regularized barycenter problem can be written as an extension of equation 4:

$$\min_{\mathbf{a} \in \Sigma_n} \sum_{s=1}^{S} \lambda_s W_{reg}(\mathbf{a}, \mathbf{b}_s) \tag{6}$$

where $\boldsymbol{\lambda} = \{\lambda_s\}_{s=1}^S \in \Sigma_S$.

4. OPTIMAL TRANSPORT-BASED PREPROCESSING

In this section, we introduce the preprocessing step that minimizes the distributional discrepancy in our network. We achieve this distribution-alignment goal by generating a target space to which we project all local data. Our proposed method has two main steps: the creation of the target space, and the projection step to perform alignment.

To create a relevant target space to align images to, it needs to contain information from all the agents in the network. We accomplish this in a two-step fashion. First, we compute representations of the local data through a channel-wise Wasserstein barycenter of the local images. The approach requires separating each local image its three color channels, grouping them by the respective color channels, and lastly computing the WB for each channel. In this work, we are using colored images, therefore, channel-wise implies red, green, and blue channels. The local computations produce an RGB-triplet called the local WBs. The second step is to generate the target space by aggregating all local WBs in a

central server, and repeating the same process of computing channel-wise barycenters; this yields the RGB-tripled called the global WB, or our target space. Lastly, we broadcast the global barycenters to the agents in the network and then align the local images to the target space by projecting them to the global WB. The projection process is composed of computing transportation plans to the target space that allows the color channels of the original images to be transferred and aligned, similarly to computing color transfer maps or domain adaptation in [6] respectively. The entire preprocessing steps are explicitly laid out in algorithm 1. For clarity, $\mathcal{WB}_{r,g,b}^a$ are the Wasserstein barycenters of agent a and the respective color channel, $\mathbf{Img}_i^{red,green,blue}$ is a specific color channel of the ith image, and $\mathcal{WB}^G$ is the target space (the global Wasserstein barycenter) composed of the global RGB-triplet.

Algorithm 1 OT-based preprocessing

Definitions: Let $\mathbf{B} = \{\mathbf{b}_s\}_{s=1}^S$ such that $\text{WB}(\mathbf{B})$ is the solution to equation (6).

For each agent $a = 1, 2, \ldots, N$
$\quad\mathbf{R} = \{\mathbf{Img}_i^{red}\}_{i=1}^M$
$\quad\mathbf{G} = \{\mathbf{Img}_i^{green}\}_{i=1}^M$
$\quad\mathbf{B} = \{\mathbf{Img}_i^{blue}\}_{i=1}^M$
$\quad\mathcal{WB}_r^a = \text{WB}(\mathbf{R})$
$\quad\mathcal{WB}_g^a = \text{WB}(\mathbf{G})$
$\quad\mathcal{WB}_b^a = \text{WB}(\mathbf{B})$
$\quad$Distribute $\mathcal{WB}_r^a, \mathcal{W}_g^a, \mathcal{WB}_b^a$ to a central server
$\mathbf{R}_G = \{\mathcal{WB}_r^a\}_{a=1}^N$
$\mathbf{G}_G = \{\mathcal{WB}_g^a\}_{a=1}^N$
$\mathbf{B}_G = \{\mathcal{WB}_b^a\}_{a=1}^N$
$\mathcal{WB}_r^G = \text{WB}(\mathbf{R}_G)$
$\mathcal{WB}_g^G = \text{WB}(\mathbf{G}_G)$
$\mathcal{WB}_b^G = \text{WB}(\mathbf{B}_G)$
$\mathcal{WB}^G = \{\mathcal{W}_r^G, \mathcal{WB}_g^G, \mathcal{WB}_b^G\}$
Distribute $\mathcal{WB}^G$ to **all** agents
For each agent $a = 1, 2, \ldots, N$
$\quad$**For each image** $i = 1, 2, \ldots, M$ **of agent** a
$\quad\quad$Project image $i \rightarrow \mathcal{WB}^G$

---

5. EXPERIMENTS AND RESULTS

Our framework is built independently of the learning algorithm, which allows for flexible integration into other pipelines that may have different goals. To demonstrate the advantages of using our preprocessing step, we require a learning algorithm. To this end, we chose to work with federated averaging, *FedAvg*, which trains local algorithms and performs parameter averaging to aggregate the local models into a global model. To show the improvements our method provides to the learning process, we designed a few experiments over the CIFAR-10 dataset. Each experiment begins

Table 1. Comparison between *FedAvg* (i) with our preprocessing (column 3); (ii) without it (column 4). The first column is number of clients in the simulated network (N) and the number of clients used to train the global model (P). The second column (Communication Rounds), is the number of synchronization steps.

Client (N/P)	Comm. Rounds	**Our Work**	*FedAvg*
5/5	35	**99.62**	71.22
10/10	70	**99.33**	70.48
20/20	200	**99.36**	69.89
50/50	250	**97.84**	69.73
100/100	500	**94.95**	65.01
10/5	100	**99.85**	71.34
20/10	150	**97.5**	65.66
50/10	500	**95.04**	66.45
100/10	1000	**93.34**	66.16

with the following steps. First, we design the network based on a desired number of agents. Next, we initialize their models identically to avoid degeneracy [1]. We use both a custom CNN and ResNet9 in our experiments. Next, we distribute the data by uniformly sampling them without replacement. Lastly, we align the data using our preprocessing algorithm (1) and begin training.

There are various ways of deciding when to perform the synchronization step, also called communication round, which is where parameter averaging happens. The trade-off between choosing the number of local epochs and communication rounds when training a global model is dependent on how many agents participate in the model aggregation step. As per convention, N is the total number of agents and P is the number of participating agents. In experiments where $N = P$, we used two local training epochs per synchronization step. When $P << N$, the number of samples in the local dataset becomes closer to the batch size and we therefore increase the number of epochs to five. The results of our simulations using the custom CNN and the ResNet9 model are shown, respectively, in Tables 1 and 2. Moreover, Table 3 shows a comparison of our algorithm with other approaches.

Our OT-preprocessing algorithm, paired with *FedAvg*, an algorithm known for its simplicity, yielded the best results. Our simulations, while not using the exact same hyperparameters, are undoubtedly comparable since we achieved higher accuracy scores with a simpler, and smaller, model. These facts open the door for merging our work with algorithms that are tailored for other applications.

6. IMPLEMENTATION DETAILS

In this short section, we will clarify details for reproducibility. The custom CNN has an input convolutional layer (conv

Table 2. This table contains the results of simulations using our approach with a ResNet9 model and our custom CNN (for comparison). For clarification, "Comm. Rounds" is short for "Communication Rounds."

	Comm. Rounds		Testing Accuracy	
Client (N/P)	ResNet9	CNN	ResNet9	CNN
5/5	554	35	98.61	99.62
10/10	200	70	95.45	99.33
20/20	200	200	92.13	99.36

layer) with a 5×5 kernel and 64 filters and its output is fed into a max pool layer with a 2×2 kernel. Next, we have one more conv layer with the same kernel size but with 128 filters. The output is fed into a series of 3 linear layer, whose respective output dimensions are 256, 128, and 10. When training either CNN or ResNet9, we use a batch size of 16 with an Adam optimizer and a learning rate of $1e-3$.

The core of our algorithm relies on efficiently computing an entropic-regularized transport and a Wasserstein barycenter efficiently . We leverage the Python Optimal Transport (POT) package [1] which contains various solvers; for the transportation problem, we use the Sinkhorn algorithm, while for the barycenter we use the Bregmen projection method. Both methods require a regularization term, for which we use $1e-2$ and $1e-1$ respectively. Furthermore, a trick used to reduce the computations required is to select subsets of pixels from the domain and codomain. In our experiments, we uniformly sample a subset of 250 pixels and use them as input when computing barycenters and performing projections.

7. COMPLEXITY ANALYSIS

An important tradeoff to consider with our algorithm is the additional time that must be paid to convert the original dataset into an aligned dataset. As our method does not affect the learning algorithm, we only analyze the complexity of computing the barycenters and projecting the local data to the global barycenter, which is done prior to training. Therefore, this is an additional cost paid in addition to training. First, [22] gives us a complexity analysis for the iterative Bregman projection method of computing regularized barycenters, as introduced by [18]. Given n samples of dimension d and regularization parameter ϵ, we have a complexity of $O(nd^2/\epsilon^2)$. Moreover, the projection of the local images onto the global barycenters requires solving the OT problem. [17] demonstrates an empirical complexity of $O(d^2)$ with respect to the input dimension d. The complexity of our preprocessing is a combination of these two, contingent on the number of agents. Assume we have N agents in our network, each with M images. We must compute N barycenters. Since

Table 3. This table compares the results of various approaches to federated learning. The first two rows are direct results from our experiments. The subsequent rows are results obtained from the respective papers. Readers should look at the cited work for implementation details which lead to the varying accuracy scores for the same algorithms (as can be seen below). For further clarification, rows 1 and 2 (referencing our Algorithm and *FedAvg*), are simulations with N=100 and P=10.

Reference	Method	Accuracy
Our work	Algorithm 1	**93.34**
[1]	*FedAvg*	66.16
[19]	MOON	69.1
[19]	FedAvg	66.3
[19]	FedProx	66.9
[19]	SCAFFOLD	66.6
[19]	SOLO	46.3
[20]	FedMA	87.53
[20]	FedProx	85.32
[20]	FedAvg	86.29
[21]	FedAvg (CCVR)	71.03
[21]	FedProx (CCVR)	70.99
[21]	FedAvgM (CCVR)	71.49
[21]	MOON (CCVR)	71.29
[5]	FedOT	72.2

these are computed in parallel, the time complexity is equivalent to computing one, with the addition of computing the global barycenter. Therefore, we have $O(Md^2/\epsilon^2)$ for local barycenters and $O(Nd^2/\epsilon^2)$ for the global barycenter. The projection time complexity is scaled in accordance with M, not N, as each agent projects in parallel. Therefore, the overall time complexity for our preprocessing algorithm is $O(Md^2/\epsilon^2) + O(Nd^2/\epsilon^2) + O(Md^2)$. If the local number of images vary per agent, the worst-case complexity is the same and requires only setting $M = \text{argmax}_i M_i$ for agents $i = 1, \ldots, N$.

8. CONCLUSION AND FUTURE WORK

In this work, we demonstrated the ability of our proposed preprocessing algorithm to improve the convergence speed and generalization of traditional FL. We accomplish this by projecting local data to a space that encodes all local data, in turn minimizing the distributional discrepancy between agents. After aligning the local datasets, we train a model using *FedAvg*. Compared with certain methods (e.g., FedAvg and MOON), our proposed solution shows higher accuracy scores. In future work, we will explore larger datasets (e.g., ImageNet) and other modalities. We also aim to develop new

OT-based methods for transforming temporal data.

9. REFERENCES

[1] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*. PMLR, 2017, pp. 1273–1282.

[2] Alysa Ziying Tan, Han Yu, Lizhen Cui, and Qiang Yang, "Towards personalized federated learning," *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[3] Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith, "Federated learning: Challenges, methods, and future directions," *IEEE signal processing magazine*, vol. 37, no. 3, pp. 50–60, 2020.

[4] Wentai Wu, Ligang He, Weiwei Lin, and Carsten Maple, "Fedprof: Selective federated learning based on distributional representation profiling," *IEEE Transactions on Parallel and Distributed Systems*, vol. 34, no. 6, pp. 1942–1953, 2023.

[5] Farzan Farnia, Amirhossein Reisizadeh, Ramtin Pedarsani, and Ali Jadbabaie, "An optimal transport approach to personalized federated learning," *IEEE Journal on Selected Areas in Information Theory*, vol. 3, no. 2, pp. 162–171, 2022.

[6] Sira Ferradans, Nicolas Papadakis, Gabriel Peyré, and Jean-François Aujol, "Regularized discrete optimal transport," *SIAM Journal on Imaging Sciences*, vol. 7, no. 3, pp. 1853–1882, 2014.

[7] Jindong Wang, Cuiling Lan, Chang Liu, Yidong Ouyang, Tao Qin, Wang Lu, Yiqiang Chen, Wenjun Zeng, and Philip Yu, "Generalizing to unseen domains: A survey on domain generalization," *IEEE Transactions on Knowledge and Data Engineering*, 2022.

[8] Yiqiang Chen, Xin Qin, Jindong Wang, Chaohui Yu, and Wen Gao, "Fedhealth: A federated transfer learning framework for wearable healthcare," *IEEE Intelligent Systems*, vol. 35, no. 4, pp. 83–93, 2020.

[9] Guile Wu and Shaogang Gong, "Collaborative optimization and aggregation for decentralized domain generalization and adaptation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 6484–6493.

[10] Liling Zhang, Xinyu Lei, Yichun Shi, Hongyu Huang, and Chao Chen, "Federated learning with domain generalization," *arXiv preprint arXiv:2111.10487*, 2021.

[11] Kaiyang Zhou, Yongxin Yang, Timothy Hospedales, and Tao Xiang, "Learning to generate novel domains for domain generalization," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVI 16*. Springer, 2020, pp. 561–578.

[12] Felix Sattler, Klaus-Robert Müller, and Wojciech Samek, "Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints," *IEEE transactions on neural networks and learning systems*, vol. 32, no. 8, pp. 3710–3722, 2020.

[13] Tzu-Ming Harry Hsu, Hang Qi, and Matthew Brown, "Federated visual classification with real-world data distribution," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part X 16*. Springer, 2020, pp. 76–92.

[14] Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar, "Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach," *Advances in Neural Information Processing Systems*, vol. 33, pp. 3557–3568, 2020.

[15] Gaspard Monge, "Mémoire sur la théorie des déblais et des remblais," *Mem. Math. Phys. Acad. Royale Sci.*, pp. 666–704, 1781.

[16] Marco Cuturi and Arnaud Doucet, "Fast computation of wasserstein barycenters," in *International conference on machine learning*. PMLR, 2014, pp. 685–693.

[17] Marco Cuturi, "Sinkhorn distances: Lightspeed computation of optimal transport," *Advances in neural information processing systems*, vol. 26, 2013.

[18] Jean-David Benamou, Guillaume Carlier, Marco Cuturi, Luca Nenna, and Gabriel Peyré, "Iterative bregman projections for regularized transportation problems," *SIAM Journal on Scientific Computing*, vol. 37, no. 2, pp. A1111–A1138, 2015.

[19] Qinbin Li, Bingsheng He, and Dawn Song, "Model-contrastive federated learning," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 10713–10722.

[20] Hongyi Wang, Mikhail Yurochkin, Yuekai Sun, Dimitris Papailiopoulos, and Yasaman Khazaeni, "Federated learning with matched averaging," *arXiv preprint arXiv:2002.06440*, 2020.

[21] Mi Luo, Fei Chen, Dapeng Hu, Yifan Zhang, Jian Liang, and Jiashi Feng, "No fear of heterogeneity: Classifier calibration for federated learning with non-iid data," *Advances in Neural Information Processing Systems*, vol. 34, pp. 5972–5984, 2021.

[22] Alexey Kroshnin, Nazarii Tupitsa, Darina Dvinskikh, Pavel Dvurechensky, Alexander Gasnikov, and Cesar Uribe, "On the complexity of approximating wasserstein barycenters," in *International conference on machine learning*. PMLR, 2019, pp. 3530–3540.