Clustered Federated Learning: Model-Agnostic Distributed Multitask Optimization Under Privacy Constraints

Felix Sattler, Klaus-Robert Müller, *Member, IEEE*, and Wojciech Samek, *Member, IEEE*

***Abstract*— Federated learning (FL) is currently the most widely adopted framework for collaborative training of (deep) machine learning models under privacy constraints. Albeit its popularity, it has been observed that FL yields suboptimal results if the local clients' data distributions diverge. To address this issue, we present clustered FL (CFL), a novel federated multitask learning (FMTL) framework, which exploits geometric properties of the FL loss surface to group the client population into clusters with jointly trainable data distributions. In contrast to existing FMTL approaches, CFL does not require any modifications to the FL communication protocol to be made, is applicable to general nonconvex objectives (in particular, deep neural networks), does not require the number of clusters to be known *a priori*, and comes with strong mathematical guarantees on the clustering quality. CFL is flexible enough to handle client populations that vary over time and can be implemented in a privacy-preserving way. As clustering is only performed after FL has converged to a stationary point, CFL can be viewed as a postprocessing method that will always achieve greater or equal performance than conventional FL by allowing clients to arrive at more specialized models. We verify our theoretical analysis in experiments with deep convolutional and recurrent neural networks on commonly used FL data sets.**

***Index Terms*— Clustering, distributed learning, federated learning, multi-task learning.**

I. Introduction

FEDERATED learning (FL) [1]–[5] is a distributed training framework, which allows multiple clients (typically, mobile or the IoT devices) to jointly train a single deep learning model on their combined data in a communication-efficient way, without requiring any of the participants to reveal their private training data to a centralized entity or to each other. FL realizes this goal via an iterative three-step protocol where, in every communication round t, the clients first synchronize with the server by downloading the latest master model θ_t. Every client then proceeds to improve the downloaded model by performing multiple iterations of stochastic gradient descent with minibatches sampled from its local data D_i, resulting in a weight-update vector

$$\Delta\theta_i^{t+1} = \mathrm{SGD}(\theta^t, D_i) - \theta^t, \quad i = 1, \ldots, M. \tag{1}$$

Finally, all clients upload their computed weight-updates to the server, where they are aggregated by weighted averaging according to

$$\theta^{t+1} = \theta^t + \sum_{i=1}^{M} \frac{|D_i|}{|D|} \Delta\theta_i^{t+1} \tag{2}$$

to create the next master model (see [1]). The procedure is summarized in Algorithm 2.

FL implicitly makes the assumption that it is possible for one single model to fit all client's data generating distributions φ_i at the same time. Given a model $f_\theta : \mathcal{X} \to \mathcal{Y}$ parameterized by $\theta \in \Theta$ and a loss function $l : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_{\geq 0}$, we can formally state this assumption as follows.

Assumption 1 (FL): There exists a parameter configuration $\theta^* \in \Theta$ that (locally) minimizes the risk on all clients' data generating distributions at the same time

$$R_i(\theta^*) \leq \min_{\substack{\theta \\ \|\theta-\theta^*\|<\varepsilon}} R_i(\theta), \qquad i = 1, \ldots, M \tag{3}$$

for some $\varepsilon > 0$. Hereby

$$R_i(\theta) = \int l(f_\theta(x), y) d\varphi_i(x, y) \tag{4}$$

is the risk function associated with distribution φ_i.

It is easy to see that this assumption is not always satisfied. Concretely, it is violated if either: 1) the model f_θ is not expressive enough to fit all distributions at the same time or 2) clients have disagreeing conditional distributions $\varphi_i(y|x) \neq \varphi_j(y|x)$. Simple counterexamples for both cases are presented in Fig. 1.

In the following, we will call a set of clients and their data generating distributions φ congruent (with respect to f and l) if they satisfy Assumption 1 and incongruent if they do not.

Manuscript received October 2, 2019; revised February 25, 2020 and June 12, 2020; accepted August 8, 2020. Date of publication August 24, 2020; date of current version August 4, 2021. This work was supported in part by the Berlin Institute for the Foundations of Learning and Data under Grant 01IS18025A and Grant 01IS18037, in part by the German Research Foundation (DFG) under Grant EXC 2046/1 and Project 390685689, and in part by the Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea Government under Grant 2017-0-00451. *(Corresponding authors: Wojciech Samek; Klaus-Robert Müller.)*

Felix Sattler and Wojciech Samek are with the Fraunhofer Heinrich Hertz Institute, 10587 Berlin, Germany (e-mail: wojciech.samek@hhi.fraunhofer.de).

Klaus-Robert Müller is with the Department of Electrical Engineering and Computer Science, Technische Universität Berlin, 10587 Berlin, Germany, also with the Max Planck Institute for Informatics, 66123 Saarbrücken, Germany, and also with the Department of Brain and Cognitive Engineering, Korea University, Seoul 136-713, South Korea (e-mail: klaus-robert.mueller@tu-berlin.de).

This article has supplementary downloadable material available at https://ieeexplore.ieee.org, provided by the authors.

Color versions of one or more of the figures in this article are available online at https://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2020.3015958

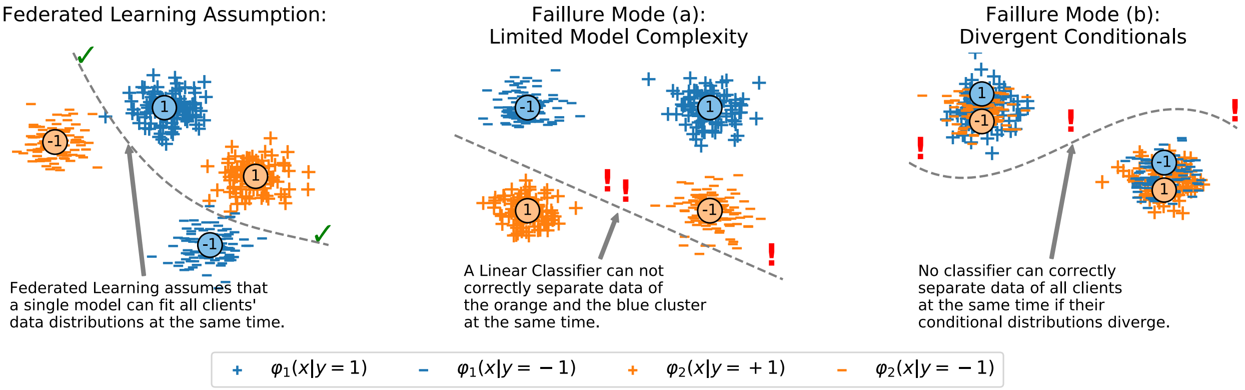


Fig. 1. Two toy cases in which the FL assumption is violated. Blue points belong to clients that follow φ_1, while orange points belong to clients that follow φ_2. Left: FL assumes that a single model can fit all clients' data distributions at the same time. Middle: federated XOR-problem. An insufficiently complex model is not capable of fitting all clients' data distributions at the same time. Right: if different clients' conditional distributions diverge, no single model can fit all distributions at the same time. In all cases, however, the data of clients belonging to the same cluster can be easily separated.

In this work, we argue that Assumption 1 is frequently violated in real FL applications, especially given the fact that, in FL, clients: 1) can hold arbitrary non-i.i.d. data, which cannot be audited by the centralized server due to privacy constraints and 2) typically run on limited hardware which puts restrictions on the model complexity. For illustration, consider the following practical scenarios.

Varying Preferences: Assume a scenario where every client holds a local data set of images of human faces and the goal is to train an "attractiveness" classifier on the joint data of all clients. Naturally, different clients will have varying opinions about the attractiveness of certain individuals, which corresponds to disagreeing conditional distributions on all clients' data. Assume, for instance, that one half of the client population thinks that people wearing glasses are attractive, while the other half thinks that those people are unattractive. In this situation, one single model will never be able to accurately predict the attractiveness of glasses-wearing people for all clients at the same time (confer also Fig. 1, right).

Limited Model Complexity: Assume that a number of clients are trying to jointly train a language model for next-word prediction on private text messages. In this scenario, the statistics of a client's text messages will likely vary a lot based on demographic factors, interests, and so on. For instance, text messages composed by teenagers will typically exhibit different statistics than those composed by elderly people. An insufficiently expressive model will not be able to fit the data of all clients at the same time (see Fig. 1, middle).

Presence of Adversaries: A special case of incongruence is given if a subset of the client population behaves in an adversarial manner. In this scenario, the adversaries could deliberately alter their local data distribution in order to encode arbitrary behavior into the jointly trained model, thus affecting the model decisions on all other clients and causing potential harm [6].

Federated Multitask Learning: The goal in federated multitask learning (FMTL) is to provide every client with a model that optimally fits its local data distribution. In all the above-described situations, the ordinary FL framework, in which all clients are treated equally and only one single global model is learned, is not capable of achieving this goal.

In order to incorporate the above-presented problems with incongruent data generating distributions, we suggest generalizing the conventional FL assumption.

Assumption 2 (CFL): There exists a partitioning $\mathcal{C} = \{c_1, \ldots, c_K\}$, $\dot{\bigcup}_{k=1}^{K} c_k = \{1, \ldots, M\}$ of the client population, such that every subset of clients $c \in \mathcal{C}$ satisfies the conventional FL assumption.

In this work, we present clustered FL (CFL), a novel algorithmic framework that is able to deal with FMTL problems that satisfy Assumption 2. By identifying the hidden clustering structure, CFL allows clients with similar data to profit from one another while minimizing the harmful interference between clients with dissimilar data. Our main contributions are given as follows.

Contributions:

1) We highlight an important practical limitation of conventional FL, namely, incongruent client data distributions (see Section I).
2) We derive a computationally efficient tool based on the cosine similarity between the clients' gradient updates that provably allows us to infer whether two members of the client population have different data generating distributions, thus making it possible for us to infer the clustering structure $\mathcal{C}$ (see Section II-A).
3) We address the question of when to cluster and derive a criterion that ensures that clustering only takes place in the incongruent case, where benefits can be expected (see Section II-B).
4) Based on these theoretical insights, we present the CFL algorithm (see Section II-C).
5) We investigate several practical concerns (varying client populations, training with formal privacy guarantees, and communication of weight-updates instead of gradients) and demonstrate that CFL can seamlessly adapt to these conditions/constraints (see Section IV).
6) We evaluate our theoretical findings on large-scale deep neural networks and demonstrate vast performance

TABLE I

SUMMARY OF SYMBOLS

Symbol	Explanation	Def.
M	number of clients	
$K \leq M$	number of data generating distributions	
$\theta \in \Theta$	neural network parameterization	
$d := \dim(\Theta)$	dimension of the parameter space	
$\varphi_k(x, y)$	data generating distribution	
$R_k(\theta)$	risk function associated with dist. φ_k	(9)
$D_i \sim \varphi_{I(i)}(x, y)$	data of client i, sampled from dist. $\varphi_{I(i)}$	(7)
$r_i(\theta)$	empirical risk function of client i	(8)
$\alpha_{i,j} \in [-1, 1]$	cosine similarity between gradient updates of clients i and j	(13)
$\varepsilon_1 > 0$	Federated Learning stopping criterion	(30)
$\varepsilon_2 > 0$	clustering stopping criterion	(31)
$\gamma_{max} \in [0, 1)$	clustering dissimilarity criterion	(36)
$c \in \mathcal{C}$	one cluster in the set of all clusters found by CFL	

improvements over the conventional FL algorithm when optimizing over incongruent clients (see Section V).

A summary of the most important symbols used throughout this article is provided in Table I. We additionally refer the reader to Table II, which gives a detailed comparison between our proposed method and the related methods for FMTL.

II. CLUSTERED FEDERATED LEARNING

In this section, we address the question of how to solve distributed learning problems that satisfy Assumption 2 (which generalizes the FL Assumption 1). This will require us to identify the correct partitioning $\mathcal{C}$, which, at first glance, seems like a daunting task, as, under the FL paradigm, the server does not have access to the clients' data, their data generating distributions, or any meta information thereof.

A. Cosine Similarity-Based Bipartitioning

An easier task than trying to directly infer the entire clustering structure $\mathcal{C}$ is to find a correct bipartitioning in the sense of the following definition.

Definition 1: Let $M \geq K \geq 2$ and

$$I : \{1, \ldots, M\} \to \{1, \ldots, K\}, \quad i \mapsto I(i) \tag{5}$$

be the mapping that assigns a client i to its data generating distribution $\varphi_{I(i)}$. Then, we call a bipartitioning $c_1 \dot{\cup} c_2 = \{1, \ldots, M\}$ with $c_1 \neq \emptyset$ and $c_2 \neq \emptyset$ correct if and only if

$$I(i) \neq I(j) \quad \forall i \in c_1, \ j \in c_2. \tag{6}$$

In other words, we call a bipartitioning correct if clients with the same data generating distribution end up in the same cluster. It is easy to see that the clustering $\mathcal{C} = \{c_1, \ldots, c_k\}$ can be obtained after exactly $K - 1$ correct bipartitions.

In the following, we will demonstrate that there exists an explicit criterion based on which a correct bipartitioning can be inferred. To see this, let us first look at the following simplified FL setting with M clients, in which the data on every client were sampled from one of two data generating distributions φ_1 and φ_2 such that

$$D_i \sim \varphi_{I(i)}(x, y). \tag{7}$$

Every client is associated with an empirical risk function

$$r_i(\theta) = \sum_{(x,y)\in D_i} l_\theta(f(x), y) \tag{8}$$

which approximates the true risk arbitrarily well if the number of data points on every client is sufficiently large

$$r_i(\theta) \approx R_{I(i)}(\theta). \tag{9}$$

For demonstration purposes, let us first assume equality in (9). Then, the FL objective becomes

$$F(\theta) := \sum_{i=1}^{M} \frac{|D_i|}{|D|} r_i(\theta) = a_1 R_1(\theta) + a_2 R_2(\theta) \tag{10}$$

with $a_1 = \sum_{i,I(i)=1} |D_i|/|D|$, $a_2 = \sum_{i,I(i)=2} |D_i|/|D|$, and $D = \bigcup_{i=1,\ldots,M} D_i$. Under standard assumptions, it has been shown [7], [8] that the FL optimization protocol described in (1) and (2) converges to a stationary point θ^* of the FL objective. In this point, it holds that

$$0 = \nabla F(\theta^*) = a_1 \nabla R_1(\theta^*) + a_2 \nabla R_2(\theta^*). \tag{11}$$

Now, we are in one of the two situations. Either it holds that $\nabla R_1(\theta^*) = \nabla R_2(\theta^*) = 0$; in that case, we have simultaneously minimized the risk of all clients. This means that φ_1 and φ_2 are congruent, and we have solved the distributed learning problem, or, otherwise, it has to hold

$$\nabla R_1(\theta^*) = -\frac{a_2}{a_1} \nabla R_2(\theta^*) \neq 0 \tag{12}$$

and φ_1 and φ_2 are incongruent. In this situation, the cosine similarity between the gradient updates of any two clients is given by

$$\begin{aligned} \alpha_{i,j} &: = \alpha(\nabla r_i(\theta^*), \nabla r_j(\theta^*)) := \frac{\langle \nabla r_i(\theta^*), \nabla r_j(\theta^*) \rangle}{\|\nabla r_i(\theta^*)\| \|\nabla r_j(\theta^*)\|} \\ &= \frac{\langle \nabla R_{I(i)}(\theta^*), \nabla R_{I(j)}(\theta^*) \rangle}{\|\nabla R_{I(i)}(\theta^*)\| \|\nabla R_{I(j)}(\theta^*)\|} \\ &= \begin{cases} 1, & \text{if } I(i) = I(j) \\ -1, & \text{if } I(i) \neq I(j). \end{cases} \end{aligned} \tag{13}$$

Consequently, a correct bipartitioning is given by

$$c_1 = \{i | \alpha_{i,0} = 1\}, \ c_2 = \{i | \alpha_{i,0} = -1\}. \tag{14}$$

This consideration provides us with the insight that, in a stationary solution of the FL objective θ^*, we can distinguish clients based on their hidden data generating distribution by inspecting the cosine similarity between their gradient updates. For a visual illustration of the result, we refer to Fig. 2.

If we drop the equality assumption in (9) and allow for an arbitrary number of data generating distributions K, we obtain the following generalized version of the result (13).

Theorem 1 (Separation Theorem): Let $D_1, \ldots, D_M$ be the local training data of M different clients, each data set sampled from one of K different data generating distributions $\varphi_1, \ldots, \varphi_K$ such that $D_i \sim \varphi_{I(i)}(x, y)$. Let the empirical risk on every client approximate the true risk at every stationary solution of the FL objective θ^* subject to

$$\|\nabla R_{I(i)}(\theta^*)\| > \|\nabla R_{I(i)}(\theta^*) - \nabla r_i(\theta^*)\| \tag{15}$$

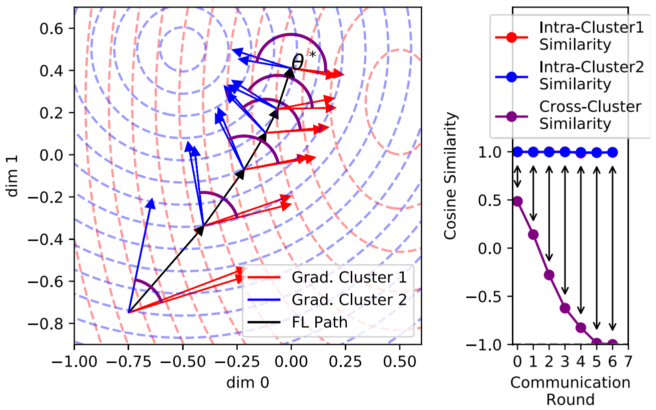


Fig. 2. Left: optimization path of FL with four clients, belonging to two different clusters with incongruent data distributions. FL converges to a stationary solution of the FL objective θ^* where the gradients of the two clients are of the positive norm and point into opposite directions (13). While the cosine similarity between gradient updates from the same cluster stays more or less constant throughout the federated training process, the cosine similarity between the gradient updates from different clusters quickly decreases.

and define

$$\gamma_i := \frac{\|\nabla R_{I(i)}(\theta^*) - \nabla r_i(\theta^*)\|}{\|\nabla R_{I(i)}(\theta^*)\|} \in [0,1). \tag{16}$$

Then, there exists a bipartitioning $c_1^* \dot\cup c_2^* = \{1,\dots,M\}$ of the client population such that the maximum similarity between the updates from any two clients from different clusters can be bounded from above according to

$$\begin{aligned}\alpha_{\text{cross}}^{\max} :&= \min_{c_1\dot\cup c_2=\{1,\dots,M\}} \max_{i\in c_1, j\in c_2} \alpha(\nabla r_i(\theta^*), \nabla r_j(\theta^*))\\ &= \max_{i\in c_1^*, j\in c_2^*} \alpha(\nabla r_i(\theta^*), \nabla r_j(\theta^*))\\ &\le \begin{cases} \cos\left(\frac{\pi}{K-1}\right)H_{i,j}+\sin\left(\frac{\pi}{K-1}\right)\sqrt{1-H_{i,j}^2} \\ \quad \text{if } H \ge \cos\left(\frac{\pi}{K-1}\right) \\ 1 \quad \text{else}\end{cases}\end{aligned} \tag{17}$$

with

$$H_{i,j} = -\gamma_i\gamma_j + \sqrt{1-\gamma_i^2}\sqrt{1-\gamma_j^2} \in (-1,1]. \tag{18}$$

At the same time, the similarity between the updates from clients that share the same data generating distribution can be bounded from below by

$$\alpha_{\text{intra}}^{\min} := \min_{\substack{i,j\\ I(i)=I(j)}} \alpha(\nabla_\theta r_i(\theta^*), \nabla_\theta r_j(\theta^*)) \ge \min_{\substack{i,j\\ I(i)=I(j)}} H_{i,j}. \tag{19}$$

Proof of Theorem 1 can be found in the Appendix.

Remark 1: In the case with two data generating distributions ($K=2$), the result simplifies to

$$\alpha_{\text{cross}}^{\max} = \max_{i\in c_1^*, j\in c_2^*} \alpha(\nabla_\theta r_i(\theta^*), \nabla_\theta r_j(\theta^*)) \le \max_{i\in c_1^*, j\in c_2^*} -H_{i,j} \tag{20}$$

for a certain partitioning, respective

$$\alpha_{\text{intra}}^{\min} = \min_{\substack{i,j\\ I(i)=I(j)}} \alpha(\nabla_\theta r_i(\theta^*), \nabla_\theta r_j(\theta^*)) \ge \min_{\substack{i,j\\ I(i)=I(j)}} H_{i,j} \tag{21}$$

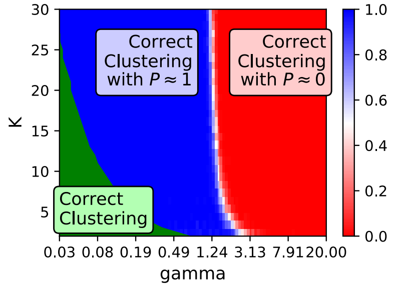


Fig. 3. Clustering quality as a function of the number of data generating distributions K and the relative approximation noise γ. For all values of K and γ in the green area, CFL will always correctly separate the clients (by Theorem 1). For all values of K and γ in the blue area, we find empirically that CFL will correctly separate the clients with probability close to 1.

for two clients from the same cluster. If additionally $\gamma_i = 0$ for all $i = 1,\dots,M$, then $H_{i,j} = 1$, and we retain result (13). From Theorem 1, we can directly deduce a correct separation rule.

Corollary 1: If, in Theorem 1, K and γ_i, $i = 1,\dots,M$, are in such a way that

$$\alpha_{\text{intra}}^{\min} > \alpha_{\text{cross}}^{\max} \tag{22}$$

then the partitioning

$$c_1, c_2 \leftarrow \arg\min_{c_1\dot\cup c_2=c} \left(\max_{i\in c_1, j\in c_2} \alpha_{i,j}\right) \tag{23}$$

is always correct in the sense of Definition 1.

Proof: Set

$$c_1, c_2 \leftarrow \arg\min_{c_1\dot\cup c_2=c} \left(\max_{i\in c_1, j\in c_2} \alpha_{i,j}\right) \tag{24}$$

and let $i \in c_1$ and $j \in c_2$; then

$$\alpha_{i,j} \le \alpha_{\text{cross}}^{\max} < \alpha_{\text{intra}}^{\min} = \min_{\substack{i,j\\ I(i)=I(j)}} \alpha_{i,j} \tag{25}$$

and hence, i and j cannot have the same data generating distribution. ■

This consideration leads us to the notion of the separation gap.

Definition 2 (Separation Gap): Given a cosine-similarity matrix α and a mapping from client to data generating distribution I, we define the notion of a separation gap

$$g(\alpha) := \alpha_{\text{intra}}^{\min} - \alpha_{\text{cross}}^{\max} \tag{26}$$

$$= \min_{\substack{i,j\\ I(i)=I(j)}} \alpha_{i,j} - \min_{c_1\dot\cup c_2=c}\left(\max_{i\in c_1, j\in c_2} \alpha_{i,j}\right). \tag{27}$$

Remark 2: By Corollary 1, the bipartitioning (23) will be correct in the sense of Definition 1 if the separation gap $g(\alpha)$ is greater than zero.

Theorem 1 gives an estimate for the similarities in the absolute worst case. In practice, $\alpha_{\text{intra}}^{\min}$ typically will be much larger, and $\alpha_{\text{cross}}^{\max}$ typically will be much smaller, especially if the parameter dimension $d := \dim(\Theta)$ is large. For instance, if we set $d = 10^2$ (which is still many orders of magnitude smaller than typical modern neural networks), $M = 3K$, and assume $\nabla R_{I(i)}(\theta^*)$ and $\nabla R_{I(i)}(\theta^*) - \nabla r_i(\theta^*)$ to be normally distributed for all $i = 1, \dots, M$; then, experimentally, we find (see Fig. 3) that

$$P[\text{“Correct Clustering”}] \geq P[g(\alpha) > 0] \approx 1 \tag{28}$$

even for large values of $K > 10$ and

$$\gamma_{\max} := \max_{i=1,\dots,M} \gamma_i > 1. \tag{29}$$

This suggests that using the cosine similarity criterion (23), we can readily find a correct bipartitioning c_1, c_2 even if the number of data generating distributions is high, and the empirical risk on every client's data is only a very loose approximation of the true risk.

B. Distinguishing Congruent and Incongruent Clients

In order to appropriately generalize the classical FL setting, we need to make sure to only split up clients with incongruent data distributions. In the classical congruent non-i.i.d. FL setting described in [1] where one single model can be learned, performance will typically degrade if clients with varying distributions are separated into different clusters due to the restricted knowledge transfer between clients in different clusters. Luckily, we have a criterion at hand to distinguish the two cases. To realize this, we have to inspect the gradients computed by the clients at a stationary point θ^*. When client distributions are incongruent, the stationary solution of the FL objective by definition cannot be stationary for the individual clients. Hence, the norm of the clients' gradients has to be strictly greater than zero. If, conversely, the client distributions are congruent, federated optimization will be able to jointly optimize all clients' local risk functions, and hence, the norm of the clients' gradients will tend toward zero as we are approaching the stationary point. Based on this observation, we can formulate the following criteria that allow us to make the decision whether to split or not: Splitting should only take place if it holds that both: 1) we are close to a stationary point of the FL objective

$$0 \leq \left\| \sum_{i=1,\dots,M} \frac{D_i}{|D|} \nabla_\theta r_i(\theta^*) \right\| < \varepsilon_1 \tag{30}$$

and 2) the individual clients are far from a stationary point of their local empirical risk

$$\max_{i=1,\dots,M} \|\nabla_\theta r_i(\theta^*)\| > \varepsilon_2 > 0. \tag{31}$$

We will also experimentally verify the clustering criteria (30) and (31) and give recommendations for the selection of the hyperparameters ε_1 and ε_2 in Section V-B.

In practice, we have another viable option to distinguish the congruent from the incongruent case. As splitting is only performed after FL has converged to a stationary point θ^*, the conventional FL solution is always computed as part of CFL. This means that if, after splitting up the clients, a degradation in the model performance is observed, it is always possible to fall back to the FL solution. In this sense, CFL will always improve the FL performance (or perform equally well at worst).

Algorithm 1 Optimal Bipartition

```
input: Similarity Matrix $\alpha \in [-1, 1]^{M,M}$
outout: bipartitioning $c_1, c_2$ satisfying (23)
$\cdot\ s \leftarrow \text{argsort}(-\alpha[:]) \in \mathbb{N}^{M^2}$
$\cdot\ \mathcal{C} \leftarrow \{\{i\}|i = 1, \dots, M\}$
for $i = 1, \dots, M^2$ do
    $\cdot\ i_1 \leftarrow s_i$ div $M$; $i_2 \leftarrow s_i$ mod $M$
    $\cdot\ c_{tmp} \leftarrow \{\}$
    for $c \in \mathcal{C}$ do
        if $i_1 \in c$ or $i_2 \in c$ then
            $\cdot\ c_{tmp} \leftarrow c_{tmp} \cup c$
            $\cdot\ \mathcal{C} \leftarrow \mathcal{C} \setminus c$
        end
    end
    $\cdot\ \mathcal{C} \leftarrow \mathcal{C} \cup \{c_{tmp}\}$
    if $|\mathcal{C}| = 2$ then
        return $\mathcal{C}$
    end
end
```

C. Algorithm

CFL recursively bipartitions the client population in a top–down way: starting from an initial set of clients $c = \{1, \dots, M\}$ and a parameter initialization θ_0, CFL performs FL according to Algorithm 2 in order to obtain a stationary solution θ^* of the FL objective. After FL has converged, the stopping criterion

$$0 \leq \max_{i \in c} \|\nabla_\theta r_i(\theta^*)\| < \varepsilon_2 \tag{32}$$

is evaluated. If criterion (32) is satisfied, we know that all clients are sufficiently close to a stationary solution of their local risk and, consequently, CFL terminates, returning the FL solution θ^*. If, on the other hand, criterion (32) is violated, this means that the clients are incongruent, and the server computes the pairwise cosine similarities α between the clients' latest transmitted updates according to (13). Next, the server separates the clients into two clusters in such a way that the maximum similarity between clients from different clusters is minimized

$$c_1, c_2 \leftarrow \arg \min_{c_1 \dot{\cup} c_2 = c} \left(\max_{i \in c_1, j \in c_2} \alpha_{i,j} \right). \tag{33}$$

This optimal bipartitioning problem at the core of CFL can be solved in $\mathcal{O}(M^3)$ using Algorithm 1. Since, in FL, it is assumed that the server has far greater computational power than the clients, the overhead of clustering will typically be negligible.

As derived in Section II-A, a correct bipartitioning can always be ensured if it holds that

$$\alpha_{\text{intra}}^{\min} > \alpha_{\text{cross}}^{\max}.$$

While the optimal cross-cluster similarity $\alpha_{\text{cross}}^{\max}$ can be easily computed in practice, computation of the intra cluster similarity requires knowledge of the clustering structure, and hence,

Algorithm 2 FL

```
Input: initial parameters $\theta$, set of clients $c$, $\varepsilon_1 > 0$
repeat
    for $i \in c$ in parallel do
        $\cdot\ \theta_i \leftarrow \theta$
        $\cdot\ \Delta\theta_i \leftarrow \mathrm{SGD}(\theta_i, D_i) - \theta_i$
    end
    $\cdot\ \theta \leftarrow \theta + \sum_{i\in c} \frac{|D_i|}{|D_c|}\Delta\theta_i$
until $\| \sum_{i\in c} \frac{|D_i|}{|D_c|}\Delta\theta_i \| < \varepsilon_1$
return $\theta$
```

Algorithm 3 CFL

```
Input: initial parameters $\theta$, set of clients $c$, $\gamma_{\max} \in [0, 1]$, $\varepsilon_2 > 0$
$\cdot\ \theta^* \leftarrow \mathrm{FederatedLearning}(\theta, c)$
$\cdot\ \alpha_{i,j} \leftarrow \frac{\langle \nabla r_i(\theta^*), \nabla r_j(\theta^*)\rangle}{\|\nabla r_i(\theta^*)\| \|\nabla r_j(\theta^*)\|}, i, j \in c$
$\cdot\ c_1, c_2 \leftarrow \arg\min_{c_1 \dot\cup c_2 = c}(\max_{i\in c_1, j\in c_2} \alpha_{i,j})$
$\cdot\ \alpha_{\text{cross}}^{\max} \leftarrow \max_{i\in c_1, j\in c_2} \alpha_{i,j}$
if $\max_{i\in c} \|\nabla r_i(\theta^*)\| \geq \varepsilon_2$ and $\sqrt{\frac{1-\alpha_{cross}^{\max}}{2}} > \gamma_{\max}$ then
    $\cdot\ \theta_i^*, i \in c_1 \leftarrow \mathrm{ClusteredFederatedLearning}(\theta^*, c_1)$
    $\cdot\ \theta_i^*, i \in c_2 \leftarrow \mathrm{ClusteredFederatedLearning}(\theta^*, c_2)$
else
    $\cdot\ \theta_i^* \leftarrow \theta^*, i \in c$
end
return $\theta_i^*, i \in c$
```

$\alpha_{\text{intra}}^{\min}$ can only be estimated using Theorem 1 according to

$$\alpha_{\text{intra}}^{\min} \geq \min_{\substack{i,j \\ I(i)=I(j)}} -\gamma_i\gamma_j + \sqrt{1-\gamma_i^2}\sqrt{1-\gamma_j^2} \tag{34}$$

$$\geq 1 - 2\gamma_{\max}^2. \tag{35}$$

Consequently, we know that the bipartitioning will be correct if

$$\gamma_{\max} < \sqrt{\frac{1-\alpha_{\text{cross}}^{\max}}{2}} \tag{36}$$

independent of the number of data generating distributions K. This criterion allows us to reject bipartitionings, based on our assumptions on the approximation noise $\gamma_{\max}$ (which is an interpretable hyperparameter).

If criterion (36) is satisfied, CFL is recursively reapplied to each of the two separate groups starting from the stationary solution θ^*. Splitting recursively continues on until (after at most $K-1$ recursions) none of the subclusters violate the stopping criterion anymore; at that point, all groups of mutually congruent clients $\mathcal{C} = \{c_1, \ldots, c_K\}$ have been identified, and the CFL problem characterized by Assumption 2 is solved. The communication burden of CFL, thus, increases at most linearly with the number of splits only after the FL solution is reached. The entire recursive procedure is presented in Algorithm 3. A schematic illustration is given in Fig. 4.

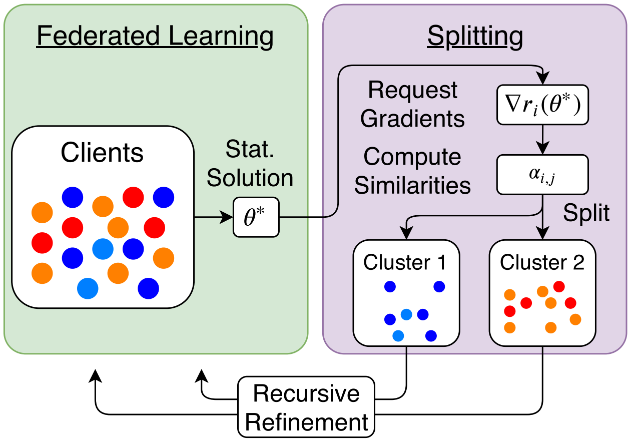


Fig. 4. Schematic overview over the CFL algorithm. By recursively bipartitioning the client population into subgroups of maximum dissimilarity, CFL produces a hierarchy of models of increasing specificity.

III. Related Work

FL [1], [2], [4], [5], [11], [12] is currently the dominant framework for distributed training of machine learning models under communication and privacy constraints. FL assumes the clients to be congruent, i.e., that one central model can fit all client's distributions at the same time. Different authors have investigated the convergence properties of FL in congruent i.i.d. and non-i.i.d. scenarios: Lin *et al.* [13], Sattler *et al.* [14], [15], and Zhao *et al.* [16] performd an empirical investigation and Li *et al.* [7], Sahu *et al.* [8], Jiang and Agrawal [17], and Yu *et al.* [18] proved convergence guarantees. As argued in Section I, conventional FL is not able to deal with the challenges of incongruent data distributions. Other distributed training frameworks [19]–[22] are facing the same issues.

The natural framework for dealing with incongruent data is multitask learning [23]–[25]. An overview of recent techniques for multitask learning in deep neural networks can be found in [26]. However, all of these techniques are applied in a centralized setting in which all data reside at one location and the server has full control over and knowledge about the optimization process. Smith *et al.* [9] presented MOCHA that extends the multitask learning approach to the FL setting, by explicitly modeling client similarity via a correlation matrix. However, their method relies on alternating biconvex optimization and is, thus, only applicable to convex objective functions and limited in its ability to scale to massive client populations. Corinzia and Buhmann [27] modeled the connectivity structure between clients and server as a Bayesian network and performed variational inference during learning. Although their method can handle nonconvex models, it is expensive to generalize to large federated networks as the client models are refined sequentially.

Finally, Ghosh *et al.* [10] proposed a clustering approach, similar to the one presented in this article. However, their method differs from ours in the following key aspects: most significantly, they use *l*2-distance instead of cosine similarity to determine the distribution similarity of the clients. This

TABLE II

QUALITATIVE COMPARISON BETWEEN THE METHODS FOR FMTL

Desideratum	**Smith et al. [9]**	**Ghosh et al. [10]**	**This work**
Works with arbitrary *non-convex* objective functions.	✗	✗	✓
Does not require changes to the FedAvg. protocol to be made.	✗	✓	✓
No major computational overhead for the clients.	-	✓	✓
Cluster number need not be known a priori.	✓	✗	✓
Theoretical guarantees on the clustering quality.	-	✓	✓
Can be implemented with formal privacy guarantees.	?	?	✓
Only performs clustering if necessary.	✗	✗	✓
Can handle varying client populations.	?	?	✓

✓ = satisfies property, ✗ = does not satisfy property, **?** = not investigated by the authors, **-** = does not apply

approach has the severe limitation that it only works if the client's risk functions are convex and the minima of different clusters are well separated. Their method also is not able to distinguish congruent from incongruent settings. This means that the method will incorrectly split up clients in the conventional congruent non-i.i.d. setting described in [1]. Furthermore, their approach is not adaptive in the sense that the decision of whether to cluster or not is made already after the first communication round. In contrast, our method can be applied to arbitrary FL problems with nonconvex objective functions. We also note that we have provided theoretical considerations that allow a systematic understanding of the novel CFL framework. For a detailed qualitative comparison between FMTL methods, we refer to Table II.

IV. IMPLEMENTATION CONSIDERATIONS

In this section, we consider the practical implementation details of our method. Concretely, we will demonstrate that CFL can be implemented without making modifications to the FL communication protocol and without compromising the privacy of the participating clients. We will also demonstrate that our method is flexible enough to handle client populations that vary over time.

A. Weight-Updates as Generalized Gradients

Theorem 1 makes a statement about the cosine similarity between the gradients of the empirical risk function. In FL, however, due to constraints on the communication budged of the client devices, instead, commonly weight-updates, as defined in (1), are computed and communicated [1]. In order to deviate as little as possible from the classical FL algorithm, it would, hence, be desirable to generalize result 1 to weight-updates. It is commonly conjectured (see [28]) that accumulated minibatch gradients approximate the full-batch gradient of the objective function. Indeed, for a sufficiently smooth loss function and low learning rate, a weight-update computed over one epoch approximates the direction of the true gradient by the Taylor expansion. Given a split

$$\bigcup_{\tau=0,\ldots,n_b-1} D_\tau = D \tag{37}$$

of a clients' data D into n_b disjoint batches, we have, after one epoch of SGD

$$\begin{aligned}\Delta\theta &= \mathrm{SGD}(\theta_0, D) - \theta_0\\ &= -\sum_{\tau=0}^{n_b-1} \eta\nabla_\theta r(\theta_\tau, D_\tau)\\ &\approx -\sum_{\tau=0}^{n_b-1} \eta\nabla_\theta r(\theta_0, D_\tau) = -\eta\nabla_\theta r(\theta_0, D)\end{aligned} \tag{38}$$

with

$$\theta_\tau = \theta_{\tau-1} - \eta\nabla_\theta r(\theta_{\tau-1}, D_{\tau-1}), \quad \tau > 0. \tag{39}$$

In the remainder of this work, we will compute cosine similarities between weight-updates instead of gradients according to

$$\alpha_{i,j} := \frac{\langle \Delta\theta_i, \Delta\theta_j\rangle}{\|\Delta\theta_i\|\|\Delta\theta_j\|}, \quad i, j \in c. \tag{40}$$

Our experiments in Section V will demonstrate that computing cosine similarities based on weight-updates, in practice, surprisingly achieves even better separations than computing cosine similarities based on gradients.

B. Preserving Privacy

Every machine learning model carries information about the data it has been trained on. For example, the bias term in the last layer of a neural network will typically carry information about the label distribution of the training data. Different authors have demonstrated that information about a client's input data ("x") can be inferred from the weight-updates that it sends to the server via model inversion attacks [29]–[33]. In privacy-sensitive situations, it might be necessary to prevent this type of information leakage from clients to the server with mechanisms, such as the ones presented in [3]. Luckily, CFL can be easily augmented with an encryption mechanism that achieves this end. As both the cosine similarity between two clients' weight-updates and the norms of these updates are invariant to orthonormal transformations P (such as the permutation of the indices)

$$\frac{\langle \Delta\theta_i, \Delta\theta_j\rangle}{\|\Delta\theta_i\|\|\Delta\theta_j\|} = \frac{\langle P\Delta\theta_i, P\Delta\theta_j\rangle}{\|P\Delta\theta_i\|\|P\Delta\theta_j\|} \tag{41}$$

a simple remedy is, for all clients, to apply such a transformation operator to their updates before communicating them to the server. After the server has averaged the updates from all clients and broadcasted the average back to the clients, they simply apply the inverse operation

$$\frac{1}{|c|}\sum_{i\in c}\Delta\theta_i = P^{-1}\left(\frac{1}{|c|}\sum_{i\in c}P\,\Delta\theta_i\right) \tag{42}$$

and the FL protocol can resume unchanged. Other multitask learning approaches require direct access to the client's data and, hence, cannot be used together with encryption, which means that CFL has a distinct advantage in privacy-sensitive situations.

C. Varying Client Populations and Parameter Trees

Up until now, we always made the assumption that all clients participate from the beginning of training. However, CFL is flexible enough to handle client populations that vary over time.

In order to incorporate this functionality, the server, while running CFL, needs to build a parameter tree $T = (V, E)$ with the following properties.

1) The tree contains a node $v \in V$ for every (intermediate) cluster c_v computed by CFL.
2) Both c_v and the corresponding stationary solution θ^*_v obtained by running the FL Algorithm 2 on cluster c_v are cached at node v.
3) At the root of the tree. v_{root} resides the FL solution over the entire client population with $c_{v_{\text{root}}} = \{1, \dots, M\}$.
4) If the cluster $c_{v_{\text{child}}}$ was created by bipartitioning the cluster $c_{v_{\text{parent}}}$ in CFL, then the nodes v_{parent} and v_{child} are connected via a directed edge $e \in E$.
5) At every edge $e(v_{\text{parent}} \rightarrow v_{\text{child}})$, the presplit weight-updates of the children clients

$$\Delta_e = \left\{\mathrm{SGD}\left(\theta^*_{v_{\text{parent}}}, D_i\right) - \theta_{v^*_{\text{parent}}} | i \in c_{v_{\text{child}}}\right\} \tag{43}$$

are cached.

An exemplary parameter tree is shown in Fig. 5. When a new client joins the training, it can get assigned to a leaf cluster by iteratively traversing the parameter tree from the root to a leaf, always moving to the branch that contains the more similar client updates according to Algorithm 4.

Another feature of building a parameter tree is that it allows the server to provide every client with multiple models at varying specificity. On the path from the root to leaf, the models get more specialized with the most general model being the FL model at the root. Depending on the application and context, a CFL client could switch between the models of different generality. Furthermore, a parameter tree allows us to ensemble multiple models of different specificity together. We believe that investigations along those lines are a promising direction for future research.

Putting all pieces from the previous sections together, we arrive at a protocol for general privacy-preserving CFL that is described in Algorithm 5.

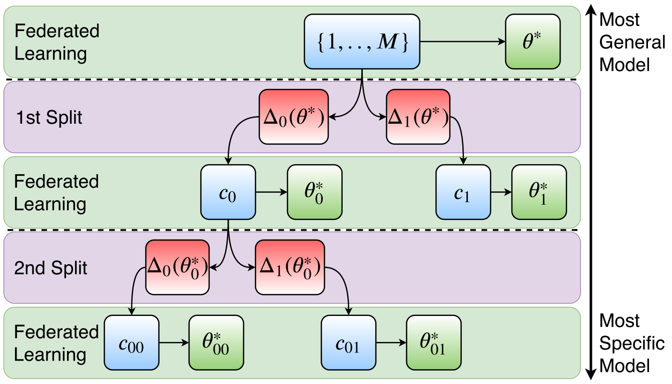


Fig. 5. Exemplary parameter tree created by CFL. At the root node resides, the conventional FL model, obtained by converging to a stationary point θ^* of the FL objective over all clients $\{1, \dots, M\}$. In the next layer, the client population has been split up into two groups, according to their cosine similarities, and every subgroup has again converged to a stationary point θ^*_0 respective θ^*_1. Branching continues recursively until no stationary solution satisfies the splitting criteria. In order to quickly assign new clients to a leaf model, at each edge e of the tree, the server caches the presplit weight-updates Δ_e of all clients belonging to the two different subbranches. This way, the new client can be moved down the tree along the path of the highest similarity.

Algorithm 4 Assigning New Clients to a Cluster

```
Input: new client with data $D_{new}$, parameter tree
  $T = (V, E)$
· $v \leftarrow v_{\text{root}}$
while $|\text{Children}(v)| > 0$ do
    · $v_0, v_1 \leftarrow \text{Children}(v)$
    · $\Delta\theta_{new} \leftarrow \text{SGD}(\theta^*_v, D_{new}) - \theta^*_v$
    · $\alpha_0 \leftarrow \max_{\Delta\theta\in\Delta_{(v\rightarrow v_1)}} \alpha(\Delta\theta_{new}, \Delta\theta)$
    · $\alpha_1 \leftarrow \max_{\Delta\theta\in\Delta_{(v\rightarrow v_2)}} \alpha(\Delta\theta_{new}, \Delta\theta)$
    if $\alpha_0 > \alpha_1$ then
        · $v \leftarrow v_0$
    else
        · $v \leftarrow v_1$
    end
end
return $c_v$, $\theta^*_v$
```

V. Experiments

A. Practical Considerations

In Section II-A, we showed that the cosine similarity criterion does distinguish different incongruent clients under three conditions: 1) FL has converged to a stationary point θ^*; 2) every client holds enough data subject to the empirical risk approximates the true risk; and 3) cosine similarity is computed between the full gradients of the empirical risk. In this section, we will demonstrate that, in practical problems, none of these conditions have to be fully satisfied. Instead, we will find that CFL is able to correctly infer the clustering structure even if clients only hold small data sets and are trained to an approximately stationary solution of the FL objective. Furthermore, we will see that cosine similarity can be computed between weight-updates instead of full gradients, which even improves the performance.

Algorithm 5 CFL With Privacy Preservation and Weight-Updates

```
input: initial parameters $\theta_0$, splitting parameters $\varepsilon_1, \varepsilon_2 > 0$, empirical risk approximation error bound $\gamma_{\max} \in [0, 1)$, number of local iterations/epochs $n$
outout: improved parameters on every client $\theta_i$
init: set initial clusters $\mathcal{C} = \{\{1, \dots, M\}\}$, set initial models $\theta_i \leftarrow \theta_0 \; \forall i = 1, \dots, m$, set initial update $\Delta\theta_c \leftarrow 0 \; \forall c \in \mathcal{C}$, clients exchange random seed to create permutation operator $P$ (optional, otherwise set $P$ to be the identity mapping)
while not converged do
    for $i = 1, \dots, M$ in parallel do
        Client $i$ does:
        · $\theta_i \leftarrow \theta_i + P^{-1}\Delta\theta_{c(i)}$
        · $\Delta\theta_i \leftarrow P(\mathrm{SGD}(\theta_i, D_i) - \theta_i)$
    end
    Server does:
    · $\mathcal{C}_{tmp} \leftarrow \mathcal{C}$
    for $c \in \mathcal{C}$ do
        · $\Delta\theta_c \leftarrow \frac{1}{|c|}\sum_{i \in c} \Delta\theta_i$
        if $\|\Delta\theta_c\| < \varepsilon_1$ and $\max_{i \in c} \|\Delta\theta_i\| > \varepsilon_2$ then
            · $\alpha_{i,j} \leftarrow \frac{\langle \Delta\theta_i, \Delta\theta_j \rangle}{\|\Delta\theta_i\|\|\Delta\theta_j\|}$
            · $c_1, c_2 \leftarrow \arg\min_{c_1 \dot{\cup} c_2 = c}(\max_{i \in c_1, j \in c_2} \alpha_{i,j})$
            · $\alpha_{\text{cross}}^{\max} \leftarrow \max_{i \in c_1, j \in c_2} \alpha_{i,j}$
            if $\gamma_{\max} < \sqrt{\frac{1-\alpha_{cross}^{\max}}{2}}$ then
                · $\mathcal{C}_{tmp} \leftarrow (\mathcal{C}_{tmp} \setminus c) \cup c_1 \cup c_2$
            end
        end
    end
    · $\mathcal{C} \leftarrow \mathcal{C}_{tmp}$
end
return $\theta$
```

In the experiments of this section, we consider the following FL setup. All experiments are performed on either the MNIST [34] or CIFAR-10 [35] data set using $M = 20$ clients, each of which belonging to one of $K = 4$ clusters. Every client is assigned an equally sized random subset of the total training data. To simulate an incongruent clustering structure, every clients' data are then modified by randomly swapping out two labels, depending on which cluster a client belongs to. For example, in all clients belonging to the first cluster, data points labeled as "1" could be relabeled as "7" and vice versa, in all clients belonging to the second cluster "3" and "5," could be switched out in the same way, and so on. This relabeling ensures that both $\varphi(x)$ and $\varphi(y)$ are approximately the same across all clients, but the conditionals $\varphi(y|x)$ diverge between different clusters. We will refer to this as "label-swap augmentation" in the following. In all experiments, we train multilayer convolutional neural networks and adopt a standard FL strategy with three local epochs of training. We report the separation gap (see Definition 2)

$$g(\alpha) := \alpha_{\text{intra}}^{\min} - \alpha_{\text{cross}}^{\max} \tag{44}$$

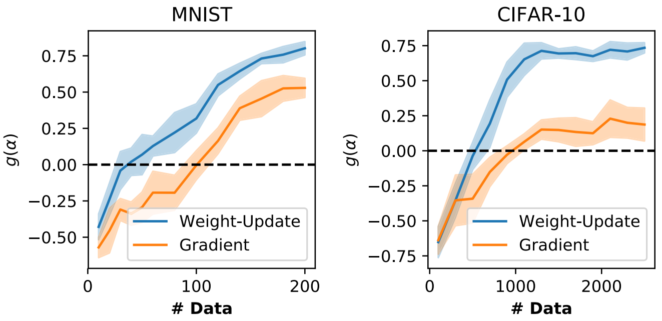


Fig. 6. Separation gap $g(\alpha)$ as a function of the number of data points on every client for the label-swap problem on MNIST and CIFAR. From Corollary 1, we know that CFL will always find a correct bipartitioning if $g(\alpha) > 0$. On MNIST, this is already satisfied if clients hold as little as 20 data points if weight-updates are used for the computation of the similarity α.

which, according to Corollary 1, tells us whether CFL will correctly bipartition the clients

$$g(\alpha) > 0 \Rightarrow \text{"Correct Clustering."} \tag{45}$$

Number of Data Points: We start out by investigating the effects of data set size on the cosine similarity. We randomly subsample from each client's training data to vary the number of data points on every client between 10 and 200 for MNIST and 100 and 2400 for CIFAR. For every different local data set size, we run FL for 50 communication rounds; after that, training progress has come mostly to halt, and we can expect to be close to a stationary point. After round 50, we compute the pairwise cosine similarities between the weight-updates and the separation gap $g(\alpha)$. The results are shown in Fig. 6. As expected, $g(\alpha)$ grows monotonically with increasing data set size. On the MNIST problem, as little as 20 data points on every client are sufficient to achieve correct bipartitioning in the sense of Definition 1. On the more difficult CIFAR problem, a higher number of around 500 data points is necessary to achieve correct bipartitioning.

Proximity to Stationary Solution: Next, we investigate the importance of proximity to a stationary point θ^* for the clustering. Under the same setting as in the previous experiment, we reduce the number of data points on every client to 100 for MNIST and 1500 for CIFAR and compute the pairwise cosine similarities and the separation gap after each of the first 50 communication rounds. The results are shown in Fig. 7. Again, we see that the separation quality monotonically increases with the number of communication rounds. On MNIST and CIFAR, as little as 10 communication rounds are necessary to obtain a correct clustering.

Weight-Updates Instead of Gradients: In both the above-mentioned experiments, we computed the cosine similarities α based on either the full gradients

$$\alpha_{i,j} = \frac{\langle \nabla_\theta r_i(\theta), \nabla_\theta r_j(\theta) \rangle}{\|\nabla_\theta r_i(\theta)\|\|\nabla_\theta r_j(\theta)\|} \quad \text{("Gradient")} \tag{46}$$

or federated weight-updates

$$\alpha_{i,j} = \frac{\langle \Delta\theta_i, \Delta\theta_j \rangle}{\|\Delta\theta_i\|\|\Delta\theta_j\|} \quad \text{("Weight-Update")} \tag{47}$$

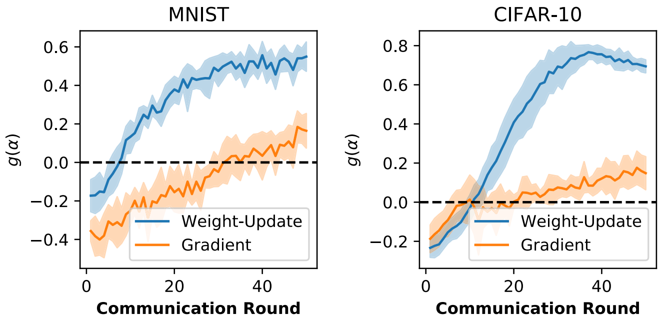


Fig. 7. Separation gap $g(\alpha)$ as a function of the number of communication rounds for the label-swap problem on MNIST and CIFAR. The separation quality monotonically increases with the number of communication rounds of FL. Correct separation in both cases is already achieved after around ten communication rounds if α is computed using weight-updates.

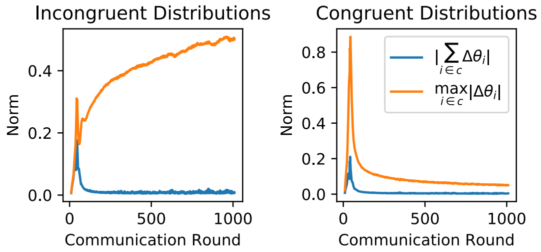


Fig. 8. Experimental verification of the norm criteria (31) and (30). Displayed is the development of gradient norms over the course of 1000 communication rounds of FL with two clients holding data from incongruent (left) and congruent distributions (right). In both cases, FL converges to a stationary point of $F(\theta)$, and the average update norm (30) goes to zero. In the congruent case, the maximum norm of the client updates (31) decreases along with the server update norm, while, in contrast in the incongruent case, it stagnates and even increases.

over three epochs. Interestingly, weight-updates seem to provide even better separation $g(\alpha)$ with fewer data points and at a greater distance to a stationary solution. This comes in very handy as it allows us to leave the FL communication protocol unchanged. In all following experiments, we will compute cosine similarities based on weight-updates instead of gradients.

B. Distinguishing Congruent and Incongruent Clients

In this section, we experimentally verify the validity of the clustering criteria (30) and (31) in an FL experiment on MNIST with two clients holding data from incongruent and congruent distributions. In the congruent case, client one holds all training digits “0” to “4,” and client two holds all training digits “5” to “9.” In the incongruent case, both clients hold a random subset of the training data, but the distributions are modified according to the “label swap” rule described earlier. Fig. 8 shows the development of the average update norm [see (30)] and the maximum client norm [see (31)] over the course of 1000 communication rounds. As predicted by the theory, in the congruent case, the maximum client norm converges to zero, while, in the incongruent case, it stagnates and even increases over time. In both cases, the average update norm tends to zero, indicating convergence to a stationary point (see Fig. 8).

The considerations in this section lead us to the following recommendations regarding the selection of the hyperparameters ε_1 and ε_2.

1) As the quality of the clustering improves with proximity to a stationary solution (see Fig. 7), the value for ε_1 should be set as small as the run-time restrictions allow. A good rule of thumb is to set it to around a tenth of the maximum average update norm $\varepsilon_1 \approx \max_t \|\Delta\theta_c^t\|/10$.
2) The value of ε_2 should be set in accordance with the number of available clients and/or prior knowledge on the heterogeneity of the client data. The smaller the value of ε_2, the more likely it is that the client population will be separated by CFL. In our experiments, we obtained good results by setting $\varepsilon_2 \in [\varepsilon_1, 10\varepsilon_1]$.

C. Clustered Federated Learning

In this section, we apply CFL as described in Algorithm 5 to different FL setups that are inspired by our motivating examples in the introduction. In all experiments, the clients perform three epochs of local training at a batch size of 100 in every communication round.

Image Classification on CIFAR-10: We split the CIFAR-10 training data randomly and evenly among $M = 20$ clients, which we group into $K = 4$ different clusters. All clients belonging to the same cluster apply the same random permutation $P_{c(i)}$ to their labels such that their modified training and test data are given by

$$\hat{D}_i = \{(x, P_{I(i)}(y))|(x, y) \in D_i\} \tag{48}$$

respective

$$\hat{D_i^{\text{test}}} = \{(x, P_{I(i)}(y))|(x, y) \in D^{\text{test}}\}. \tag{49}$$

The clients then jointly train a five-layer convolutional neural network on the modified data using CFL with three epochs of local training at a batch size of 100. Fig. 9 (top) shows the joint training progression: In the first 50 communication rounds, all clients train one single model together, following the conventional FL protocol. After these initial 50 rounds, training has converged to a stationary point of the FL objective, and the client test accuracies stagnate at around 20%. Conventional FL would be finalized at this point. At the same time, we observe (see Fig. 9, bottom) that a distinct gap $g(\alpha) = \alpha_{\text{intra}}^{\min} - \alpha_{\text{cross}}^{\max}$ has developed (①), indicating an underlying clustering structure. In communication round 50, the client population is, therefore, split up for the first time, which leads to an immediate 25% increase in validation accuracy for all clients belonging to the “purple” cluster that was separated out ②. Splitting is repeated in communication rounds 100 and 150 until all clusters have been separated, and $g(\alpha)$ has dropped to below zero in all clusters (③), which indicates that clustering is finalized. At this point, the accuracy of all clients has more than doubled the one achieved by the FL solution and is now at close to 60% ④. This underlines that, after standard FL, our novel CFL can detect the necessity for subsequent splitting and clustering that enable arriving at a significantly higher performance. In addition, the cluster structure

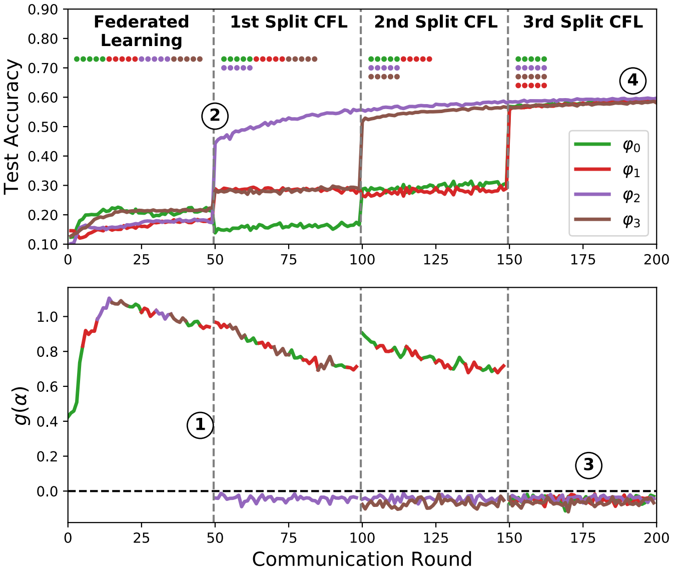


Fig. 9. CFL applied to the “permuted labels problem” on CIFAR with 20 clients and four different permutations. Top: accuracy of the trained model(s) on their corresponding validation sets. Bottom: separation gaps $g(\alpha)$ for all different clusters. After an initial 50 communication rounds, a large separation gap has developed, and a first split separates the purple group of clients, which leads to an immediate drastic increase of accuracy for these clients. In communication rounds 100 and 150, this step is repeated until all clients with incongruent distributions have been separated. After the third split, the model accuracy for all clients has more than doubled, and the separation gaps in all clusters have dropped to below zero, which indicates that the clustering is finalized.

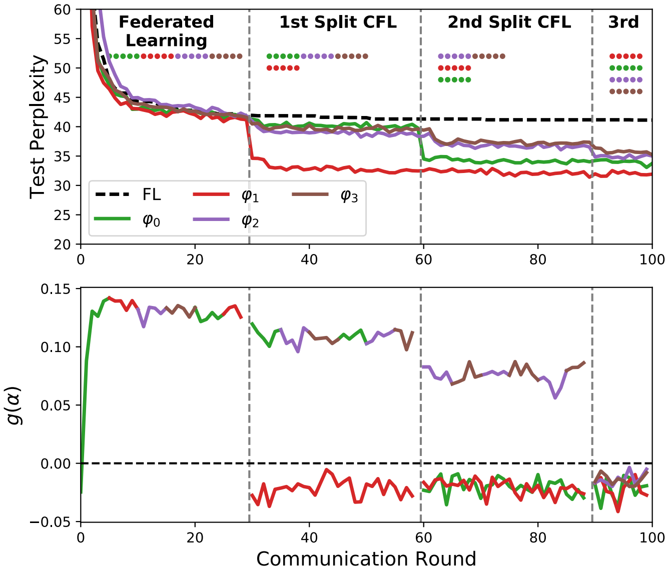


Fig. 10. CFL applied to the Ag-News problem. Top: perplexity achieved by the different clients on their local test set (lower is better). The clients are separated in communication rounds 30, 60, and 90. After the final separation, the perplexity of all clients on their local test set has dropped to less than 36, while the FL solution (black dotted) still stagnates at a perplexity of 42.

found can potentially be illuminating as it provides interesting insight into the composition of the complex underlying data distribution.

Language Modeling on Ag-News: The Ag-News corpus is a collection of 120000 news articles belonging to one of the four topics “World,” “Sports,” “Business,” and “Sci/Tech.” We split the corpus into 20 different subcorpora of the same size, with every subcorpus containing only articles from one topic and assign every corpus to one client. Consequently, the clients form four clusters based on what type of articles they hold. Every client trains a two-layer LSTM network to predict the next word on its local corpus of articles. Fig. 10 shows 100 communication rounds of multistage CFL applied to this distributed learning problem. As we can see, FL again converges to a stationary solution after around 30 communication rounds. At this solution, all clients achieve a perplexity of around 43 on their local test set. After the client population has been split up in communication rounds 30, 60, and 90, the four true underlying clusters are discovered. After the 100th communication round, the perplexity of all clients has dropped to less than 36. The FL solution, trained over the same amount of communication rounds, still stagnates at an average perplexity of 42.

VI. Conclusion

In this article, we presented CFL, a framework for FMTL that can improve any existing FL framework by enabling the participating clients to learn more specialized models. CFL makes use of our theoretical finding that (at any stationary solution of the FL objective) the cosine similarity between the weight-updates of different clients is highly indicative of the similarity of their data distributions. This crucial insight allows us to provide strong mathematical guarantees on the clustering quality under mild assumptions on the clients and their data, even for arbitrary nonconvex objectives.

We demonstrated that CFL can be implemented in a privacy-preserving way and without having to modify the FL communication protocol. Moreover, CFL is able to distinguish situations in which a single model can be learned from the clients’ data from those in which this is not possible and only separates clients in the latter case.

Our experiments on convolutional and recurrent deep neural networks show that CFL can achieve drastic improvements over the FL baseline in terms of classification accuracy/perplexity in situations where the clients’ data exhibit a clustering structure.

Finally, we note that our work also exposes a new privacy issue in FL as it demonstrates that the information about client data similarity can be inferred from their weight-updates. We argue that the privacy loss inflicted is tolerable in most situations as the mere knowledge of client similarity does not reveal anything about the clients’ data. Nevertheless, this fact should be considered when implementing FL for privacy-sensitive applications.

In future work, we will explore to what extent CFL can be used in conjunction with differential privacy mechanisms [36], [37], as well as parameter update compression methods [21], [38]–[40]. Furthermore, it would be interesting to study explainable AI techniques [41], [42] also in the context of (clustered) FL.

REFERENCES

[1] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," 2016, *arXiv:1602.05629*. [Online]. Available: http://arxiv.org/abs/1602.05629

[2] J. Konečný, H. B. McMahan, D. Ramage, and P. Richtárik, "Federated optimization: Distributed machine learning for on-device intelligence," 2016, *arXiv:1610.02527*. [Online]. Available: http://arxiv.org/abs/1610.02527

[3] K. Bonawitz *et al.*, "Practical secure aggregation for privacy preserving machine learning," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur. (CCS)*, 2017, pp. 1175–1191.

[4] K. Bonawitz *et al.*, "Towards federated learning at scale: System design," 2019, *arXiv:1902.01046*. [Online]. Available: http://arxiv.org/abs/1902.01046

[5] T. Li, A. Kumar Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," 2019, *arXiv:1908.07873*. [Online]. Available: http://arxiv.org/abs/1908.07873

[6] F. Sattler, K.-R. Müller, T. Wiegand, and W. Samek, "On the Byzantine robustness of clustered federated learning," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2020, pp. 8861–8865.

[7] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of FedAvg on non-IID data," 2019, *arXiv:1907.02189*. [Online]. Available: http://arxiv.org/abs/1907.02189

[8] T. Li, A. Kumar Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," 2018, *arXiv:1812.06127*. [Online]. Available: http://arxiv.org/abs/1812.06127

[9] V. Smith, C.-K. Chiang, M. Sanjabi, and A. S. Talwalkar, "Federated multi-task learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 4424–4434.

[10] A. Ghosh, J. Hong, D. Yin, and K. Ramchandran, "Robust federated learning in a heterogeneous environment," 2019, *arXiv:1906.06629*. [Online]. Available: http://arxiv.org/abs/1906.06629

[11] S. Caldas *et al.*, "LEAF: A benchmark for federated settings," 2018, *arXiv:1812.01097*. [Online]. Available: http://arxiv.org/abs/1812.01097

[12] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Trans. Intell. Syst. Technol.*, vol. 10, no. 2, pp. 12:1–12:19, 2019.

[13] T. Lin, S. U. Stich, K. Kshitij Patel, and M. Jaggi, "Don't use large mini-batches, use local SGD," 2018, *arXiv:1808.07217*. [Online]. Available: http://arxiv.org/abs/1808.07217

[14] F. Sattler, S. Wiedemann, K.-R. Müller, and W. Samek, "Sparse binary compression: Towards distributed deep learning with minimal communication," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2019, pp. 1–8.

[15] F. Sattler, S. Wiedemann, K.-R. Müller, and W. Samek, "Robust and communication-efficient federated learning from non-i.I.d. data," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Nov. 1, 2019, doi: 10.1109/TNNLS.2019.2944481.

[16] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated learning with non-IID data," 2018, *arXiv:1806.00582*. [Online]. Available: http://arxiv.org/abs/1806.00582

[17] P. Jiang and G. Agrawal, "A linear speedup analysis of distributed deep learning with sparse and quantized communication," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 2525–2536.

[18] H. Yu, S. Yang, and S. Zhu, "Parallel restarted SGD with faster convergence and less communication: Demystifying why model averaging works for deep learning," 2018, *arXiv:1807.06629*. [Online]. Available: http://arxiv.org/abs/1807.06629

[19] A. Koloskova, T. Lin, S. U. Stich, and M. Jaggi, "Decentralized deep learning with arbitrary communication compression," 2019, *arXiv:1907.09356*. [Online]. Available: http://arxiv.org/abs/1907.09356

[20] A. Koloskova, S. U. Stich, and M. Jaggi, "Decentralized stochastic optimization and gossip algorithms with compressed communication," 2019, *arXiv:1902.00340*. [Online]. Available: http://arxiv.org/abs/1902.00340

[21] S. U. Stich, J.-B. Cordonnier, and M. Jaggi, "Sparsified SGD with memory," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 4447–4458.

[22] V. Smith, S. Forte, C. Ma, M. Takac, M. I. Jordan, and M. Jaggi, "CoCoA: A general framework for communication-efficient distributed optimization," 2016, *arXiv:1611.02189*. [Online]. Available: http://arxiv.org/abs/1611.02189

[23] R. Caruana, "Multitask learning," *Mach. Learn.*, vol. 28, no. 1, pp. 41–75, 1997.

[24] L. Jacob, J.-P. Vert, and F. R. Bach, "Clustered multi-task learning: A convex formulation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2009, pp. 745–752.

[25] A. Kumar and H. Daume, III, "Learning task grouping and overlap in multi-task learning," 2012, *arXiv:1206.6417*. [Online]. Available: http://arxiv.org/abs/1206.6417

[26] S. Ruder, "An overview of multi-task learning in deep neural networks," 2017, *arXiv:1706.05098*. [Online]. Available: http://arxiv.org/abs/1706.05098

[27] L. Corinzia and J. M. Buhmann, "Variational federated multi-task learning," 2019, *arXiv:1906.06268*. [Online]. Available: http://arxiv.org/abs/1906.06268

[28] Y. Lin, S. Han, H. Mao, Y. Wang, and W. J. Dally, "Deep gradient compression: Reducing the communication bandwidth for distributed training," 2017, *arXiv:1712.01887*. [Online]. Available: http://arxiv.org/abs/1712.01887

[29] A. Bhowmick, J. Duchi, J. Freudiger, G. Kapoor, and R. Rogers, "Protection against reconstruction and its applications in private federated learning," 2018, *arXiv:1812.00984*. [Online]. Available: http://arxiv.org/abs/1812.00984

[30] B. Hitaj, G. Ateniese, and F. Perez-Cruz, "Deep models under the GAN: Information leakage from collaborative deep learning," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2017, pp. 603–618.

[31] M. Fredrikson, S. Jha, and T. Ristenpart, "Model inversion attacks that exploit confidence information and basic countermeasures," in *Proc. 22nd ACM SIGSAC Conf. Comput. Commun. Secur. (CCS)*, 2015, pp. 1322–1333.

[32] N. Carlini, C. Liu, Ú. Erlingsson, J. Kos, and D. Song, "The secret sharer: Evaluating and testing unintended memorization in neural networks," 2018, *arXiv:1802.08232*. [Online]. Available: http://arxiv.org/abs/1802.08232

[33] L. Melis, C. Song, E. De Cristofaro, and V. Shmatikov, "Exploiting unintended feature leakage in collaborative learning," 2018, *arXiv:1805.04049*. [Online]. Available: http://arxiv.org/abs/1805.04049

[34] Y. LeCun. (1998). *The MNIST Database of Handwritten Digits*. [Online]. Available: http://yann. lecun. com/exdb/mnist/

[35] A. Krizhevsky, V. Nair, and G. Hinton. (2014). *The CIFAR-10 dataset*. [Online]. Available: http://www.cs.toronto.edu/kriz/cifar.html

[36] C. Dwork, "Differential privacy: A survey of results," in *Proc. Int. Conf. Theory Appl. Models Comput.* Cham, Switzerland: Springer, 2008, pp. 1–19.

[37] M. Abadi *et al.*, "Deep learning with differential privacy," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2016, pp. 308–318.

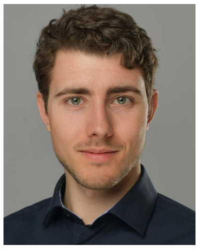
[38] D. Alistarh, D. Grubic, J. Li, R. Tomioka, and M. Vojnovic, "QSGD: Communication-efficient SGD via gradient quantization and encoding," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 1709–1720.

[39] S. Wiedemann *et al.*, "DeepCABAC: A universal compression algorithm for deep neural networks," *IEEE J. Sel. Topics Signal Process.*, vol. 14, no. 4, pp. 700–714, May 2020, doi: 10.1109/JSTSP.2020.2969554.

[40] S. Wiedemann, K.-R. Müller, and W. Samek, "Compact and computationally efficient representation of deep neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 3, pp. 772–785, Mar. 2020.

[41] W. Samek, G. Montavon, A. Vedaldi, L. K. Hansen, and K.-R. Müller, *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning* (Lecture Notes in Computer Science), vol. 11700. Cham, Switzerland: Springer, 2019, doi: 10.1007/978-3-030-28954-6.

[42] S. Lapuschkin, S. Wäldchen, A. Binder, G. Montavon, W. Samek, and K.-R. Müller, "Unmasking clever hans predictors and assessing what machines really learn," *Nature Commun.*, vol. 10, no. 1, p. 1096, Mar. 2019, doi: 10.1038/s41467-019-08987-4.



Felix Sattler received the B.Sc. degree in mathematics, the M.Sc. degree in computer science, and the M.Sc. degree in applied mathematics from Technische Universität Berlin, Berlin, Germany, in 2015 and 2018, respectively.

He is currently with the Machine Learning Group, Fraunhofer Heinrich Hertz Institute, Berlin. His research interests include distributed machine learning, neural networks, and multi-task learning.

Klaus-Robert Müller (Member, IEEE) studied physics in Karlsruhe, Germany, from 1984 to 1989. He received the Ph.D. degree in computer science from Technische Universität Karlsruhe, Karlsruhe, in 1992.

After completing a post-doctoral position at GMD FIRST, Berlin, Germany, he was a Research Fellow with The University of Tokyo, Tokyo, Japan, from 1994 to 1995. In 1995, he founded the Intelligent Data Analysis Group, GMD FIRST (later Fraunhofer FIRST), and directed it until 2008. From 1999 to 2006, he was a Professor with the University of Potsdam, Potsdam, Germany. He has been a Professor of computer science with Technische Universität Berlin since 2006, where he is co-directing the Berlin Big Data Center. His research interests are intelligent data analysis, machine learning, signal processing, and brain–computer interfaces.

Dr. Müller was elected to be a member of the German National Academy of Sciences-Leopoldina in 2012 and the Berlin Brandenburg Academy of Sciences in 2017 and an External Scientific Member of the Max Planck Society in 2017. He was awarded the 1999 Olympus Prize by the German Pattern Recognition Society, DAGM. In 2006, he received the SEL Alcatel Communication Award. In 2014, he was granted the Science Prize of Berlin awarded by the Governing Mayor of Berlin. In 2017, he received the Vodafone Innovation Award.

Wojciech Samek (Member, IEEE) received the Dipl.-Inf. degree in computer science from the Humboldt University of Berlin, Berlin, Germany, in 2010, and the Dr. rer. nat. degree from the Technical University of Berlin, Berlin, in 2014.

He was a Visiting Researcher with the NASA Ames Research Center, Mountain View, CA, USA. He is currently the Head of the Machine Learning Group, Fraunhofer Heinrich Hertz Institute, Berlin. He is also a PI with the Berlin Institute for the Foundations of Learning and Data (BIFOLD), Berlin, a member of the European Laboratory for Learning and Intelligent Systems (ELLIS), and an Associate Faculty with the DFG Graduate School BIOQIC, Berlin. He has coauthored more than 100 peer-reviewed journal articles and conference papers, predominantly in the areas of deep learning, explainable AI, neural network compression, and federated learning.

Dr. Samek is also an Editorial Board Member of *Digital Signal Processing*, *PLOS ONE*, and the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS and an elected member of the IEEE MLSP Technical Committee. During his studies, he was awarded scholarships from the Studienstiftung des deutschen Volkes and the DFG Research Training Group GRK 1589/1. He is also a part of the MPEG-7 Part 17 Standardization and was an organizer of special sessions, workshops, and tutorials on topics such as explainable AI and federated learning at top-tier machine learning and signal processing conferences.