A Survey on Cluster-based Federated Learning

Omar El-Rifai, Michael Ben Ali, Imen Megdiche, André Peninou, Olivier Teste

A Survey on Cluster-based Federated Learning

OMAR EL-RIFAI, Université Toulouse, UT3, IRIT, CNRS, France
MICHAEL BEN ALI, Université Toulouse, UT3, IRIT, CNRS, France
IMEN MEGDICHE, INU Champollion, ISIS Castres, IRIT, CNRS, France
ANDRÉ PENINOU, Université Toulouse, UT2J, IRIT, CNRS, France
OLIVIER TESTE, Université Toulouse, UT2J, IRIT, CNRS, France

As the industrial and commercial use of Federated Learning (FL) has expanded, so has the need for optimized algorithms. In settings were FL clients' data is non-independently and identically distributed (non-IID) and with highly heterogeneous distributions, the baseline FL approach seems to fall short. To tackle this issue, recent studies, have looked into personalized FL (PFL) which relaxes the implicit single-model constraint and allows for multiple hypotheses to be learned from the data or local models. Among the personalized FL approaches, cluster-based solutions (CFL) are particularly interesting whenever it is clear - through domain knowledge - that the clients can be separated into groups. In this paper, we study recent works on CFL, proposing: i) a classification of CFL solutions for personalization; ii) a structured review of literature iii) a review of alternative use cases for CFL.

CCS Concepts: • **General and reference** → **Surveys and overviews**; • **Computing methodologies** → **Machine learning**; • **Information systems** → **Clustering**; • **Security and privacy** → *Privacy-preserving protocols.*

Additional Key Words and Phrases: Federated Learning, Clustered Federated Learning, non-IID

ACM Reference Format:
Omar El-Rifai, Michael Ben Ali, Imen Megdiche, André Peninou, and Olivier Teste. 2025. A Survey on Cluster-based Federated Learning. 1, 1 (January 2025), 22 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Federated Learning (FL) was designed to train a global federated model on a network of devices without sharing raw data across different client devices (see Figure 1). The original assumptions of FL are that devices

Authors' addresses: Omar El-Rifai, Omar.El-Rifai@irit.fr, Université Toulouse, UT3, IRIT, CNRS, Toulouse, France; Michael Ben Ali, Michael-Eddy.Ben-Ali@irit.fr, Université Toulouse, UT3, IRIT, CNRS, Toulouse, France; Imen Megdiche, imen.megdiche@irit.fr, INU Champollion, ISIS Castres, IRIT, CNRS, Castres, France; André Peninou, andre.peninou@irit.fr, Université Toulouse, UT2J, IRIT, CNRS, Toulouse, France; Olivier Teste, olivier.teste@irit.fr, Université Toulouse, UT2J, IRIT, CNRS, Toulouse, France.

Manuscript submitted to ACM

are massively distributed, the data is non-independently and identically distributed (non-IID), unbalanced, and network communication is limited [29]. As such, one of the key claims of the original FL framework is that it is efficient to train a deep learning model in a heterogeneous setting. In this context the heterogeneity can be twofold, namely, system heterogeneity and data heterogeneity.

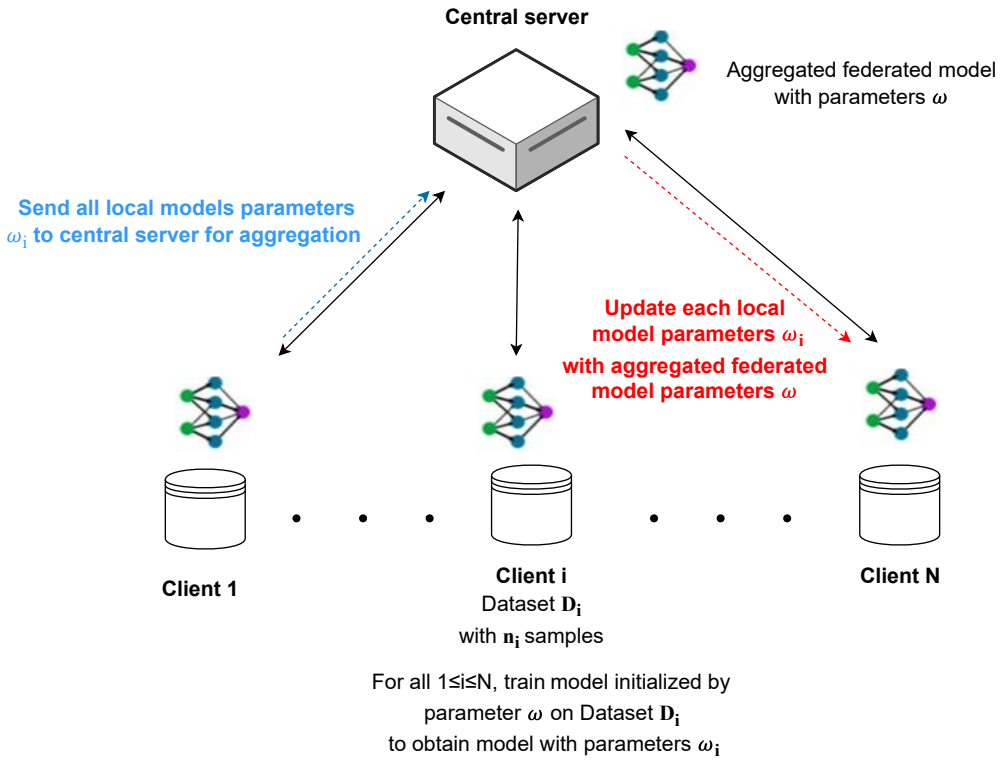
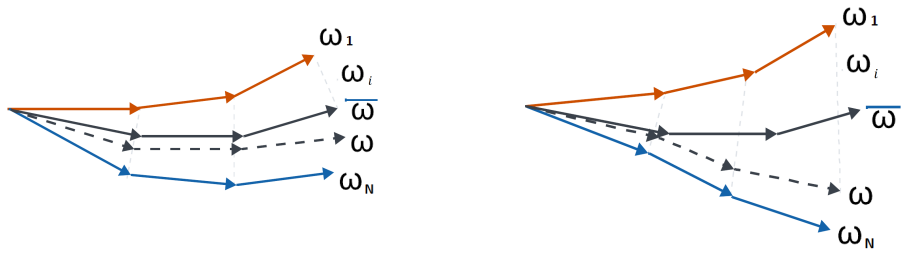


Fig. 1. Federated learning classical architecture

All the same, consequent studies have demonstrated several shortcomings in the baseline approach [17]. In particular, when faced with heterogeneity, FL solutions have been shown to be inadequate in many practical applications [33, 43]. A recent survey on device-heterogenous FL [33] argues that the sparsity of current real-world production applications of FL is due to heterogeneity challenges. Similarly, in [43] the authors argue that FL "heavily relies on the assumption that all the participants share the same network structure and possess similar data distributions" and then expand on the types of possible heterogeneities and the current literature which addresses these problems.

In this paper, we look in details into the problem of training a single global solution when the datasets have heterogeneous distributions among the clients in FL configuration. The difficulty of such scenario is illustrated in Figure 2 whereby the optimization direction of a model trained on centralized clients' data

is compared to those of the federated scenario with both IID and non-IID data. In the non-IID scenario, the global federated models weights ω drift from the non-federated model weights $\bar{\omega}$, trained on the same data but in a centralized way, resulting in a larger gap between them than in the case of homogeneous distributions. This results was shown in [44] which noted a significant reduction in accuracy in scenarios with highly skewed non-IID data.



(a) IID scenario: All clients i, $1 \leq i \leq N$, train their local model's parameters ω_i, on homogeneous data. In this scenario, the federated model parameters ω are close to a hypothetical non-federated model with parameters $\bar{\omega}$ trained on the same data but in a centralize manner

(b) Non-IID scenario: All clients i, $1 \leq i \leq N$, train their local model's parameters ω_i, on heterogeneous data. In this scenario, the federated model parameters ω diverges from the non-federated model parameters $\bar{\omega}$ trained on the same data but in a centralize manner

Fig. 2. Illustration of non-IID problem in FL [44]

As a response to this challenge, Personalized FL (PFL) literature [20, 38] emerged with different approaches that relax the single model constraint and instead train several specialized federated models to cater for data heterogeneity. Recent trend in PFL literature uses cluster-based federated learning (CFL) methods for personalization which unlike other methods, explicitly prescribe the number of federated models to train.

1.1 Organization of the Content of the Paper

The field of CFL for personalization has been very active in recent years with studies focusing on finding efficient algorithmic solutions and accounting for operational constraints [9, 30]. However, as the field grows, so do the use cases and type of solutions proposed. So much so that it is presently difficult to have an overview of the different problems addressed by it and to objectively compare the solutions. To have a better understanding of CFL, we start this survey with some basic FL definition in Section 2 and others specific to CFL for personalization in Section 3. This allow us to have a common framework for a foundational understanding that can be used as reference.

Then with these definitions in mind, we tackle three gaps in the CFL literature aiming at facilitating future research. First, we review the different algorithms found in the literature and categorize the solutions into three types which helps us compare results and single out contributions in Section 4.

Then, in Section 5.1, we shed light on an important ambiguity in the definition and use of non-IID terminology which makes studies comparison difficult. In response we propose a precise taxonomy which could summarize our findings and facilitate reproducibility.

Finally in Section 5.2, we identify the different use cases of clustering in the context of FL. For although personalization has been the main focus of cluster-based studies in FL, the problems of "client selection" and "attack prevention" have also been tackled in some studies using clustering. We also provide a comprehensive list of notations used throughout this paper in Table 1 for ease of reference.

2 FEDERATED LEARNING IN HETEROGENEOUS ENVIRONMENT

In order to have a clear frame of reference for further discussions, we begin by transcribing the FL framework described in [29] and pin down the heterogeneity problem as alluded to in that study.

2.1 Federated Learning Formulation

We consider a network with N clients [1] with $N \geq 2$ and one central server. Each client $i \in I = \{1, \ldots, N\}$ has a local dataset D_i which can not be shared with others. The goal of the federated model is to find a set of parameters $\omega \in \mathbb{R}^d$, where $d \in \mathbb{N}^*$ is the number of parameters of the neural network, that minimize an global objective function $f(\omega)$ using each local dataset.

We refer to this objective as finding an optimal federated model and write it as such:

$$\min_{\omega \in \mathbb{R}^d} f(\omega) := \frac{1}{N} \sum_{i=1}^{N} f_i(\omega) \tag{1}$$

where $f_i(\omega)$ is the local objective function associated to each client i typical of neural network settings (For a machine learning problem, we typically take $f_i(\omega) = \mathbb{E}_{(X,Y)\sim D_i}[L(X, Y, \omega)]$, that is, the expected value of the loss function L calculated with feature and target (X, Y) following the distribution of dataset D_i with model of parameters $\omega \in \mathbb{R}^d$.) Generally, using a larger number of data sample leads to more accurate and robust models.

Since the data cannot be shared across clients, the authors in [29], propose to iteratively optimize $f_i(\omega)$ on local datasets and then to aggregate the models on the server using a weighted average function. We assume that each client $i \in I$ over which the data is partitioned have a local dataset D_i where the number of the dataset samples is $|D_i| = n_i$. Thus, we can re-write the objective function (1) as:

$$\min_{\omega \in \mathbb{R}^d} f(\omega) := \sum_{i=1}^{N} \frac{n_i}{\sum_{j=1}^{N} n_j} f_i(\omega) \tag{2}$$

where $\sum_{j=1}^{N} n_j$ correspond to the total number of samples across all clients.

[1] In this study, we use clients and devices interchangeably

Notations	Description
N	Number of clients in the FL ecosystem, $N \geq 2$
I	$I = \{1, \ldots, N\}$ set of all clients index in the FL ecosystem
i	Index of client ($i \in I$)
D_i	Local dataset of client i noted as $D_i = \left\{(x_j^{(i)}, y_j^{(i)}) \mid 1 \leq j \leq n_i\right\}$, where $(x_j^{(i)})_{1 \leq j \leq n_i}$ and $(y_j^{(i)})_{1 \leq j \leq n_i}$ represents respectively the dataset features and target
n_i	Number of samples in D_i. We note $\lvert D_i \rvert = n_i$
d	Number of parameters of considered neural networks, $d \in \mathbb{N}^*$
ω	Federated neural network model parameters with $\omega \in \mathbb{R}^d$
ω_i	Local neural network parameters of client i with $\omega_i \in \mathbb{R}^d$
$\bar{\omega}$	Neural network model parameters $\bar{\omega} \in \mathbb{R}^d$ in a non-federated ecosystem trained on dataset $D = \bigcup_{i=1}^{N} D_i$
f	Federated model objective function
f_i	Local objective function for client i defined to optimize a local model on D_i
$(\Omega, \mathcal{F}, \mathcal{P})$	Probability space where Ω is the sample space, representing all possible outcomes of the random experiment associated with sampling from the dataset $D = \bigcup_{i=1}^{N} D_i$ $\mathcal{F}$ the associate sigma algebra and $\mathcal{P}$ the probability measure
X	Random variable representing the possible outcome of features of all clients datasets $(D_i)_{i \in I}$ in the federated learning ecosystem. In this case, each $(x_j^{(i)})_{1 \leq j \leq n_i}$ for $i \in I$ is a realization of X.
Y	Random variable representing the possible outcome of targets of all clients datasets $(D_i)_{i \in I}$ in the federated learning ecosystem. In this case, each $(y_j^{(i)})_{1 \leq j \leq n_i}$ for $i \in I$ is a realization of Y.
(X, Y)	Joint random variable of the possible outcome of features and targets of all clients datasets $(D_i)_{i \in I}$ in the federated learning ecosystem. We have $(X, Y) : \Omega \mapsto D := \bigcup_{i=1}^{N} D_i$
$\mathcal{P}_i(X)$	Marginal distribution of dataset D_i features
$\mathcal{P}_i(Y)$	Marginal distribution of dataset D_i target
$\mathcal{P}_i(X, Y)$	Joint distribution of features and targets of dataset D_i
$\mathcal{P}_i(X \lvert Y)$	Conditional distribution of features conditioned by target of dataset D_i
$\mathcal{P}_i(Y \lvert X)$	Conditional distribution of targets conditioned by features of dataset D_i
K	Number of clusters, $K \leq N$
C_k	k-th cluster with $k \in \{1, \ldots, K\}$; a cluster C_k will be considered as a subset of I. Meaning that $C_k \subseteq I$ and $I = \bigcup_{k=1}^{K} C_k$
F_k	Objective function to optimize the federated model on cluster C_k
$\text{dist}(\cdot, \cdot)$	Generic distance metric
$\mathbb{1}_{i \in C_k}$	Equal to 1 if client i is in cluster C_k, 0 otherwise
μ_k	Cluster C_k representative point essential for cluster formation; a vector of size d calculated using weights ω_i for all $i \in C_k$
η_i	Metadata parameters of client i; may correspond to local data statistics, distribution statistics or any type of metadata representation of client i (see Section 3.2.1)
δ_k	Cluster C_k metadata-based representative point essential for cluster formation; This point share the same type as metadata parameters η_i for all clients $i \in C_k$

Table 1. Summary of notations

Equation (2) reflects the Federated Average (FedAvg) protocol and is used in the original framework [29]. We note that in the original paper, the data distribution of each client as well as the devices computation and communication capacities are considered to be heterogeneous.

2.2 System and Statistical Heterogeneity

The FL network consists of user-owned non-standardized devices. As such, the statistical property of the data and hardware capacities of the devices are naturally heterogeneous. Heterogeneity affects systems (Section 2.2.1) as well as the data (Section 2.2.2).

2.2.1 System Heterogeneity. In massively distributed environments, such as a network of user-owned devices, the heterogeneity of the systems implies different computation and communication capacities. In addition, devices can also experience downtime due to connectivity problem or simply being shut down by the user. This heterogeneity is problematic if synchronization needs to happen before aggregating the models by a central server. To alleviate this problem the FL protocol advances a client selection mechanism whereby a subset of clients is appointed for training in each round. System heterogeneity is not the focus of PFL approaches which instead focus on statistical heterogeneity.

2.2.2 Statistical Heterogeneity: non-iid. The baseline FedAvg approach [29] was trained on both an IID and a non-IID setting and is shown to be effective in one specific type of heterogeneous context which is the label distribution skew scenario. In this non-IID case, the authors explore partitioning the MNIST dataset by labels (digits) prior to being distributed over clients. However, there exists other forms of non-IID data situations (statistical heterogeneity) which we summarize below as proposed in the taxonomy of [17].

For each client $i \in I = \{1, \ldots, N\}$, we assume that its associate local dataset D_i which is noted as a samples set of the form

$$D_i = \left\{ (x_j^{(i)}, y_j^{(i)}) \mid 1 \leq j \leq n_i \right\}$$

where $(x_j^{(i)})_{1 \leq j \leq n_i}$ represents the features and $(y_j^{(i)})_{1 \leq j \leq n_i}$ represents the target and with $|D_i| = n_i$.

Using this notation, we define the random variables X representing the possible outcome of features and Y representing the possible outcome of the targets of all client datasets $(D_i)_{i \in I}$ (meaning that in this case, for all $i \in I$, $(x_j^{(i)})_{1 \leq j \leq n_i}$ are realizations of X and $(y_j^{(i)})_{1 \leq j \leq n_i}$ are realizations of Y).

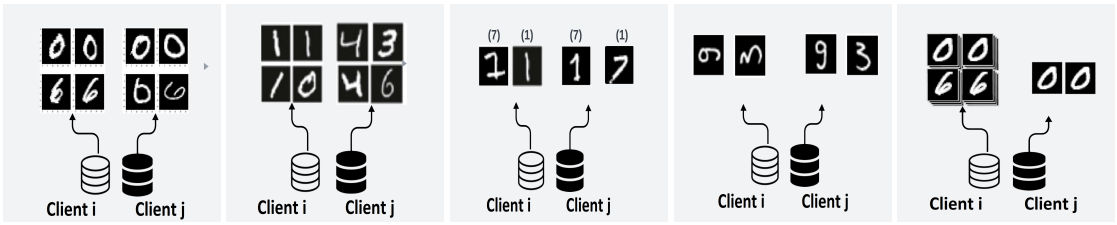
If we denote the probability space as $(\Omega, \mathcal{F}, P)$ where Ω is the sample space, representing all possible outcomes of the random experiment associated with sampling from the dataset $D = \bigcup_{i=1}^{N} D_i$, $\mathcal{F}$ its associate sigma algebra and $\mathcal{P}$ the probability measure, then the joint random variable (X, Y) can be defined as :

$$(X, Y) : \Omega \mapsto D := \bigcup_{i=1}^{N} D_i$$

Each clients dataset D_i is represented by its local joint distribution $\mathcal{P}_i(X, Y)$ which we can factor as follows using Bayes' rule:

$$\mathcal{P}_i(X, Y) = \mathcal{P}_i(Y)\mathcal{P}_i(X|Y) = \mathcal{P}_i(X)\mathcal{P}_i(Y|X) \tag{3}$$

where $\mathcal{P}_i(X)$ the marginal distribution of the features (respectively $\mathcal{P}_i(Y)$ the target) and $\mathcal{P}_i(Y|X)$ the distribution of the target conditioned by the features (respectively $\mathcal{P}_i(X|Y)$ the distribution of features conditioned by the target) for client $i \in I$.



(a) Feature Distribution Skew (b) Label Distribution Skew (c) Concept Shift on Labels (d) Concept Shift on Features (e) Quantity Skew

Fig. 3. Illustration of non-IID categories for two client i and j with samples from the MNIST dataset

We thus define 5 types of non-IID data situations (based on the taxonomy of [17]) each with a corresponding exemple from the MNIST dataset in Figure 3 (The letter items from the categories correspond to the letter items in the Figure). For client $i \in I$ and client $j \in I$ with $i \neq j$, we can consider D_i and D_j as non-IID if one or multiple of the following conditions apply:

a. **Feature distribution skew**: $\mathcal{P}_i(X) \neq \mathcal{P}_j(X)$ even if $\mathcal{P}_i(Y|X) = \mathcal{P}_j(Y|X)$. In the federated MNIST dataset (FEMNIST), client's dataset contain images written by a specific person. As each person's handwriting is different, the features can be considered as skewed. For exemple, we can have that the "6" is written differently for client i and j like in Figure 3.a.
b. **Label distribution skew**: $\mathcal{P}_i(Y) \neq \mathcal{P}_j(Y)$ even if $\mathcal{P}_i(X|Y) = \mathcal{P}_j(X|Y)$. We can create this scenario by unevenly distributing the digits to the clients such that one client has a majority "1"s, another a majority of "4"s and so on as illustrated in Figure 3.b.
c. **Concept shift on labels**: (same features, different label) $\mathcal{P}_i(Y|X) \neq \mathcal{P}_j(Y|X)$ even if $\mathcal{P}_i(X) = \mathcal{P}_j(X)$. Although more likely in dataset reflecting personal preferences, we can image such a scenario in the MNIST dataset if for instance the digits "1" in a user's dataset resembles digit "7" in another user's (see Figure 3.c.).
d. **Concept shift on features**: [2] (same label, different features) $\mathcal{P}_i(X|Y) \neq \mathcal{P}_j(X|Y)$ for clients i and j even if $\mathcal{P}_i(Y) = \mathcal{P}_j(Y)$. This can be artificially attained in MNIST if we rotate the images in certain devices. (see Figure 3.d.)

[2]Shortened to "Concept Drift" in [17]

e. **Quantity Skew**: the volume of data differs significantly among clients. Meaning that $|D_i| \ll |D_j|$ or $|D_j| \ll |D_i|$. (see Figure 3.e.)

3 CLUSTERING FOR PERSONALIZED FEDERATED LEARNING

A host of different methods exist besides clustering for personalization in FL. For instance, some works have studies Transfer Learning, Multi-task Learning, Meta Learning, Knowledge Distillation, and Parameters Decoupling [20, 38]. But clustering approaches stand out from other methods for two main reasons. First, unlike other approaches which work on client-level, clustering forms groups and enables the categorization of clients into functionally equivalent clusters. As such clustering is useful when groups are implicit in the network given domain knowledge. For instance, when training a language model we can assume that adolescents and adults might have different syntactical preferences. Second, clustering is a well studies field with readily available implementations in several languages and easily explainable results. Although this argument might seem anecdotal from a purely scientific perspective, it is often a determinant factor when choosing a technology in real-life applications.

3.1 Cluster-Based Federated Learning Formulation

In this Section, we define a generic cluster-based federated optimization formulation, and then, present the three type of solution approaches we found in the literature : namely, "Models and Gradients-based (server-side) Clustering" in Section 3.2.1, "Training Loss-based (client-side) Clustering" in Section 3.2.2 and "Metadata-based Clustering" in Section 3.2.3.

Let clients $i \in \{1, \ldots, N\}$ having datasets D_i each containing $|D_i| = n_i$ data samples which are supposed to span together K different data distributions $\mathcal{P}_1, \ldots, \mathcal{P}_K$ such that $1 \leq K \leq N$. That is, the number of total distribution is less than or equal to the number of clients. This implies that the clients can be partitioned into K disjoint clusters $C_1, \ldots, C_K$ where the clients' datasets distributions are homogeneous intra-cluster[3]. For the sake of simplicity in notations, each cluster will be considered as a disjoint partition of $I = \{1, \ldots, N\}$ with the notation $i \in C_k$ meaning that the client of index i is inside cluster C_k (i.e. : $I = \bigcup_{k=1}^{K} C_k$). We would then have K different clusters and for $k \in \{1, \ldots, K\}$ a corresponding objective function F_k to optimize the federated model on cluster C_k of the form:

$$\min_{\omega \in \mathbb{R}^d} F_k(\omega) := \sum_{i \in C_k} \frac{n_i}{\sum_{j \in C_k} n_j} f_i(\omega) \tag{4}$$

where $\sum_{j \in C_k} n_j$ correspond to the total number of samples of all clients inside cluster C_k and :

$$f_i(\omega) = \mathbb{E}_{(X,Y) \sim D_i}[L_i(X, Y, \omega)] \ \forall i \in C_k \tag{5}$$

[3]In some recent works, the disjonction hypothesis is relaxed.

that is the expected values of the expected value of the local loss function L_i calculated with feature and target (X, Y) following the distribution of dataset D_i with model of parameters $\omega \in \mathbb{R}^d$.

Based on this formulation , we describe the three main cluster based personalization solution approaches which we found in the literature. We base our taxonomy below on how clusters are computed.

3.2 Categorization of Cluster-Based Federated Learning

3.2.1 Models and Gradients-based (server-side) Clustering. First, and arguably the most intuitive approach, is referred to as gradient-based (or model-based) clustering [2, 3, 5, 9, 10, 15, 26, 27, 30, 36, 42]. The studies following this approach solve Equation (4) by only using models weights or gradient information. This strategy has the advantage of requiring minimum set-up compared to the baseline FL approach. In the most straightforward version, the central server calculates clusters based on the Lloyd algorithm (k-means) applied to the models weights and then cluster-specific federated models are optimized separately for each cluster.

Specifically, we first need to define a cluster associate representative point $\mu_k \in \mathbb{R}^d$ essential for the formation of each cluster C_k for all $k \in \{1, \ldots, K\}$. The central server will determine each client $i \in \{1, \ldots, N\}$ cluster membership by minimizing the distance between its model parameters $\omega_i \in \mathbb{R}^d$ and cluster-associated representative point. The formulation can be expressed as k-means-like optimization as follows:

$$\min_{\mu_k, k \in \{1,\ldots,K\}} \sum_{k=1}^{K} \sum_{i=1}^{N} \mathbb{1}_{i \in C_k} \operatorname{dist}(\omega_i, \mu_k) \tag{6}$$

This approach (or minor variants) will be used in many studies with **dist** being a generic distance function between models parameters or their gradients and $\mathbb{1}_{i \in C_k} = 1$ if client i is in cluster C_k, 0 otherwise. Several studies rely on the l_2 distance between models parameters as the used metric which is common in k-means but can lead to incorrect data partitions as shown in [36]. Since the cluster assignment is determined by the central server, throughout the remainder of this paper, this type of clustering method will be denominate as "server-side" clustering.

3.2.2 Training Loss-based (client-side) Clustering. As an alternative to server-side clustering, some studies delegated the computational burden of determining cluster identities to client nodes [8, 14, 18, 22, 28, 34, 39]. In these studies, clients are presented with several possible models (usually initialized by the central server) and evaluate their cluster membership by calculating the training loss on these different models.

Formally, let $\mu_k \in \mathbb{R}^d$ be the cluster representative point of cluster C_k for all $k \in \{1, \ldots, K\}$ essential for the formation of each cluster. These representative points can be initialized by the central server. Each clients $i \in \{1, \ldots, N\}$ will evaluate their training data on the K different models, each initialized with parameters μ_k and update theirs models parameters ω_i with the cluster representative point which minimizes their

local loss function :

$$\omega_i = \underset{\mu_k, k \in \{1,\dots,K\}}{\arg\min} \; \mathbb{E}_{(X,Y)\sim D_i}[L_i(X, Y, \mu_k)], \quad \forall i \in \{1, \dots, N\} \tag{7}$$

that is the representative point μ_k that minimize the expected values of the expected value of the local loss function L_i calculated with feature and target (X, Y) following the distribution of dataset D_i with model of parameters μ_k.

Once an initial, cluster assignment has been chosen, cluster representative point can be updated and clients recalculate the assignment iterativelly (usually the representative point will be updated by a weighted average of all cluster members models parameters [14] by the central server). Client-side clustering can be both an advantage or a disadvantage depending on the context. The authors in [14], argue that "one of the major advantages of [their] algorithm is that it does not require a centralized clustering algorithm, and thus significantly reduces the computational cost at the center machine". It has also been argued in [26], that client-based clustering "posts high communication and computation overheads because the selected nodes will spend more resources for receiving and running multiple global models". Since the cluster assignment in this case is determined by the client, throughout the remainder of this paper, this type of clustering method will be denominate as "client-side" clustering.

3.2.3 Metadata-based Clustering. The third category of studies rely on exogenous information to partition clients [6, 7, 21]. Specifically, metadata such as local data statistics or distribution statistics are transferred to the central server in order to cluster clients with similar distributions. One problem with using exogenous information is the risk of transmitting potentially sensitive data across the network. Workarounds often include homomorphic encryption or differential privacy [41].

For each client $i \in \{1, \dots, N\}$, we define metadata parameters η_i and then we define for each cluster C_k a metadata-based representative point δ_k for all $k \in \{1, \dots, K\}$ essential for the cluster formation of the same type as the metadata parameters. Each client i membership is determine by using a formulation similar to that of Equation (6):

$$\min_{\delta_1,\dots,\delta_K} \sum_{k=1}^{K} \sum_{i=1}^{N} \mathbb{1}_{i \in C_k} \text{dist}(\eta_i, \delta_k) \tag{8}$$

where **dist** is a generic distance between the defined metadata and $\mathbb{1}_{i \in C_k} = 1$ if client i is in cluster C_k, 0 otherwise. Since the cluster assignment is determined by the clients metadata, throughout the remainder of this paper, this type of clustering method will be denominate as "metadata-based" clustering.

4 REVIEW OF MAIN SOLUTIONS FROM LITERATURE

The literature on cluster-based PFL has been growing critically in the last couple of years. But despite new studies addressing advanced operational and computational constraints, the essential ideas in server-side and client-side solutions were developed early on. We discuss these foundational papers in Section 4.1. Then,

in Section 4.2, we describe follow up studies which took the methods a step further either by optimizing the algorithms or accounting for more realistic contexts. Finally in Section 4.3 we look into papers which propose methods at the intersection of clustering and other approaches. These later papers highlight the properties as well as the limitations of clustering and as such are important to discuss here. We include in this last section the metadata-based solution which violate the strict FL frame of reference which only allows sharing of gradients, model weights and local number of clients samples.

In addition to our description of the papers, we give a high level synthesis of the papers in Table 2. The column "Clustering Category" identifies the category of the solution approach as presented in Section 3. "Nomenclature" is the name (or acronym) of the solution as presented in the paper itself or as it is referred to in later papers."Clustering Algorithm" and "Clustering Evaluation Metric" are details on the clustering method used.

4.1 Foundational Studies

When working with FL, an important precept is the privacy of user's data. The central server should only have access to user's models or gradients but not to the raw data. However, clustering group of clients with similar data distributions using only gradients information seems like a dubious undertaking at first sight. And indeed, the problem of the efficacy of clusters based on gradients instead of raw data was raised early on in [36]. In that paper, the authors propose a cosine similarity measure for gradients to partition nodes into clusters using k-means. By doing so, they prove, that under mild conditions, the cosine similarity partitioning criteria is optimal compared to the usual euclidean distance. In other words, clusters obtained with this partitioning scheme are identical to those that would have been obtained if the server had access to the raw data. Similarly, [3] also describes a server-side method based on gradient but uses a hierarchical clustering algorithm. Their method is argued to be applicable to a wider range of non-IID settings than in [36]. Further, the solution in [36] is achieved by multiple rounds of bipartite separation, each requiring the FL algorithm to run until convergence. As promising as the aforementioned results are, they suffer from an important limitation: being computationally intensive on the server. These computational costs, along with other shortcomings, led to the development of client-side solutions.

Client-side solutions were initially proposed in [28] and [14]. Instead of leaving all the computational burden on to the central server, their methods rely on client nodes evaluating several models locally and choosing the model that best fits their data (c.f. 3.2.2). The study in [28] provides generalization guarantees to the solution while in [14] the authors develop a convergence rate analysis for the solution termed Iterative Federated Clustering Algorithm (IFCA).

Finally, the last category of studies (namely, metadata based studies) suggest transferring metadata to the server to help finding optimal clusters. An example of such study is [7] whereby clustering is performed in two phases. In the first phase, clusters of data are formed locally on each client dataset and their local centroids are computed. Then, these centroids are taken as metadata and transferred to the central

server and based on their similarity, client clusters are computed. This approach is shown to optimize the communication costs as it requires only one additional round of communication with the central server.

Similarly, in [6] the server creates clusters based on infrastructure-specific features (same access point/ coordinates etc.). However, the authors assume that users can communicate and transfer models amongst them which is not a standard assumption in the classical FL set-up.

Reference	Clustering Category	Nomenclature	Clustering Algorithm	Clustering Evaluation Metric
Ghosh et al. 2019 [13]	Server-side	N/A	Trimmed k-means	L_2 norm
Sattler et al. 2020 [36]	Server-side	CFL	Recursive bi-partitioning	Cosine similarity
Ghosh et al. 2020 [14]	Client-side	IFCA	Client loss minimisation	Local training loss
Mansour et al. 2020 [28]	Client-side	HypCluster	Client Loss minimisation	Local training loss
Briggs et al. 2020 [3]	Server-side	FL+HC	Agglomerative Hierarchical Clustering	L_1 /L_2 /Cosine similarity
Dennis et al. 2021 [7]	Metadata-based	k-Fed	K-means	L_2
Duan et al. 2021 [9]	Server-side	FedGroup	K-means/ Hierarchical clustering	MADD [35] / Custom Metric
Duan et al 2021 [10]	Server-side	FlexCFL	K-means/ Hierarchical clustering	MADD[35] / Custom Metric
Kim et al. 2021 [18]	Client-side	Dynamic Gan-Based Clustering	3 Steps Clustering: 1) Cluster GAN [31] 2) Client loss minimization 3) Divisive clustering	Local training loss
Li et al. 2021 [22]	Client-side	N/A	Client loss minimisation	Local training loss
Xiao et al. 2021 [42]	Server-side	CFMTL	Hierarchical clustering	L_2
Luo et al. 2021 [27]	Server-side	CAFL	Agglomerative Hierarchical Clustering	L_2
Ruan et al. 2022 [34]	Client-side	FedSoft	Client loss minimisation	Local training loss
Wolfrath et al. 20202 [41]	Metadata-based	HACCS	Density based Algorithm (OPTICS)	Hellinger Distance
Gong et al. 2022 [15]	Client-side	AutoCFL	Voting scheme	L_2
Castellon et al. 2022 [5]	Server-side	FLIC	Louvain algorithm	Cosine similarity
Long et al. 2023 [26]	Client-side	FeSEM	K-means like	N/A
Liang et al. 2023 [25]	Server-side	FedEOC	K-means	Cosine similarity
Augello et al. 2023 [2]	Server-side	DCFL	Affinity propagation algorithm [12]	Custom distance
Mehta et al. 2023 [30]	Server-side	FLACC	Agglomerative Hierarchical Clustering	Cosine similarity

Table 2. High level summary of reviewed papers for foundationnal studies (ordered by year)

4.2 Solving for Operational Constraints

While the studies above laid the ground algorithmically, they were followed up by papers which paid more attention to details initially overlooked. These follow-up studies proposed improvements to handle additional operational as well as computational constraints. We organize our review of the remaining literature by "improvement" category. Namely, in Section 4.2.1 we list papers which have worked on

optimizing the computational and/or communication costs. Then in Section 4.2.2, we look into studies which examine the framework's security and propose solutions to mitigate the possible attack risks. In 4.2.3, we include studies which add a mechanism to integrate network dynamics. These dynamics can be new clients joining the network or having the distribution of clients evolve over time. Finally in 4.2.4 we look into approaches which expand on the problem of identifying an optimal number of clusters. All of these results are summarized in Table 3 for ease of reference. We additionally add two columns in Table 3 (Namely "Hard/Soft" and "One-shot/iterative"). The first column defines whether the clustering algorithm uses hard clustering (i.e mutually exclusive clusters) or soft clustering and the second whether the framework is iterative or clustering is done in one shot (more communication efficient).

4.2.1 Computational and Communication Efficiency. In terms of server-side methods using gradient information, the framework in [9] offers computational improvements over the work in [36]. The framework first decomposes the model updates matrix into m directions using singular value decomposition (SVD) then computes the cosine similarity using these new directions. Similarly, in [5] the authors tackle the problem of communication efficiency. Instead of having dedicated communication rounds for clustering as in [36] and [3], this paper suggests capitalizing on the past weight updates of non-participant of the current round of FL. A similarity matrix between models is built iteratively on the server where only current round participants will update the matrix and past similarity of non-participants are kept, then clustering is performed using this matrix. The work of [27] follow up on the works of [3], but seeks to improve accuracy. As such, the authors trains multiple models at each client to make the gradients robust to noise. Conversely, the work of [30] prefers to favour computation speed and chooses to cluster clients without waiting for full model convergence. This form of greedy clustering reduces the computational burden and is shown to be efficient in many cases. In [15] the authors start with the observation that the computation burden is exacerbated when the number of samples per client is highly imbalanced. To counter that, they propose a solution which favors balancing clients with large datasets across clusters. Finally in [25], the authors propose to use a single layer instead of entire models to calculate the similarity between models.

On the other hand, papers using training-loss based solutions also improved on the initial proposals with follow up studies. In particular, the authors in [15] compute distance between models using partial weights similarity to decrease computation costs. Weights chosen are those in the layers closest to the output as they were shown to be most representative. In [26] layer-wise matching aggregation is chosen for faster convergence (cf. [40]).

In a different manner [22, 34] root their approaches in the client-side, training-loss based approach inspired by [14]. Specifically, in [34] the authors propose soft clustered federated learning which relaxes the assumption of a unique data distribution per client. Every local dataset is considered as a mixture of multiple source distributions. On updating the local models, clients optimizes a proximal local objective function (inspired by FedProx [23]) which accounts for local data as well as clusters' information which makes for more realistic data distribution scenarios.

Reference	Computation and Communication Tweaks	Security Tweaks	Newcomers	Number of Clusters	Hard/Soft	One-shot / Iterative
Ghosh et al. 2019 [13]	N/A	Outlier detection	N/A	User defined parameter	Hard	One shot
Sattler et al. 2020 [36]	N/A	Encryption compatible	Newcomers traverse saved clusters-parameter's tree to find best fitting cluster	Threshold Based	Hard	Iterative
Ghosh et al. 2020 [14]	N/A	N/A	N/A	User defined parameter	Hard	Iterative
Mansour et al. 2020 [28]	N/A	N/A	N/A	User defined parameter	Hard	iterative
Briggs et al. 2020 [3]	N/A	N/A	N/A	Stop merging when the distance between cluster is below a chosen threshold	Hard	One shot
Dennis et al. 2021 [7]	One shot communication client-server communication	N/A		User defined parameter	Hard	One shot
Duan et al. 2021 [9]	1) Custom clustering metric 2) SVD decomposition on model updates	N/A	Newcomers assigned by optimization direction	User defined parameter	Hard	Iterative
Duan et al. 2021 [10]	1) Same optimization steps as [9] 2) New client migration strategy.	N/A	Detect client data distribution shift over time. Reassign to new cluster if needed	User defined parameter	Hard	Iterative
Kim et al. 2021 [18]	N/A	N/A	Newcomers perform function evaluation	Initially unsupervised by GAN then threshold based (divisive clustering)	Hard	iterative
Li et al. 2021 [22]	Relax hard clustering constraint	N/A	N/A	User defined parameter	Soft	Iterative
Xiao et al. 2021 [42]	1) Allow inter-clusters communication for more generalizable models 2) Client subset selection strategy	Two party secure computation	N/A	User defined parameter	Hard	One shot
Luo et al. 2021 [27]	1) Stochastic training for models robustness 2) Client subset selection strategy	N/A	Newcomers work as a single new cluster	Distance threshold based	Hard	iterative
Ruan et al. 2022 [34]	Client subset selection strategy	N/A	N/A	User defined parameter	Soft	iterative
Wolfrath et al. 2022 [41]	Client subset selection strategy	Differential privacy	Newcomers handled dynamically like other nodes	Density based clustering parameters.	Hard	Iterative.
Gong et al. 2022 [15]	1) Optimize "number of training epochs" variable 2) Models similarity calculated using subset of model weights	N/A	N/A	Model similarity matrix based stopping threshold	Hard	One shot
Castellon et al. 2022 [5] Long et al. 2023 [26]	Client subset selection strategy Layer-wise matching aggregation for faster convergence	Median aggregation N/A	N/A N/A	Based on Louvain algorithm User defined parameter	Hard Hard	Iterative iterative
Liang et al. 2023 [25]	1) Optimize local gradients updates 2) Models similarity calculated using subset of model weights	N/A	N/A	User defined parameter with minimum size for clusters	Hard	One shot
Augello et al. 2023 [2]	N/A	Differential privacy	Newcomers given average global model then assigned to the nearest cluster after local training	Threshold based	Hard	Iterative
Mehta et al. 2023 [30]	Greedy clustering strategy	N/A	N/A	Threshold based	Hard	Iterative

Table 3. Scope summary of reviewed papers

Following the same line of reasoning that [34], the authors in [22] propose to combine loss-based cluster assignment approach of IFCA [14] with a soft clustering method. Instead of choosing one cluster identity exclusively, each client selects a set of K clusters to which its data belongs to. At the server, the cluster models are updated based on all the clients which have data belonging to it. But unlike in [34] where weights indicate the importance of each cluster model for each client, here client models are a simple average over the K cluster models that yield the lowest loss.

4.2.2 Security Concerns. As data privacy is at the center of FL, accounting for the security of exchanges is important. Consequently, some papers introduced security tweaks to protect the FL framework from malicious attacks. For instance in [2] the authors proposes a clustering metric which, unlike cosine similarity, remains "effective under noisy conditions typical of differential privacy". The clustering metric is a mixture of euclidean distance and a divergence measure. [18] builds on the study in [28] and enhances it by using GAN models which purportedly to help with data privacy using differential privacy.

4.2.3 Newcomers and dynamic. Some studies include an algorithm to integrate new clients in the network dynamically or allow clients to change clusters dynamically. For instance, in [9] newcomer devices are assigned to the clusters that are most closely related to their optimization direction (proximity is evaluated using the normalized cosine dissimilarity between group and newcomer model update). Further in [10] the authors pursue the previous work [9] by also including a client migration strategy. That is, in dynamic environments where there can be data distribution shifts, clients can be reassigned to a different cluster if their data distribution at communication round t evolves beyond a certain threshold from the distribution at communication round $t-1$. The Wasserstein distance between the previous and current local distribution is used to evaluate the shift. In a different way, the authors of [30] propose an algorithm that is robust in dynamic environments where not all clients are online at all time and ensure that the algorithm works when only a fraction of clients participate in a training round. Finally, in [18] the authors uses a cluster division procedure to change the number of clusters dynamically.

4.2.4 Number of Clusters. Two studies ([30] and [2]) improve on existing approaches by clustering clients using clustering algorithms that do not require a user defined number of clusters to be specified (see Table 2). As this parameters can drastically change the results and is difficult to set, avoiding it by using a parameter free algorithm can be essential for real-life situations. In [8] a different approach is taken. The study extend the client-side method in [14] to adapt it to a dynamic environment. As such, they do not specify a fixed number of clusters a priori but instead create, at each round, a new clusters whose representative point is based on the client with the highest loss and allow other clients to migrate to this new cluster if need be. At the end of each round, cluster which are no longer relevant will be removed.

4.3 Edge Studies

The solution strategies we presented in the previous paragraphs rely on clustering exclusively to personalize FL models. And despite the advantages of cluster-based methods, they still suffer from some limitations. For instance, contrary to other personalization approaches, the models created by clusters typically lead to siloed, group-specific models, which are difficult to generalize. Consequently, other studies have approached the problem of statistical heterogeneity with a mix of methodologies to tackle the limitation of strictly clustering-based approaches.

The authors in [42] propose a federated multi-task learning (MTL) framework whereby clustering is performed as an initial step for multi task learning. This clustering step groups similar clients together and helps select a subset of clients for each round. In [26], a joint optimization approach is used for inter cluster communication. As such, the optimization problem proposed seeks to find both federated cluster models and local models simultaneously which allows both to mutually influence each other and find the best compromise. A general distance function instead of the common l_2 distance is used.

5 REVIEW OF APPLICATIONS

The CFL for personalization solutions reviewed in Section 4 allowed us to organize papers into solution categories and describe how frameworks evolved to include more operational constraints. Solutions categorization is important to understand when to use one solution over another from a purely formal perspective. In this section, we continue the categorization of the literature but this time from the perspective of applications. Specifically, in Section 5.1 we look into the datasets that have been used and the type of heterogeneity solutions have been tested on. This new categorization sheds light on some gaps in the literature and possible future direction for research. And in Section 5.2 we see two other applications in cluster-based federated learning that go beyond personalization.

5.1 Datasets and Tasks

Although the non-iid taxonomy presented in Section 2.2.2 is generally accepted, it has sometimes been in the literature. For instance in [38] "feature distribution skew" includes datasets which can be obtained "by augmenting [features] via rotations" which is considered elsewhere as concept drift (more precisely "concept shift on features") [5]. Similarly, the scenario where labels are swapped between clients for the MNIST datasets (e.g images representing 1 are labeled as 2 for some clients and vice versa) is sometimes defined as concept shift on features (e.g [27, 38]) and elsewhere as concept shift on labels (e.g [5]).

Also, in [43] an analogous taxonomy for the classification of non-iid data was proposed. In that survey, label distribution skew, feature distribution skew, label preference skew (concept shift on labels) and feature condition skew (concept shift on features) and quantity skew correspond to the categories in our taxonomy. However, the authors add "quality skew" for both labels and features which are particular cases of concept

shift on labels and feature respectively. Adding these special cases is not very relevant in our case as we are only interested to look at type of non-iid data without identifying the reason behind it.

The ambiguity in the nomenclature is not entirely surprising as a joint distribution between features and target $\mathcal{P}(X, Y)$ can be factored using Bayes' rule (see equation (3)) as we have seen and hence altering the distribution of labels will have an impact on features and vice versa. That being said, those conflicting definitions make it harder for a reader to compare results.

To alleviate this ambiguity and provide a common ground for comparisons, we summarize the different use cases tested in the literature in Table 4 and specify to which non-iid category they most adequately belong to. To be clear, this categorization is is not definitive in the sense that some use cases can be seen from different angles as we have seen in the section above. Nonetheless, we believe that having a clear formalization that works across studies is helpful for comparisons.

In Table 4, "Dataset" is either a public dataset such as "MNIST", "FEMNIST" or "Shakeaspear", or a private one such as some of the time series data used in [18]. Then, the column "Task Type" defines which category of task is tackled with that dataset (whether it is a classification task, a prediction task, etc.). In Figure 4, a pie-chart representing the distribution of experimental designs from Table 4 is presented. Already an initial pattern comes into sights with a vast majority (88%) of experimental designs focusing on "image classification" tasks on "MNIST" or some variant of it .

In "Heterogeneity Information" we extract from the papers information on how the non-iid setting was induced. Predictably, this information is expressed in various ways across the papers and we therefore suggest in "Suggested Category of non-IID situation" the non-iid category under which we can classify the use case. This suggestion is a recommendation to facilitate the comparison of different studies but is relative to our perception of the problem at hand. From these last two column, we can see that, with the exception of "quantity skew" (only evaluated in [15]) the other type of non-iid categories seem to be adequately represented.

5.2 Other Use Cases

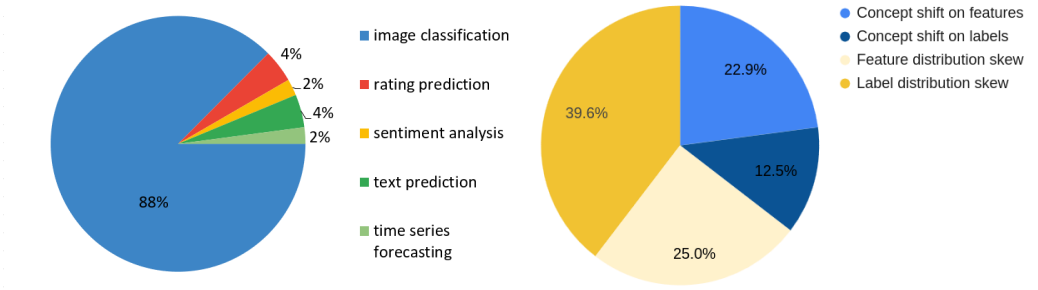
Although the literature on CFL revolves mainly around personalization, there are two other use cases worth mentioning, namely: client selection [11, 21, 37, 41] and attack prevention [1, 13, 19, 45].

With client selection, instead of choosing random clients to participate in each round of training, the CFL framework strategically chooses subset of clients to participate in a round of training based on the properties of the clusters they are in. For instance, clients clustered by computational capacity can be prioritized for fast convergence. Likewise, clients clustered by statistical distribution (as with personalization) can help ensure all data distributions are represented at each round of communication.

The features used for clustering in client-selection studies are rather diverse. For instance, in [41], clusters are calculated using data distribution statistics. Client are then selected across these clusters to ensure an adequate representation of data distributions at each training step. In [21], data labels' variance are used as features for clustering prior to client selection. The aim for this study is to improve global convergence time.

Dataset	task Type	Heterogeneity Information	Suggested Category of non-IID situation	papers
MNIST	Image classification	Label swapping Image rotations Subset of labels per client	Concept shift on labels Concept shift on features Label distribution skew	[3, 5, 10, 36] [5, 7, 14, 15, 30, 41] [3, 9, 15, 22, 42]
EMNIST	Image classification	Image rotations Subset of labels per client	Concept shift on features Label distribution skew	[2, 30, 34] [2]
FEMNIST	Image classification	Split input by author Label swapping Subset of labels per client	Feature distribution skew Concept shift on labels Label distribution skew	[9, 14, 26, 28, 30] [10] [41]
Fashion MNIST	Image classification	Label swapping Subset of labels for each client Image rotations	Concept shift on labels Label distribution skew Concept shift on features	[10, 27] [15, 22] [15]
FedCelebA	Image classification	Split by celebrity	Feature distribution skew	[26]
Shakespear	Next character prediction	Split by play character	Feature distribution skew	[7]
Sentiment140	Sentiment analysis	Split by user	Feature distribution skew	[9]
Time series data	Time series forecasting	Split by class	Feature distribution skew	[18]
ml-100K	Rating prediction	Split by user	Concept shift on labels	[25]
ml-1M	Rating prediction	Split by user	Concept shift on labels	[25]
CIFAR-10	Image classification	Label swapping Image rotations Subset of labels per client	Concept shift on labels Concept shift on features Label distribution skew	[27, 30, 36] [14, 15, 34] [15, 30, 41, 42]
Ag-News	Word prediction	Split by topic	Feature distribution skew	[36]

Table 4. Summary of datasets and tasks tested in the literature



(a) Distribution of use cases studies across papers (b) Distribution of non-iid categories studies across papers

Fig. 4. Distribution of experimental designs across reviewed papers

In [37], communication latency, computational performance, and training time are proposed as features for clustering. At each communication rounds, client are randomly selected from one cluster only. All clusters will be sequentially explored during the FL process in a round-robin way. Finally, in [11], clusters are created based on dataset size or local models similarity. Following this, clients are given a participation probability based on their cluster attributes.

On the other hand, attack prevention is about countering adversarial attacks (poisoning attacks or byzantine attacks [16]) . Preventing these types of attacks is important as they can pollute the global

model's performance or prevent convergence. For instance, an adversarial client might purposefully updates its local model with the opposite of the true gradient, so as to hinder the global model's convergence when aggregated. In that respect, clustering can isolate outliers which are potentially malicious users. As an example Ghosh et al. [13] use a robust version of k-means algorithm combined with outlier-robust optimization to exclude byzantine machine of the FL process. Similarly [19] and [45], use clustering to identify "malicious" nodes whose distributions are too different from the majority. In [1], the authors note that using outlier detection mechanisms can introduce a fairness bias. Particularly clients whose distinct distributions is legitimate might be wrongly excluded too hastily and it is important to choose the criteria for exclusion carefully so as not to lose important information. The authors then propose a solution based on metadata sharing to counter this bias. In [24], client's clusters are found using k-means on metadata such as geo-features (client IP-address), time-features (sending-time) or user-features (client ID). The authors argue that these meta features are often representative of client's similarity. The authors then use a robust aggregation mechanism by cluster to mitigate attacks. Finally, In the FLAME framework [32], the authors use the HDBSCAN algorithm [4], a noise-robust version of DBSCAN to categorize models into three categories: core points, border points and noise points. The models considered as noise will be excluded from the update for the current round.

6 CONCLUSION

This survey examines cluster-based Federated Learning papers with an emphasis on personalization as a use case. It presents a novel classification of solution approaches in Section 3.1 which highlights the similarities and variations over the baseline approach. Then, the review in Section 4 classifies the papers by topics addressed and brings out the relationships between these studies. The classification in the form of a table gives a frame of reference for quickly comparing solution approaches and choosing a framework appropriate to new problems. One thing that stands out form this comparison is the limited literature on some topics like dynamic changes in data distributions and soft-clustering. However, both of these problems are particularly relevant in a real-world scenario where datasets are not static but constantly evolving and often involving a mix of distributions.

Section 5 then goes on examining the papers from the perspective of applications this time. The datasets and task instances which have been tested in the literature are analyzed in Section 5.1, bringing forward general tendencies summarized in Table 4 and Figure 4. Here too we observer a limitation of the current literature with a predominant tendency towards image classification tasks at the expense of other types of tasks/datasets. Additionally, Table 4 serves as an aid to reproducibility by suggesting a common nomenclature to the problems addressed across papers. Finally, Section 5.2 is a review of clustering use-cases in FL beyond personalization which allows future researchers to have a thorough vision of clustering in FL.

REFERENCES

[1] Singh Ashneet Khandpur, Blanco-Justicia Alberto, and Domingo-Ferrer Josep. Fair detection of poisoning attacks in federated learning on non-i.i.d. data. *Data Mining and Knowledge Discovery*, 2023.

[2] Andrea Augello, Giulio Falzone, and Giuseppe Lo Re. Dcfl: Dynamic clustered federated learning under differential privacy settings. In *2023 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops)*, pages 614–619. IEEE, 2023.

[3] Christopher Briggs, Zhong Fan, and Peter Andras. Federated learning with hierarchical clustering of local updates to improve training on non-iid data. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–9. IEEE, 2020.

[4] Ricardo JGB Campello, Davoud Moulavi, and Jörg Sander. Density-based clustering based on hierarchical density estimates. In *Pacific-Asia conference on knowledge discovery and data mining*, pages 160–172. Springer, 2013.

[5] Fabiola Espinoza Castellon, Aurélien Mayoue, Jacques-Henri Sublemontier, and Cédric Gouy-Pailler. Federated learning with incremental clustering for heterogeneous data. In *2022 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2022.

[6] Zhikun Chen, Daofeng Li, Rui Ni, Jinkang Zhu, and Sihai Zhang. Fedseq: A hybrid federated learning framework based on sequential in-cluster training. *IEEE Systems Journal*, 2023.

[7] Don Kurian Dennis, Tian Li, and Virginia Smith. Heterogeneity for the win: One-shot federated clustering. In *International Conference on Machine Learning*, pages 2611–2620. PMLR, 2021.

[8] Run Du, Shuo Xu, Rui Zhang, Lijuan Xu, and Hui Xia. A dynamic adaptive iterative clustered federated learning scheme. *Knowledge-Based Systems*, 276:110741, 2023.

[9] Moming Duan, Duo Liu, Xinyuan Ji, Renping Liu, Liang Liang, Xianzhang Chen, and Yujuan Tan. Fedgroup: Efficient federated learning via decomposed similarity-based clustering. In *2021 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking (ISPA/BDCloud/SocialCom/SustainCom)*, pages 228–237. IEEE, 2021.

[10] Moming Duan, Duo Liu, Xinyuan Ji, Yu Wu, Liang Liang, Xianzhang Chen, Yujuan Tan, and Ao Ren. Flexible clustered federated learning for client-level data distribution shift. *IEEE Transactions on Parallel and Distributed Systems*, 33(11):2661–2674, 2021.

[11] Yann Fraboni, Richard Vidal, Laetitia Kameni, and Marco Lorenzi. Clustered sampling: Low-variance and improved representativity for clients selection in federated learning. In *International Conference on Machine Learning*, pages 3407–3416. PMLR, 2021.

[12] Brendan J Frey and Delbert Dueck. Clustering by passing messages between data points. *science*, 2007.

[13] Avishek Ghosh, Justin Hong, Dong Yin, and Kannan Ramchandran. Robust federated learning in a heterogeneous environment. *arXiv preprint arXiv:1906.06629*, 2019.

[14] Avishek Ghosh, Jichan Chung, Dong Yin, and Kannan Ramchandran. An efficient framework for clustered federated learning. *Advances in Neural Information Processing Systems*, 33:19586–19597, 2020.

[15] Biyao Gong, Tianzhang Xing, Zhidan Liu, Wei Xi, and Xiaojiang Chen. Adaptive client clustering for efficient federated learning over non-iid and imbalanced data. *IEEE Transactions on Big Data*, 2022.

[16] Wen Jie, Zhang Zhixia, Lan Yang, Cui Zhihua, Cai Jianghui, and Zhang Wensheng. A survey on federated learning: challenges and applications. *International Journal of Machine Learning and Cybernetics*, 2023.

[17] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *Foundations and Trends in Machine Learning*, 14(1–2):1–210, 2021.

[18] Yeongwoo Kim, Ezeddin Al Hakim, Johan Haraldson, Henrik Eriksson, José Mairton B da Silva, and Carlo Fischione. Dynamic clustering in federated learning. In *ICC 2021-IEEE International Conference on Communications*, pages 1–6. IEEE, 2021.

[19] Yadav Krishna and Gupta B.B. Clustering based rewarding algorithm to detect adversaries in federated machine learning based iot environment. In *2021 IEEE International Conference on Consumer Electronics (ICCE)*. IEEE, 2021.

[20] Viraj Kulkarni, Milind Kulkarni, and Aniruddha Pant. Survey of personalization techniques for federated learning. In *2020 Fourth World Conference on Smart Trends in Systems, Security and Sustainability (WorldS4)*, pages 794–797. IEEE, 2020.

[21] Hunmin Lee and Daehee Seo. Fedlc: Optimizing federated learning in non-iid data via label-wise clustering. *IEEE Access*, 2023.

[22] Chengxi Li, Gang Li, and Pramod K Varshney. Federated learning with soft clustering. *IEEE Internet of Things Journal*, 9(10): 7773–7782, 2021.

[23] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *Proceedings of Machine learning and systems*, 2:429–450, 2020.

[24] Yanli Li, Dong Yuan, Abubakar Sadiq Sani, and Wei Bao. Enhancing federated learning robustness in adversarial environment through clustering non-iid features. *Computers & Security*, page 103319, 2023.

[25] Tingting Liang, Cheng Yuan, Cheng Lu, Youhuizi Li, Junfeng Yuan, and Yuyu Yin. Efficient one-off clustering for personalized federated learning. *Knowledge-Based Systems*, page 110813, 2023.

[26] Guodong Long, Ming Xie, Tao Shen, Tianyi Zhou, Xianzhi Wang, and Jing Jiang. Multi-center federated learning: clients clustering for better personalization. *World Wide Web*, 26(1):481–500, 2023.

[27] Yibo Luo, Xuefeng Liu, and Jianwei Xiu. Energy-efficient clustering to address data heterogeneity in federated learning. In *ICC 2021-IEEE International Conference on Communications*, pages 1–6. IEEE, 2021.

[28] Yishay Mansour, Mehryar Mohri, Jae Ro, and Ananda Theertha Suresh. Three approaches for personalization with applications to federated learning. *arXiv preprint arXiv:2002.10619*, 2020.

[29] 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*, pages 1273–1282. PMLR, 2017.

[30] Manan Mehta and Chenhui Shao. A greedy agglomerative framework for clustered federated learning. *IEEE Transactions on Industrial Informatics*, 2023.

[31] Sudipto Mukherjee, Himanshu Asnani, Eugene Lin, and Sreeram Kannan. Clustergan: Latent space clustering in generative adversarial networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 4610–4617, 2019.

[32] Thien Duc Nguyen, Phillip Rieger, Roberta De Viti, Huili Chen, Björn B Brandenburg, Hossein Yalame, Helen Möllering, Hossein Fereidooni, Samuel Marchal, Markus Miettinen, et al. {FLAME}: Taming backdoors in federated learning. In *31st USENIX Security Symposium (USENIX Security 22)*, pages 1415–1432, 2022.

[33] Kilian Pfeiffer, Martin Rapp, Ramin Khalili, and Jörg Henkel. Federated learning for computationally-constrained heterogeneous devices: A survey. *ACM Computing Surveys*, 2023.

[34] Yichen Ruan and Carlee Joe-Wong. Fedsoft: Soft clustered federated learning with proximal local updating. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 8124–8131, 2022.

[35] Soham Sarkar and Anil K Ghosh. On perfect clustering of high dimension, low sample size data. *IEEE transactions on pattern analysis and machine intelligence*, 42(9):2257–2272, 2019.

[36] 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*, 32(8):3710–3722, 2020.

[37] Ammar Tahir, Yongzhou Chen, and Prashanti Nilayam. Fedss: federated learning with smart selection of clients. *arXiv preprint arXiv:2207.04569*, 2022.

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

[39] Ye Lin Tun, Minh NH Nguyen, Chu Myaet Thwal, Jinwoo Choi, and Choong Seon Hong. Contrastive encoder pre-training–based clustered federated learning for heterogeneous data. *Neural Networks*, 2023.

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

[41] Joel Wolfrath, Nikhil Sreekumar, Dhruv Kumar, Yuanli Wang, and Abhishek Chandra. Haccs: heterogeneity-aware clustered client selection for accelerated federated learning. In *2022 IEEE International Parallel and Distributed Processing Symposium*

(IPDPS), pages 985–995. IEEE, 2022.

[42] Yao Xiao, Jiangang Shu, Xiaohua Jia, and Hejiao Huang. Clustered federated multi-task learning with non-iid data. In *2021 IEEE 27th International Conference on Parallel and Distributed Systems (ICPADS)*, pages 50–57. IEEE, 2021.

[43] Mang Ye, Xiuwen Fang, Bo Du, Pong C Yuen, and Dacheng Tao. Heterogeneous federated learning: State-of-the-art and research challenges. *ACM Computing Surveys*, 56(3):1–44, 2023.

[44] Yue Zhao, Meng Li, Liangzhen Lai, Naveen Suda, Damon Civin, and Vikas Chandra. Federated learning with non-iid data. *arXiv preprint arXiv:1806.00582*, 2018.

[45] Li Zhuohang, Liu Luyang, Zhang Jiaxin, and Liu Jian. Byzantine-robust federated learning through spatial-temporal analysis of local model updates. In *27th International Conference on Parallel and Distributed Systems (ICPADS)*, pages 372–379. IEEE, 2021.