Data Independent Warmup Scheme for Non-IID Federated Learning

Mohamad Arafeh[a], Hakima Ould-Slimane[b], Hadi Otrok[c], Azzam Mourad[d,e], Chamseddine Talhi[a], Ernesto Damiani[c]

[a]*Department of Software Engineering, Ecole de Technologie Superieure (ETS), Montreal, QC, Canada*
[b]*Department of mathematics and computer science, Universite de Quebec a Trois-Rivieres (UQTR), Canada*
[c]*Center of Cyber-Physical Systems (C2PS), Department of EECS, Khalifa University, Abu Dhabi, UAE*
[d]*Cyber Security Systems and Applied AI Research Center, Department of CSM, Lebanese American University, Lebanon*
[e]*Division of Science, New York University, Abu Dhabi, United Arab Emirates*

Abstract

Machine Learning's influence increases daily, and its application prevails in different, critical, and life-changing fields. However, collecting the needed data is challenging due to the increasing concerns about clients' privacy. In this context, federated learning addresses privacy issues by adopting an on-device model training strategy while communicating only the model parameters rather than raw data. Such an approach allows for preserving users' information and shielding them from harm following malicious or suspicious parties' dissemination of their private information. Nevertheless, the distributed nature of federated learning makes it vulnerable to the weight divergence problem caused by the Non-IID (Non-Independent And Identically Distributed) clients. As a result, we can observe a substantial accuracy reduction and convergence time increase. In this paper, we propose a scheme for federated learning addressing the Non-IID clients while also handling the dynamic evolution of the learning context without sacrificing the clients' privacy. Our scheme takes advantage of the client's weights to select a compatible subset with the minimum weight divergence to aggregate the initial global model. Our experiments show that the proposed method can boost the federated learning performance while reducing the con-

vergence time.

Keywords: Federated Learning, Non-IID, Genetic Algorithm, Machine Learning, Distributed Learning

1. Introduction

The current era witnesses an outstanding development in the internet domain and its infrastructure. The introduction of the 5g and 6g networks led to a more practical installation of IoT (internet of things) devices across the world [1]. Therefore, the data generated from these devices has been increasing daily. Internet services such as online social networks, e-commerce, streaming services, and others play a significant role in data production. According to [2] blog, there are 2.5 quintillion bytes of data created each day, and 90% of the world's data was generated in the last two years only. Several scientific fields can benefit from the fast-paced advancement of data generation. Numerous Machine Learning (ML) technologies, such as deep learning, have been introduced in the last decade to leverage the immense amount of data to increase the efficiency and accuracy of artificial intelligence [3, 4, 5]. Furthermore, the development of artificial intelligence led to many quality of life improvements (e.g. smart cities [6, 7, 8], healthcare [9], and finance [10]).

Federated learning (FL) has been a field of interest during the last few years as a practical solution that addresses users' privacy and replaces the traditional centralized servers [11]. FL gains much influence with its premise of utilizing owners' data to locally train ML models in a distributed manner without the need to transmit raw data to external parties (e.g., server, edge). Such an approach revolutionizes ML in privacy-sensitive firms like mobile crowdsensing, drugs, medical history, and vehicular networks [12], where users share sensitive information such as their locations, written text, or medical information. Figure 1 presents a traditional FL architecture. A device can be a vehicle [13], mobile [14], smartwatch, or IoT devices [15, 16, 17]. Each device trains and uploads its locally trained model to a server that derives a global model. However, FL

distributed nature limits the server control over the training data, leading to the Non-IID issue. Non-IID occurs when trainers conduct the data collection. Hence, it is possible that the collected data is mostly biased and belongs to a different distribution from other trainers. In the FL context, aggregating such models is subject to weight divergence, which considerably increases the total training time compared to the centralized training [18].

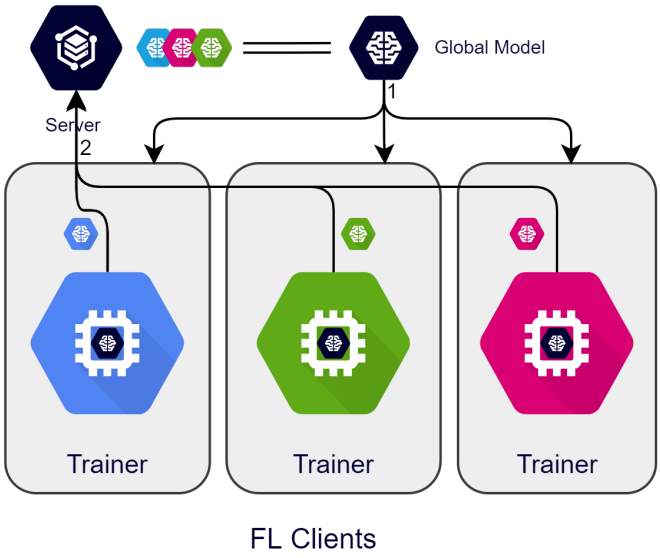

Figure 1: Collaborative and distributed model training using federated learning.

To address the aforementioned problem, authors in [18] proposed an approach in which clients share part of their raw data with the server to train an initial model. In addition, the server allows future clients to use the raw data while training their local models. Their approaches have proven to reduce the issue of Non-IID. However, it relies on clients sharing part of their sensitive data with the server, a clear breach of their privacy. Another solution proposed by [19] includes clustering clients and running a separate FL for each cluster. This approach handles the Non-IID issue by having all the similar clients in the same cluster. However, it modifies the FL structure, which limits its applicability. In addition, the architecture suffers from scalability problems where clusters might

not have sufficient clients, or the cluster number might be subject to continuous dynamic changes. An alternative solution addressing Non-IID clients is discussed in [20]. Instead of working with client selection, the authors operate on the model aggregation algorithm. They used a novel aggregation technique to detect clients' model distribution to fix their weight divergence. However, their approach requires an estimate of an indeterminable parameter. To the best of our knowledge, none of the current approaches in the literature has targeted avoiding the need for clients to share part of their sensitive data with the server while addressing the dynamic constraints posed by the continuous evolution of FL environments.

In this paper, we address the aforementioned challenges by proposing a novel privacy-preserving scheme that lessens the negative impact of weight divergence through an initial trained model to soothe the data distribution problem. A set of clients is selected to aggregate a global model that will act as an initial model for the rest of the FL process. However, creating such a model requires aggregators to have the minimum weight divergence. Our scheme introduces a preliminary round incorporating the maximum number of available clients. These clients send their locally trained model to the server, which will operate a compatibility selection and choose a subset to build a global model that can reduce the impact of diverged weight in future rounds. The preliminary round clients are selected based on their model's weights. The server scores each selection by calculating the weight divergence between the aggregated model and the selected client's model. The optimal solution is a model with the minimum weight divergence, which will act as an initial model for future rounds. Due to the number of possible combinations, finding the optimal solution is challenging due to the limited time and available resources. We propose a genetic-based client selection to reach a sub-optimal solution within limited time and resource constraints. The main contributions of the paper are summarized as follows:

1. Formalize the challenges of the Non-IID in FL contexts as a weight divergence problem and provide a details analysis on the evolution of the

clients' weights.

2. Solve the problem of weight divergence by adapting a selection-based approach. We seek a subset capable of building an initial model with the minimum diverged weights and apply genetic algorithm to reduce the time required to reach a fitting solution.

3. Present the capabilities of our approach by conducting extensive experiments under various Non-IID constraints and FL configurations. In these experiments, we demonstrate our approach capabilities compared to the standard FL workflow and Warmup approach [18, 21] as a baseline. Additionally, we compare against various clustering algorithms, including K-Means and K-Center, under various Non-IID levels and data distributions.

2. Non-IID Problem Illustration

We provide the needed background about the communication protocol of Federated Learning illustrated in Figure 2.

Step 1: The server starts by initiating a model and decides the configuration and the hyper-parameters.

Step 2: The server selects clients to train a model in each round. The selection procedure is an essential part of the framework to minimize the time needed to complete one round and reduce the bandwidth and computation resource cost. The default client selection scheme includes a random client selection [22]. However, it is possible to select specific clients, based on predefined criteria, that would contribute more than others [23], or finish faster, reducing the overall waiting time.

Step 3: Selected clients download the latest model from the server.

Step 4: Selected clients train, in parallel, their model locally using collected or available data.

Step 5: Clients upload their model to the server.

Step 6: Server aggregates the received model and creates the new global

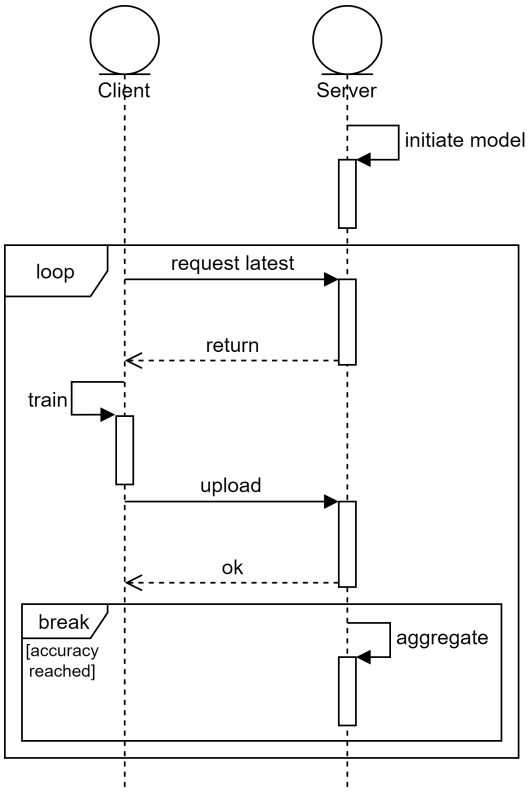

Figure 2: Federated Learning Communication Protocol.

model. For this task, Federated Average (FedAVG), as in Equation 1, is used for model aggregations as follows:

$$w_{t+1} = \sum_{k=1}^{K} \frac{n_k}{n} w_{t+1}^k \tag{1}$$

where w refers to the aggregated weights on the server, w^k refers to the received weights of client k, and n is the number of samples used to train the model.

Step 7: Server stops if it achieves any of the stopping criteria or starts from step 2 otherwise.

According to [22], FedAVG suffers from weights divergence, which takes place when the client's data are non-identically distributed (Non-IID). Non-IID occurs when at least one of the following data distributions exists:

- Non-Independent: Refers to the inconsistencies in the data generations between clients. It happens when the same labels have different features' values. For example, clients take pictures of the same image from varying angles.

- Non-Identically Distributed: Related to the deviation in data distribution between the participating clients.

In federated learning, each user uses his locally generated data to train models. Thus, it is highly susceptible to being Non-Independent since each user behaves differently from other clients while generating his own skewed features. Furthermore, each user has a different environment from another, which signifies a generation of Non-Identically Distributed data. Consequently, each client produces a biased dataset with either different or missing features and labels. The aggregated models of such clients reduce the overall global model accuracy and incur a high loss due to the considerable weight divergence between the clients' models. Without employing any solution, it is possible that the situation could resolve itself over time. However, it might take several communication rounds, hindering FL's benefits and creating drawbacks, especially when working with devices with limited capabilities. According to [22], the number of rounds it took to reach the desired accuracy increased by almost 1000% when working with Non-IID data. Furthermore, the global model efficiency is affected by a diversity of Non-IID severeness, which can be separated into three levels: Mild Non-IID, Highly Non-IID, and Unique. For Mild Non-IID, all clients have access to enough labels (at least 25% of the labels) and a fair amount of data compared to other users. On the other hand, Highly Non-IID consists of clients with local data of a single label/class. Finally, Unique Non-IID extends the Highly case when considering clients with only one label. However, in this case, labels are unique to each client. For example, with a ten labels dataset, we can create a maximum of 10 clients, which is different from the Highly case, where we can create 100 clients when duplicate label clients are allowed. All three cases impact the global model efficiency due to weight divergence issues incurred by

the federated learning aggregation.

To clarify the impact of weight divergence when working with Non-IID clients, consider two cases in which 100 clients want to participate in a federated learning task. In the first case, each client holds only one label (i.e. Highly Non-IID), while in the second case, the global data distribution between clients is IID. The used model is simplified, containing one hidden layer to ease the analysis. Additionally, only the linear weight is considered while excluding the bias. The model's final shape is a matrix H×W where H is the number of labels (i.e. output size), and W is the number of features. When the server receives the clients' model, it flattens the weights and presents them in a graph as depicted in Figure 3. Each plot line represents the model's weight of a different client in a certain round.

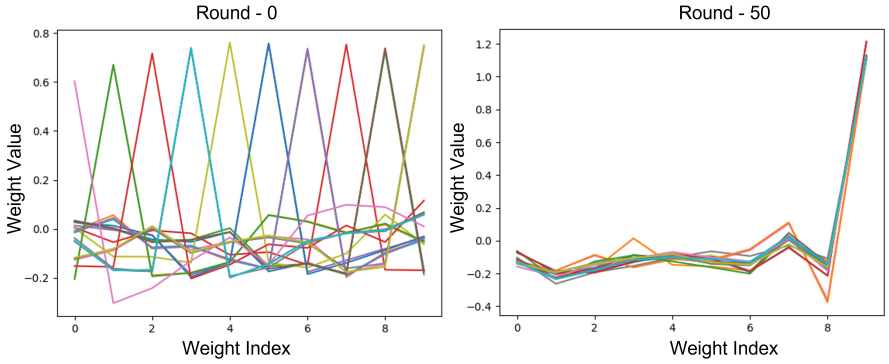

Figure 3: The development of 100 clients' weights in a Non-IID environment in rounds 0 and 50. Each plot line represents the compressed linear weights of a single client. The X axis is the flattened weight value, while the Y axis is the flattened weight index

In normal circumstances, it is hard to clearly represent the differences when showing the entirety of the model's weights. Thus, the server applies principal component analysis (PCA), similar to [24], to compress the weights and extract only the changing components. As illustrated in Figure 3, the clients are forming clusters where each produces a model entirely different from the other. Aggregating such models using the standard FedAVG would solve the issue, but

the process would take more time as the model is trying to solve the weight divergence issue first. This example took the server around 50 rounds to have non-diverged weights between clients. The transition to a non-diverged weight context is fundamental because the model accuracy would not improve without this step. When the model transmission to a non-diverged weight is complete, the global model would not be affected further, even when the clients persist in providing Non-IID data.

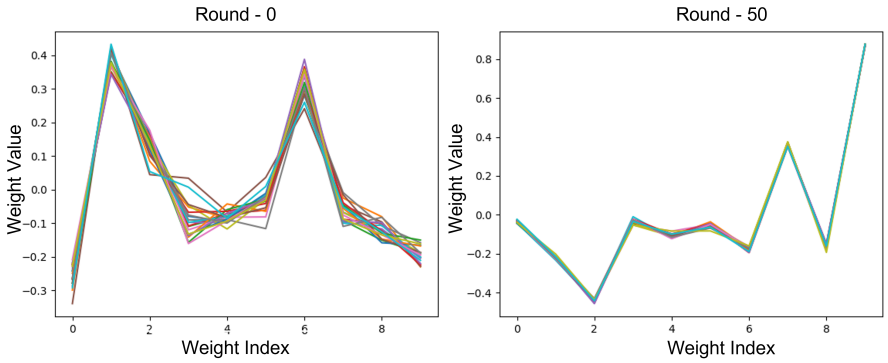

Figure 4: The development of 100 clients' weights in an IID environment in rounds 0 and 20. Each plot line represents the compressed linear weights of a single client. The X axis is the flattened weight value, while the Y axis is the flattened weight index.

Different from a Non-IID context, the same experiments are executed with the same configuration but with IID data distribution. The weights are available in Figure 4. Unlike a Non-IID context, the weights were similar, and the accuracy improved from the start.

The problem is mathematically explained in [18], in which the authors proposed a solution to weight divergence. Their proposal included clients sharing part of their data with the server to build a warmup model that acts as an initial model for further training rounds. Figure 5 presents the clients' model weight when the server receives them after the first rounds. Clients start from a warmup model and bypass the transmission phase entirely to non-diverged weights, which might take a decent number of rounds, especially with more

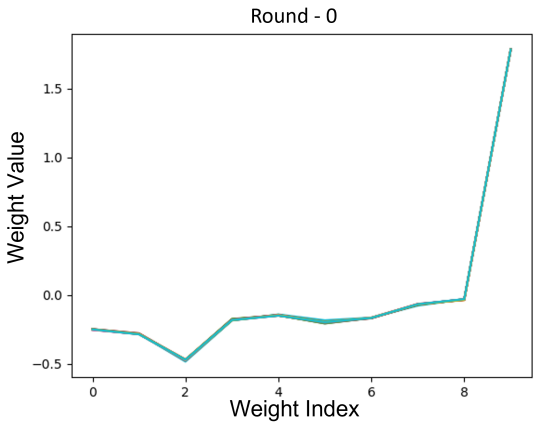

Figure 5: The positive impact of warmup initialization on the initial clients' weights, in a Non-IID environment, after the first round. X is the flattened weights index, and Y is the value of the flattened weights

complicated models. However, their proposal required the clients to share part of their data with the server to train the initial model, which contradicts the primary purpose of federated learning in protecting client privacy by sharing only models.

Overall, we can conclude that Non-IID clients produce models with diverged weights. The aggregation of such models generates a model with lower quality. Therefore, the warmup model approach's capabilities and positive effect in reducing the starting weight divergence allow the model to improve faster and reach convergence in fewer rounds. However, finding a replacement for clients sharing part of their private data with the server is essential.

3. Literature Review

The proposed solutions addressing the Non-IID issues are available in Table 3 and fall into two main categories:

1. **Aggregation:** The main issue of Non-IID is affected mainly by the aggregation method. It is possible to employ particular aggregation algorithms

to fix the problem or bypass the model's weight divergence when trained by clients with Non-IID data.

2. **Client selection:** These approaches select clients by either their performance or based on their model's influence on the global model. The latter tends to use the received model weights, while the first tends to use the accuracy of the models.

Non-IID Solutions			
Aggregation	**Client selection**		
[25]	**Model-Based**	**Client-Based**	
[20]	[19]	**Single**	**Group**
[26]	[27]	[28]	[23]
[18]	[29]	[30]	[29]
	[31]	[32]	[33]
			[24]

3.1. Aggregation Approaches

In the following, we present the current approaches related to the aggregation techniques addressing the weights divergence issues. In [25], the authors present the weight divergence issue as a problem of weight drifting. They noticed that a biased client trained a skewed model with weights that drifted to his data distribution. Thus, they proposed solving the local weight drift by introducing a control variable with the same shape as the global model. Each client uses the difference between its control variable and the one received from the server (i.e. the average of all control variables) to correct the local drifting of the model weights in each round. Federated Learning Framework Robust to Affine distribution shifts (FLRA) [20] is another solution that tries to solve the model of shifting model weights. The authors assumed that each client data follows an affine distribution shift representing the model weights of each client shift from the ground distribution. The authors formulated their objective as a minimax optimization problem that tries to find the global model by minimizing the local affine transformation loss. The main limitation of this approach is their assumption about the feasibility of calculating the clients' affinity. Federated

Learning with Matched Averaging [26] (FedMA) is proposed as a replacement for FedAVG. Instead of averaging the model weights, the authors tried to find the best permutation of model weights in each layer. The authors in [18] proposed an approach that depends on starting federated learning from a warmed-up (i.e. pre-trained) model. Having the server model gradient trained on a portion of the data reduces the local weight divergence between clients. However, this solution requires the server to have part of its clients' raw data to create the warmup model before starting the federated learning.

3.2. Client Selection Approaches Based on Model Performance

In this section, we present all the related work that uses the model weights to select the most suitable clients in each round. The authors in [19] proposed a hierarchical clustering scheme for federated learning. By measuring the distance between model weights, it is possible to cluster federated clients into multiple groups based on their similarities, such as training models using equivalent labels. The experiments compare the accuracy of the group clustering based on different distance-measuring algorithms. Once the clients are clustered into multiple groups, the authors carry out a federated learning task separately, resulting in numerous global models for the same tasks. Such behaviour is similar to what Google is trying to achieve with their new strategy called "FLOC" (federated learning of cohorts), in which they tried to cluster the client based on private data (e.g. model) to avoid advertisers accessing users' personal information like cookies. FedGroup [27] followed a similar approach by clustering clients and running a separate task for each group. Furthermore, they discussed the cold start where new clients' cluster is unknown. Additionally, the authors employed an improved version of Cosine Distance called Decomposed Cosine Similarity in combination with K-Mean instead of a hierarchical algorithm for clustering. The proposed solution can immediately handle new trainers, which was considered challenging in other approaches. The iterative clustering [29] proposed an approach that uses gradient descent to find similarities. It starts with the server broadcast models. Later, clients identify their cluster memberships and run lo-

cal updates. Then clients send back the local models to the server. Finally, the models aggregated within the same estimated cluster. Similarities are calculated by finding the model parameter with the lowest loss. This approach's premise is to boost the starting accuracy, which results in attaining the desired accuracy faster, leading to a reduction in communication costs [34]. Their experiments showed that this approach reaches the desired accuracy between 5% to 8% faster using non-convex ML models. Since the model cluster identity is calculated from its gradient, this approach differs from the previous one. As a result, it is possible for the clustering to occur while training rather than doing a preliminary procedure, reducing the time needed to boot the federated learning. However, this approach iteratively calculates each model's distance to every other model, which requires a significant amount of resources that increases exponentially with the number of trainers. Unlike the above approaches, the authors in [31] proposed a framework for client selection to reduce the communication rounds by preventing relevant updates. The relevance can be defined by the number of local model parameters holding the same sign as the estimated global model. The sign, in this case, determines the direction of the update. Thus models with a higher number of signs to the global model are going in the same direction and have high relevance. A data-independent scoring mechanism proposed by authors in [3] includes a serverless federated learning scheme controlled by cluster heads. Cluster heads motivate new clients to join the cluster by having a higher reputation score calculated using their clients' model weights. However, the authors disregarded the Non-IIDness of the clients, which results in two issues: first, a client with Non-IID data might be considered an adversary. Second, if most clients are highly Non-IID, the clustering will converge to maintain similar clients, worsening the final results when cluster heads aggregate their models.

Even though a good performance using clustering as in [27] and [19] is perceived, these solutions yet suffer from multiple issues. They are highly affected by the cluster-size hyper-parameter, which defines the number of available FL hosts for clients of similar models. To correctly set the cluster-size value, the authors required former knowledge of the data distribution. In most cases,

such knowledge is hard to get or unknown priory in dynamic and unpredictable environments. Additionally, the mentioned approaches include a considerable change in the structure of the FL framework by having multiple models as an output. Furthermore, framework [19] required the participation of all clients in the first few rounds to identify their clusters. In normal circumstances, such a requirement is impracticable due to bandwidth limitations, unavailable clients, or the server's incapability to handle a massive amount of clients in parallel.

3.3. Client Selection Approaches Based on Client Performance

This section presents the related works that try to infer the client's performance in federated learning to address the Non-IID problem. The definition of client performance differs from one work to another. Some approaches define the client's performance by the size of the collected data used to build the local model and the time a client needs to finish the training. It can be described by how much a client is willing to sacrifice to complete the task. Alternatively, other works took another approach by considering the client's performance in the group rather than a single entity. These methods measure the client's performance while simultaneously considering the global model's performance. We can categorize the work in this section as single entity performance and group performance. In the following, we present different works that fall under either category.

3.3.1. Single Entity Approaches

Single Entity approaches defined the client performance entirely based on the client's self properties and gave importance to one of them. For example, in [28], the authors extended the federated learning scheme for mobile edge computing (MEC) devices and proposed a unique client selection algorithm that focuses on clients' completion time. The main property used for client selection is how fast a client can train and upload its model to the server. Another form of single entity approach focused on the size of the data shared by each client and the device capabilities for its client selections. Clients with more data and

superior computational powers have a higher selection probability. In [30], the authors proposed a contract theory scheme to incentivize clients to participate in a federated learning task. Clients are rewarded based on how much data they contribute to their local model while considering the device's capability (CPU). A similar approach proposed in [32] employs a tier-based selection system in which clients are grouped into multiple tiers based on their processing speed and how fast they respond. The main limitation of these approaches is the focus on quantity rather than quality. The clients are selected based on the committed data size and processing speed, while there is little or no mention of the data distribution. That made them intolerant of clients with limited capabilities but possessed valuable information.

3.3.2. Group Approaches

During the participants' selection, the authors proposed in [23] an algorithm measuring the participants' contribution to the global model. They considered both vertical and horizontal federated learning contexts in their work and provided a distinct strategy for each. In horizontal federated learning, all the trainers train their data on the same feature. The problem, in this case, is how much a trainer model is beneficial to the global model accuracy. The authors took advantage of the trainer model's availability on the server to measure its effectiveness. For this purpose, they suggested a deletion algorithm that compares the prediction results' probabilities between the global model and the global models without the client model. As such, a model's importance is measured by its impact on the accuracy of the global model. Additionally, They proposed a coalition game theory algorithm for vertical learning that depends on shapely values to measure each feature's importance. In vertical federated learning, the data of the same client exists on two or more different parties. Therefore, each party may have its own set of features that differ from others. For example, client data exists on a retail store and a bank. The authors used Shapley values to calculate each feature's importance in solving these issues. It is possible to take advantage of this information to reduce the number of un-

necessary data. However, it's essential for the server to have access to test data to calculate each of the model contributions. This requirement is not always possible as there are cases where test data does not exist. Additionally, most cases might require continuously updated or live data for these approaches to work. Furthermore, the authors emphasized measuring the model's effectiveness in distributing incentives over solving the Non-IID issues. On the other hand, the work in [29] provided a detailed analysis asserting that shapely value is unsuitable because its values are significantly dependent on the used machine learning model. Hence, in some cases, its values might be unfair leading trainers to quit.

A self-balancing framework by training a server mediator on the server was proposed in [33] to solve the global class imbalance of client distribution. A new entity called server mediator adapts to the participants' distribution and requests from clients to update their data by up-scaling or down-sampling based on their distribution. However, the authors acknowledged the access of clients' data distribution which might cause inference on their private information and breach their privacy. A reinforcement learning-based client selection was proposed in [24]. The authors extended the server with a deep agent capable of predicting the performing clients for the next round to yield the best accuracy. The agent selects clients based on the evolution of their weights and their effects on the global model accuracy, which is the reward after each selection. The issue with the mentioned works is the need for a validation set on the server to infer the client's accuracy. In [33], the authors proposed architecture with additional nodes that handle the accumulation of data to validate the models. Consequently, applying a federated learning solution will be more challenging due to the high deployment cost and questionable scalability.

4. Proposed Scheme

4.1. Architecture

In this subsection, we provide a detailed overview of our scheme. In an additional preliminary round, clients train a local model and send it to the server. The size of the selected clients depends on the server's capabilities and the client's availability. The server uses the received models to select a subset and aggregates a warmup model. The rationality behind the selection process is to avoid the creation of a biased warmup model for a specific part of the data, which may occur due to either data un-balancement or class un-balancement. Authors in [22] explained in detail the aforementioned problems.

The warmup model's primary role is bypassing the diverged weights transition issue when the client's data follow a Non-IID distribution. Furthermore, a compatibility-based model selection process is required to measure the effectiveness of the resulting global model after each selection. A group of models is aggregated and considered a solution when it learns from every part of the data and avoids bias toward a particular part of the data.

In the following, we put forward the architecture and workflow of the proposed scheme illustrated in Figure 6, and Algorithm 1.

- **Step 1:** Process starts by issuing a preliminary selection round consisting of the maximum number of available clients to train a model from their local data
- **Step 2-3:** Clients send back the model through the communicator component, which supervises all the communication between the server and the participants.
- **Step 4:** During the preliminary round, the communicator sends the models to the preliminary selector component that searches for a subset of compatible clients to aggregate the warmup model. The server starts from the warmup model in the subsequent rounds.

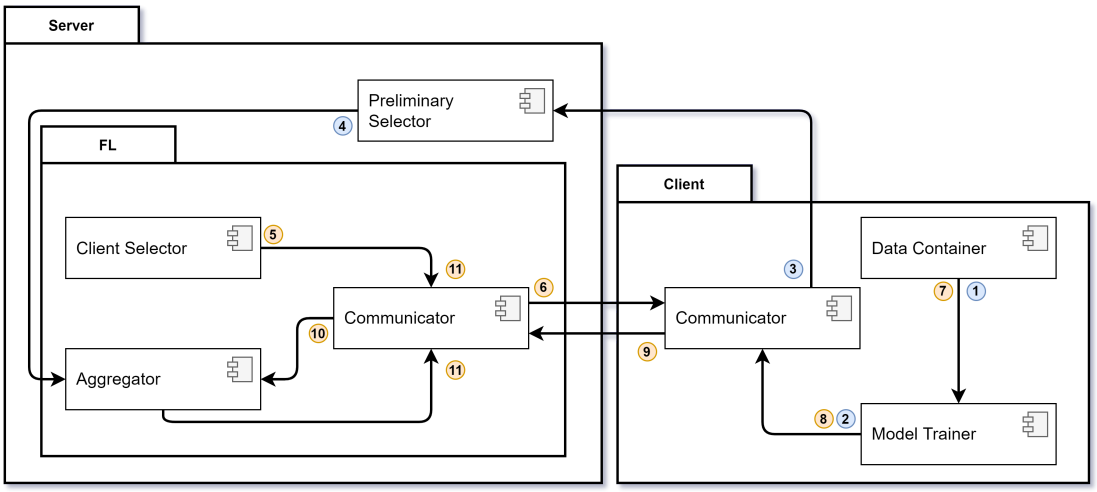

Figure 6: Federated Learning with preliminary initialization strategy. An overview of the framework components and their relationships.

- **Step 5-6:** Next process starts with the communicator receiving the next batch of clients chosen by the client selector and sending them the initial model.
- **Step 7-8-9:** Selected clients receive and train the model from their locally available or collected data and send it back to the server.
- **Step 10-11:** The aggregator receives the models from the communicator, creates the new global model and assigns it to the communicator preparing for the next round.
- **Repeat** steps 5-10 until the server achieves the stopping criterion.

4.2. Preliminary Selector

This section presents the formulation behind our Preliminary Selector. Let $C = \{c_1,\ c_2,\ ...,\ c_K\}$ be a set of K clients. We aim to select a subset of clients $S\ \subset C$, where $|S| < K$, is the number of clients representing the compatible clients from C. Hence, these selected clients should satisfy the following maximization objective:

Algorithm 1: Proposed Scheme

```
/* step 1                                                    */
C ← available clients
all_parameters ← ∅
for c in range(start=0, stop=|C|, step=1) do
    /* steps 2-3                                             */
    /* In parallel, on client side                           */
    all_parameters ← train(c)
/* step 4                                                    */
preliminary_clients = preliminary_selector(all_parameters)
initial_model = aggregate(preliminary_clients)
/* steps 5-10                                                */
federated_learning(initial_model, ...configs)
```

$$max\ \varphi_t(S) \tag{2}$$

where φ is a function that measures the compatibility between clients S at round t described as the following:

$$\varphi_t(S) = \frac{|S|}{\sum_{i \in S}(\|\triangle w_t^i\|_2 - \|\overline{\triangle w_t^S}\|_2)^2} \tag{3}$$

where $\|\triangle w_t^i\|_2$ is the l_2-norm between the local model update of client i, w_t^i, and the global model w_{t-1}. This distance is calculated as:

$$\|\triangle w_t^i\|_2 = \sqrt{\sum_{j=1}^{n}(w_{t-1,j} - w_{t,j}^i)^2} \tag{4}$$

and the average l_2-norm of local model updates generated by the selected clients is calculated by the following:

$$\|\overline{\triangle w_t^S}\|_2 = \frac{1}{|S|}\sum_{i \in S}\|\triangle w_t^i\|_2 \tag{5}$$

The subset of clients satisfying Equation 3 have the minimum weight divergence. Such models have similar weights, including their outputs. As such, minimizing weight divergence yields a subset of nearly identical clients that fall under the same data distribution, aggregating an overfitted global model with limited outputs. To avoid this issue, the selection process relies on clustering techniques as a solution. From the model's weights, clustering algorithms can distinguish the data distribution of each client [19, 24] to identify its cluster group. Thus, having similar clients in the same cluster allows the selection algorithm to even out the data distribution and avoid building subsets of clients that target only a specific part of the data. The selection algorithm applies K-Means [35] clustering on the clients' weights to cluster them into predefined number groups. Thus, the updated φ can be described as:

$$\varphi_t(S) = \frac{|S|}{\sum_{Cl_j \in CL} \sum_{i \in Cl'_j} (\|\triangle w_t^i\|_2 - \|\overline{\triangle w_t}\|_2)^2} \tag{6}$$

where $CL = \{Cl_1, Cl_2, ..., Cl_r\} \subset 2^C$ is a set of clusters generated by the K-means algorithm and $r = |CL|$ is the number of obtained clusters.

These clusters satisfy the following conditions:

- $\forall 1 \leq i \leq r, Cl_i \neq \emptyset$
- $\forall 1 \leq i, j \leq r, i \neq j, Cl_i \cap Cl_j = \emptyset$
- $\forall 1 \leq i \leq r, \bigcup_{i=1}^{r} Cl_i = C$

From each cluster Cl_j, select a subset $Cl_j^{'}$, $|Cl_j^{'}| = q$, where q is the number of selected clients from each cluster. Hence, the number of selected clients is $|S| = r \times q$.

Consequently, our new objective can be formulated as follows:

$$\max \ \varphi_t(S) \text{ subject to } q \leq \min\{|Cl_1|, |Cl_2|, ..., |Cl_r|\} \tag{7}$$

Following the above objectives, client selection involves seeking a compatible subset of clients while navigating over contradictory objectives with properties

of combinatorial decision-making. This process is essentially NP-hard due to the exponential number of possible solutions, similar to the combinatorial explosion seen in the Knapsack problem.

$$P = \binom{|Cl_1|}{q} \times \binom{|Cl_2|}{q} \times ... \times \binom{|Cl_r|}{q} \tag{8}$$

where r is the number of clusters and q is the number of selected clients from each cluster.

Finding the subset S requires the framework to iterate over every possible combination, which will take a significant amount of time, and it is highly affected by r and q and the complexity of the ML model. Due to the excess of time needed to have a feasible solution and the time limitation imposed on the server, we resolve to a meta-heuristic approach that seeks to find a sub-optimal solution instead, but in a much shorter time, as elaborated in the next section.

Additionally, balancing objectives such as maximizing diversity and minimizing divergence introduces complexities comparable to multi-objective optimization, which is notably difficult to solve due to conflicting goals. Moreover, the constraints demanding non-repetitive selection and representation from each cluster further align this problem with NP-hard challenges like the Set Cover and Constraint Satisfaction Problems, which require satisfying specific criteria rather than merely seeking an optimal solution.

Searching for a set of compatible clients from the clusters is computationally expensive and takes time to solve. We consider any client's combination as a solution. Hence, we can model the problem as a combinatorial problem where we are seeking a solution from a finite set of possible solutions that have the following complexity:

$$P = \binom{|Cl_1|}{q} \times \binom{|Cl_2|}{q} \times ... \times \binom{|Cl_r|}{q} \tag{9}$$

where r is the number of clusters and q is the number of selected clients from each cluster.

Finding the subset S requires the framework to iterate over every possible

combination, which will take a significant amount of time, and it is highly affected by r and q and the complexity of the ML model. Due to the excess of time needed to have a feasible solution and the time limitation imposed on the server, we resolve to a meta-heuristic approach that seeks to find a sub-optimal solution instead, but in a much shorter time, as elaborated in the next section.

5. A Genetic Based Client Selector

To address the complexity of the preliminary selector, we resolve to a Genetic Algorithm (GA) for the ability to control and reduce the amount of time it needs to have a solution while exploring most of the highest quality solutions [36, 37]. We elaborate on the following three key components: initial population, fitness evaluation, and evolution.

5.1. Initial Population

We start by building our initial population, a collection of chromosomes created using an array of genes. A gene definition is distinctive to each problem. In our case, a gene holds a reference to a client model within a chromosome that contains a subset t of clients n. We try to validate the effectiveness of their combined models. Algorithm 2 describes the steps we follow while building our initial population. Usually, a chromosome is built by selecting genes randomly, allowing duplicate genes to exist in the same chromosome. However, under the circumstances strained by federated learning and following the same strategy as in Section 4.2, each chromosome should follow the following rules:

- Each model exists only once in the aggregated model. Thus, no chromosome with duplicate genes is allowed
- To avoid the selector residing at a minimum; the next gene should target part of the data new to the current chromosome (i.e. gene from a different cluster).

Algorithm 2: Initial Population

input : $clients\_model, population\_size,$
$chromosome\_size, cluster\_size$

$clustering(clients\_model, cluster\_size)$

$population \leftarrow \emptyset$

while $|population| < population\_size$ **do**
  $chromosome \leftarrow \emptyset$
  while $|population| < chromosome\_size$ **do**
    $M'_c \leftarrow$ set of random models following (5)
    $r = random(0, |M'_c|)$
    $chromosome \leftarrow chromosome + M'_c[r]$
  $population \leftarrow population + chromosome$
return $population$

To satisfy the aforementioned rule, we employ the K-Means algorithm to identify the clients' clusters and avoid overfitting the chromosomes with similar ones. Following Section 4.2, the cluster size p is a predefined parameter that can be configured based on the application's needs. We suppose each chromosome contains the same number of clients q selected from each cluster. Thus, the chromosome size can be presented as $|S| = r \cdot q$.

5.2. Fitness Evaluation

The selection algorithm evaluates the chromosomes by applying the utility function mentioned in Section 4.2. Each chromosome aggregates a global model and uses it to measure the weight divergence of each client model. Then score the chromosome by computing the inverse variance of the measurement array, which indicates the combined compatibility of the client's model. At the start of each iteration, the algorithm scores the current population chromosomes and updates the best solution.

5.3. Evolution

Algorithm 3 describes the evolution of the proposed GA solution. We follow the traditional procedures: evaluation, selection, crossover, and mutation.

Equation 3 is used to measure the collective fitness of each chromosome and update to a better solution if it exists. Later, filter the population during the selection by taking its best half, succeeded by the crossover and mutation to produce the new generation.

A crossover happens between two chromosomes when a randomly generated float number between 0-1 is greater than a predefined crossover ratio provided by the GA configuration. When a crossover occurs, both chromosomes are cut in a random position and switch their places. Next, mutation follows a similar structure to the crossover by defining a mutation ratio for each gene inside a chromosome. If a mutation occurs, replace the current gene with a random client following the same rules as in Section 5.1.

The result of the genetic selection is a chromosome, a collection of clients whose combined model has the minimum weight divergence. At the same time, its knowledge contains all the possible outputs. This model is used later as the initial model.

6. Experimental Results

In this section, we present the results of our experiments, in which we run our tests on two image classification datasets MNIST [38], and CIFAR-10 [39]. MNIST is a handwritten simple data containing 60,000 28*28 pixels images divided into ten digits from 0 to 9. CIFAR-10 contains 60,000 3*32*32 colored images divided into ten classes. For the MNIST classification problem, we used logistic regression as a ML model while designing a convolution neural network [40] for CIFAR-10 with configuration available in Table 1. Cross-Entropy Loss is used as a criterion, considering that the experiments fall under image classification problems.

For Non-IID data distribution methods, we applied a label distribution algorithm, where data is grouped by labels and distributed to clients according to a predefined difficulty. The Non-IID difficulty is determined by the number of unique labels each client receives and represented by Ψ. Smaller Ψ value

Algorithm 3: Genetic Algorithm Implementation

```
input : genes = clients_model, ga_config
population ← initial_population(genes, population_size,
                                chromosome_size)
solution ← tuple(int, float)
n_iter ← 0
while n_iter < max_iter do
    scores ← fitness(population)(3)
    solution ← update(solution, scores)
    population ← selection(population, scores)
    children ← ∅
    for i in range(start=0, stop=|population|, step=2) do
        p1 ← population[i]
        p2 ← population[i + 1]
        for c in crossover(p1, p2, r_cross) do
            mutation(c, genes, r_mut)
            children ← children + c
    n_iter ← n_iter + 1
return solution
```

indicates a more difficult Non-IID distribution. Each client receives 600 records labelled according to the Ψ values. For example, for $\Psi = 2$, client data holds 600 records of two random labels, 300 records each. Data distribution is considered IID when Ψ is set to 10, indicating that each client receives 60 records from each label. Regarding our approach parameter, we fixed p to 10 and q to 2, designating two unique clients from each of the ten clusters to construct the initial model. Finally, the number of chromosomes is fixed to 200.

For Federated Learning, the configurable parameters used to test the proposed schemes are the following:

- **Aggregation Algorithm:** The aggregation algorithm has the role of aggregating the received models from clients into one new global model. In the experiments, traditional FedAVG is employed to test the performance

of our approach.

- **Client Selector:** In the form of a random client selector that takes a client ratio CR as a parameter where $CR \in \mathbb{Z} : 0 < CR <= 1$. CR indicates the percentage of clients selected randomly in each round.

- **Number of rounds:** A round R is represented by the sequence execution of the following components: selection, training, and aggregation. In the following experiment, R is fixed to 500, which is considered good enough to perceive the performance difference between the frameworks.

- **Number of epochs:** An epoch indicates the number of times a model update occurs on the client device and relates to the computational cost spent on a client's device. Reaching higher accuracy with fewer epochs is considered suitable for lowering the training cost on clients' machines. Traditionally, higher epochs positively impact accuracy. Throughout the experiments, E is set to 1 for FedSGD and $E = 25$ otherwise.

- **Batch size:** B, which describes the training device's memory (RAM) capacity. Devices supporting a higher batch are adequate to finish the training procedure faster. In the experiments, $B = \infty$ for FedSGD configuration. Since the training speed is out of the paper scope, $B = 50$ is used for the other experiments.

- **Learn Rate:** A hyper-parameter, represented by LR, controls the rate of the model's weights updates from the new gradient. The model configuration and the dataset highly affect the value of LR. In order to select the best LR value, which shows the minimum loss, we ran different experiments and noticed that $LR = 0.1$ and $LR = 0.001$ yield superior results when working with MNIST and CIFAR-10 respectively.

- **FedSGD:** Is a federated learning configuration where the client updates the gradient only once and sends it back to the server. FedSGD has a fixed configuration of $E = 1$ and $B = \infty$.

Table 1: CIFAR-10 Model's Configuration

Layer	**Shape**
Input	(3,32,32)
Conv2d	(6,28,28)
Relu	(6,28,28)
MaxPool2d	(6,14,14)
Conv2d	(16,10,10)
Relu	(16,10,10)
MaxPool2d	(16,5,5)
Reshape	(400,)
Linear	(120,)
Relu	(120,)
Linear	(84,)
Relu	(84,)
Linear	(10,)
Softmax	(10,)

6.1. Accuracy Evaluation Compared to Warmup Approach

The experiments detail the evolution of the frameworks' accuracies throughout the rounds. We compare our proposed scheme to the Basic [22], and Warmup [18] approaches. The Basic approach represent the traditional federated workflow and it start with an empty initial model in the first round. On the other hand, for the Warmup approach, the server collects a random 10% of the clients' data to train the initial warmup model for 100 epochs. For our solution, the maximum number of available clients participate in a preliminary round that includes training a model and sending it back to the server to build the initial model. Each set of experiments shows the accuracy under the influence of different label distribution ratios, ψ , of values 2, 4, and 10. The number of rounds R reaches 500. Finally, the selection ratio CR is fixed to 0.1, which indicates that 10% of the client selected in each round.

Figure 7 illustrates the accuracy evolution using the MNIST Dataset. The test is done under the FedSGD environment in figures 7a, 7b and 7c. Our approach and Warmup accuracy start-ups were better due to their initial models trained in the preliminary rounds. Furthermore, we notice a continuous growth for the Basic approach, which starts with low accuracy, especially with a lower ψ value. The accuracy of Basic increases till it reaches almost a similar accuracy to the other approaches with a maximum of 76% compared to 83% for Warmup and 84% for our approach after 500 rounds. This experiment indicates that our approach and Warmup can handle the problem of weight divergence better under FedSGD. The Basic performance achieves equivalent results faster with milder Non-IID constraints (ψ equals 4 and 10). Apart from the challenging distribution, our approach and Warmup attained similar accuracy without any significant slopes.

Figures 7d, 7e, and 7f are another batch of experiments to test the frameworks without the epochs limitation posed by FedSGD. The number of epochs, in this case, is set to 25, $E = 25$ while $B = 50$. The accuracy improvement is evident, achieving a maximum of 90% regardless of the distribution compared to 84% FedSGD. The main difference is that now the frameworks can improve throughout the rounds. Following the outcome of our MNIST experiment, we can observe that our preliminary selection was capable of achieving comparable results with the Warmup approach without sharing raw data to build the initial model.

In Figure 8, we evaluate the accuracy using CIFAR-10 dataset. In addition to the three starting frameworks, we added the clustering approach. And we fixed the number of clusters to 10 and the number of preliminary rounds, where all the clients participate in forming clusters, to 5. Regarding the dataset distribution, we split the dataset first into test data and distributed the rest to 100 clients according to the label distribution. We then used 5% of the distributed data to run the clustering algorithm's preliminary rounds and formed the clusters. Later, we ran different federated learning for each cluster and tested the model accuracy using the mentioned test data after each round. It is also worth

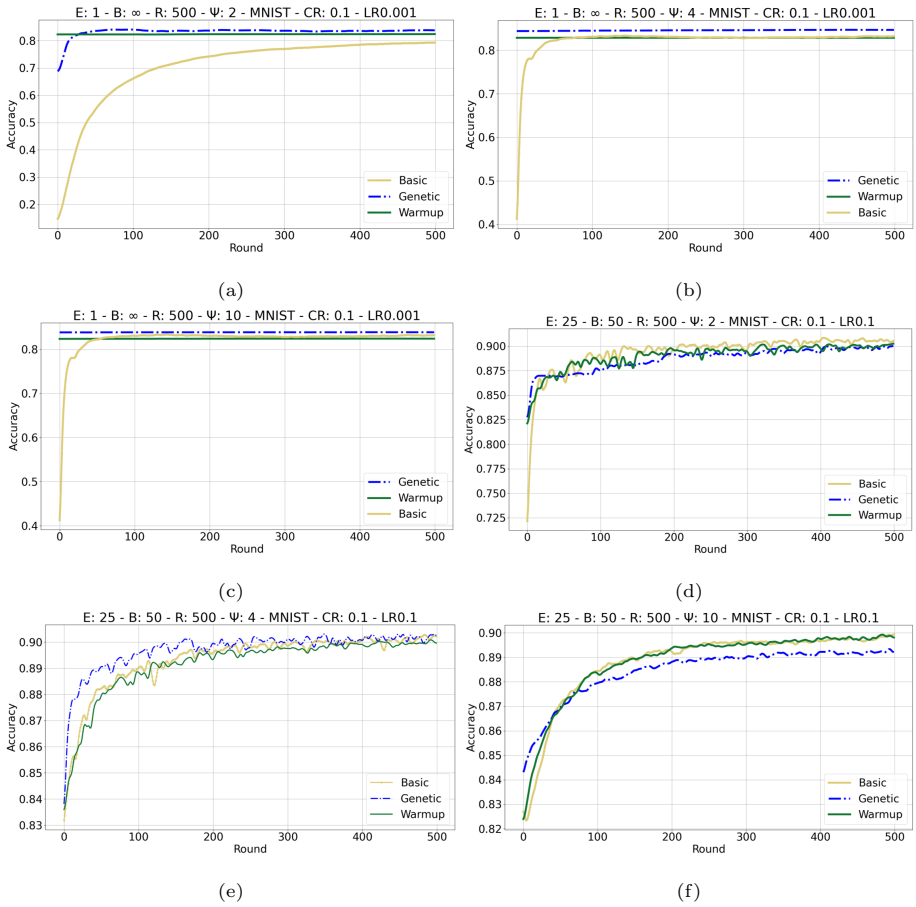

Figure 7: Accuracy progress of Our approach, Basic, and Warmup during the 500 rounds on MNIST dataset with different labels distribution $\psi = [2, 4, 10]$. (A, B, C) presents experiments under FedSGD while (D, E, F) under a preset configuration.

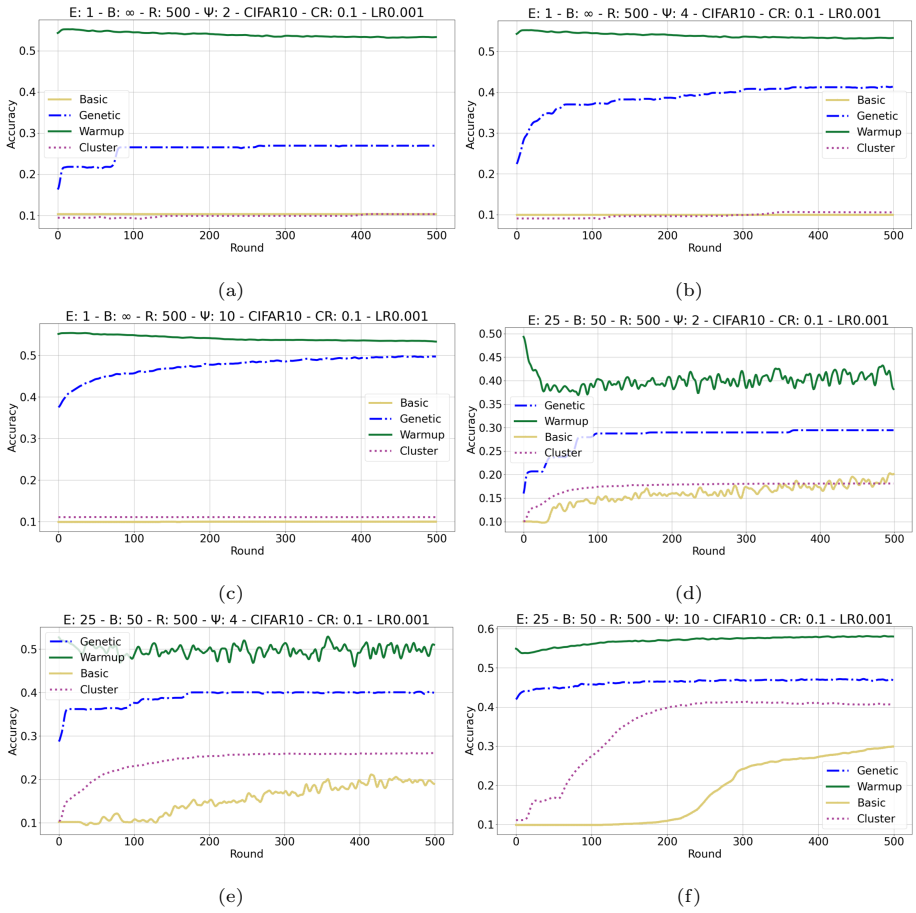

Figure 8: Accuracy progress of Our approach, Basic, and Warmup during the 500 rounds on CIFAR-10 dataset with different labels distribution $\psi = [2, 4, 10]$. Figures (A, B, C) present experiments under FedSGD while (D, E, F) under a preset configuration.

mentioning that the clustering approach requires a dedicated number of clustering that varies according to the data distribution between the clients. Thus, previous knowledge of the data itself and the data distribution is required in order to have the best cluster number, which would yield the best accuracy for each one.

Due to the small number of epochs and the complexity of the dataset, in Figures 8a, 8b, and 8c, Clustering and Basic implementation were not capable of having notable improvement under the FedSGD environment. The Warmup model started with higher accuracy but could not increase in the subsequent rounds. Our approach starts with better accuracy than other approaches, yet below the Warmup, and follows up by slowly improving. In none FedSGD experiments, we remark a noticeable improvement in the Clustering and Basic while the Warmup and our approach remained the same. These experiments indicate that our approach, next to Warmup, can be considered a better solution where the device's capabilities are substandard. The model can improve by lowering the weight divergence even when the constraints (i.e. device limitations) on the federated task are high.

6.2. Weight Divergence Analysis

To test the effect of our selection on the weight divergence, we use the EMD [18] (Earth mover's distance) to measure the average weight divergence between each of the received models and the global model per round. *wd* refers to the weight divergence, and *acc* refers to the accuracy. Figure 9 describes the evolution of EMD based on models trained on the MNIST dataset. In Figure 9. we investigate EMD in a Non-IID context with $\Psi = 2$ and in an IID context, in B, where $\Psi = 10$. The basic framework was last in performance with 0.004 EMD compared to 0.0027 for our approach and 0.0016 for Warmup in the starting round. The performance of our approach and Warmup remains unchanged for the latter experiments on the IID dataset achieving better starting *wd* with values equal to 0.0006 for our approach and 0.0004 for Warmup compared to 0.0011 for Basic.

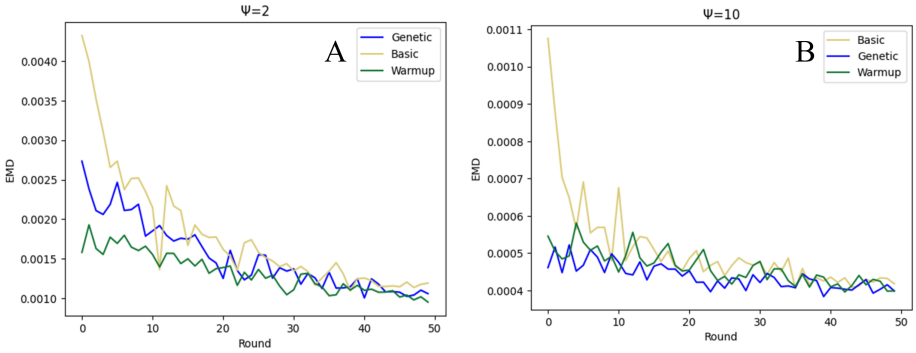

Figure 9: Comparison of weight divergence between two label distributions with $\psi = 2$ in A presenting a Non-IID context and $\psi = 10$ in B presenting an IID context. The experiment is based on the MNIST dataset.

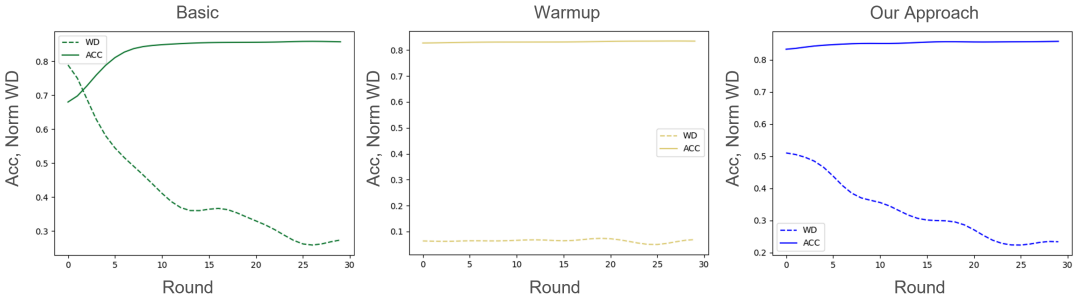

Figure 10: The impact of weight divergence on the accuracy following a Non-IID experiment on MNIST dataset. The dotted lines present the weight divergence *wd* values, while the solid line represents the accuracy.

In order to verify the impact of models' weight divergence on the accuracy, we did the following experiments by showing both EMD and accuracy side by side. To shape the results in a straightforward format solving the issue of small *wd* values compared to *acc*, we took the normalized values of the *wd* according to the Equation 10, wherein our case, X_{min} refers to the lowest *wd* value across the three different experiments. In contrast, X_{max} refers to the highest.

$$X_{norm} = \frac{X - X_{min}}{X_{\max -} X_{min}} \tag{10}$$

In Figure 10 we observe a reverse *wd* effect on the accuracy, especially in the

first few rounds. Furthermore, the Warmup model achieved the most negligible *wd* and the highest starting accuracy compared to the other approaches. Regarding our approach experiments, the application of our selection contribution to the evolution of the model achieving a *wd* smaller than the Basic, which can boost the startup accuracy of the model while achieving faster convergence (5 rounds for our approach compared to 9 for Basic).

6.3. Accuracy Evaluation Compared to Clustering Approaches

According to [24], clustering in FL can improve accuracy and convergence speed. In this experiment, we replaced the Warmup approach with clustering algorithms in which the clients send their trained models to the server instead of their raw data. Later, the server exploits the models' weights and groups the clients into clusters. Finally, the server selects sequentially from each cluster to fill the required number of clients. We applied a modified version of two clustering mechanisms to cope with our initialization strategies and compared them to our approach due to the existing similarities of clients clustering. The main difference between our approach and the clustering ones is the weight divergence minimization of clients model during the initial round, which makes our global model capable of reaching higher accuracy while working under the same configuration. In these experiments, we increased the complexity of the data distribution with additional size un-balancement and randomized label selection to show the capabilities of each approach. We used logistic regression as a ML model for MNIST and CNN for CIFAR-10 as represented in Table 1. Regarding the clustering configuration, we have separated the clients into 5 clusters and selected two clients from each, which makes our initial selection consists of 10 clients.

In Figure 11, we show our approach compared to the K-Means clustering approach using the MNIST dataset under different Non-IID levels (2, 3 and 4). In this experiment, we devise an un-balancement data distribution by generating 25% of the clients holding only 50 records while the rest hold 600. We notice that our approach can boost the start of FL better than K-Means while achieving

convergence faster. During the 100 rounds, the max accuracy of K-Means was always below our approach, while the difference subsided with lower Non-IID difficulty (i.e. higher ψ values). Thus, We can conclude that our approach is more suitable than the standard clustering mechanism when working under harder Non-IID contexts.

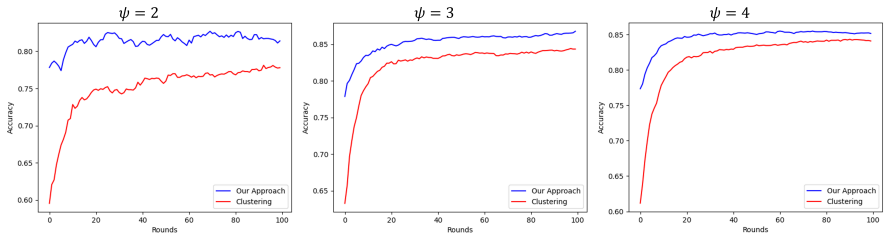

Figure 11: Accuracy progress of our approach compared to K-Means algorithm during the 100 rounds on MNIST dataset with labels distribution $\psi = [2, 3, 4]$ and $E = 1$.

Figure 12 compares our approach to the K-Center algorithm using MNIST and CIFAR-10 datasets under a higher level of Non-IIDness. In this experiment, we increased the distribution difficulty to show its effects on each algorithm. We divided the clients into five sets. Each client in the set holds two labels ($\psi = 2$), with 50% of the clients holding 250 records, 25% holding 50 records and the rest holding only 5 records. Thus, we create higher data deficiencies in some clients, which will significantly impact the global model's progress. We notice that the Base approach is highly affected by this distribution, reaching a maximum accuracy of 0.547 after 500 rounds, which is lower than our previous experiments and remains stagnant when working with CIFAR-10. On the contrary, other approaches achieved higher accuracy due to the initial boosting algorithm. Compared to the K-Center, our approach benefits from the weight divergence minimization to reach even higher accuracy on the MNIST dataset while allowing the model to improve further when working with the CIFAR-10 dataset.

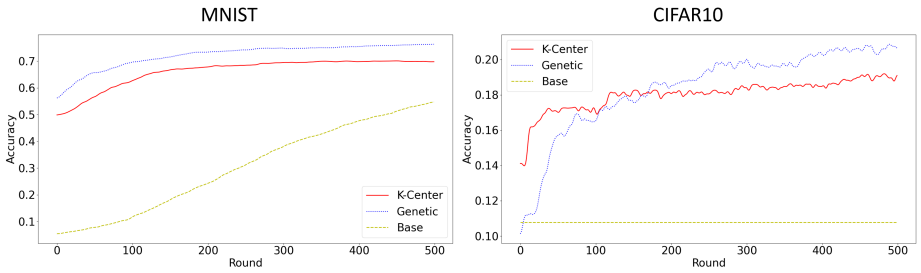

Figure 12: Accuracy progress of our approach compared to K-Center algorithm during the 500 rounds on MNIST and CIFAR10 dataset with labels distribution $\psi = 2$ and $E = 1$.

7. Conclusion

The rapid growth of ML allows its involvement in every aspect of our real life. However, this growth is limited behind the wall of respecting the client's privacy. Federated learning is a practical solution that allows clients to aid model building without breaching its privacy. The ability to replace raw data with models opens opportunities for integrating ML in various sensitive fields. However, due to the host's inability to control data, FL suffers from the non-identical data distribution between clients, limiting the model improvement and increasing the time needed to build a model. Therefore, this paper presents an optimization technique for an initial model built from aggregated models collected from participating clients. These models are compatible, with the minimum diverged weight and shared knowledge across every possible output. Our experiments indicate that our approach is advantageous against available solutions addressing the Non-IID issue and has a massive edge over employing essential FL. In this context, our solution involves a preliminary selector that relies on the model's weight to search for a set of compatible clients to build the initial model. In this context, a client's compatibility can be defined as a set of clients aggregating, combined, a model with minimum weight divergence. Weight distance measurement is employed as a fitness function for divergences evaluation while clustering to avoid node replication. Finally, due to the overwhelming number of possible solutions and the limited available time, the selector integrates

a genetic algorithms approach, which allows our framework to visit the most optimal solutions within the minimum amount of time.

References

[1] F. Guo, F. R. Yu, H. Zhang, X. Li, H. Ji, V. C. Leung, Enabling massive iot toward 6g: A comprehensive survey, IEEE Internet of Things Journal 8 (2021). doi:10.1109/JIOT.2021.3063686.

[2] B. Marr, How Much Data Do We Create Every Day? The Mind-Blowing Stats Everyone Should Read, shorturl.at/EKM15, "[Online; accessed 15-March-2022]" (2018).

[3] J. S. Ng, W. Y. B. Lim, Z. Xiong, X. Cao, J. Jin, D. Niyato, C. Leung, C. Miao, Reputation-aware hedonic coalition formation for efficient serverless hierarchical federated learning, IEEE Transactions on Parallel and Distributed Systems 33 (2022). doi:10.1109/TPDS.2021.3139039.

[4] M. Abououf, H. Otrok, R. Mizouni, S. Singh, E. Damiani, How artificial intelligence and mobile crowd sourcing are inextricably intertwined, IEEE Network 35 (2021). doi:10.1109/MNET.011.2000516.

[5] M. Abououf, S. Singh, H. Otrok, R. Mizouni, E. Damiani, Machine learning in mobile crowd sourcing: A behavior-based recruitment model, ACM Transactions on Internet Technology 22 (2022). doi:10.1145/3451163.

[6] J. C. Jiang, B. Kantarci, S. Oktug, T. Soyata, Federated learning in smart city sensing: Challenges and opportunities, Sensors 20 (21) (2020). doi:10.3390/s20216230.
URL https://www.mdpi.com/1424-8220/20/21/6230

[7] M. Shurrab, S. Singh, R. Mizouni, H. Otrok, Iot sensor selection for target localization: A reinforcement learning based approach, Ad Hoc Networks 134 (2022) 102927. doi:https://doi.org/10.1016/j.adhoc.2022.102927.

URL https://www.sciencedirect.com/science/article/pii/S1570870522001135

[8] A. Hammoud, H. Sami, A. Mourad, H. Otrok, R. Mizouni, J. Bentahar, Ai, blockchain, and vehicular edge computing for smart and secure iov: Challenges and directions, IEEE Internet of Things Magazine 3 (2020). doi:10.1109/iotm.0001.1900109.

[9] M. Bakator, D. Radosav, Deep learning and medical diagnosis: A review of literature, Multimodal Technologies and Interaction 2 (3) (2018). doi:10.3390/mti2030047.
URL https://www.mdpi.com/2414-4088/2/3/47

[10] G. Tello, G. Gianini, R. Mizouni, E. Damiani, Machine learning-based framework for log-lifting in business process mining applications, in: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), Vol. 11675 LNCS, 2019. doi:10.1007/978-3-030-26619-6_16.

[11] S. Abdulrahman, H. Tout, H. Ould-Slimane, A. Mourad, C. Talhi, M. Guizani, A survey on federated learning: The journey from centralized to distributed on-site learning and beyond, IEEE Internet of Things Journal 8 (2021). doi:10.1109/JIOT.2020.3030072.

[12] A. Uprety, D. B. Rawat, J. Li, Privacy preserving misbehavior detection in iov using federated machine learning, in: 2021 IEEE 18th Annual Consumer Communications and Networking Conference, CCNC 2021, 2021. doi:10.1109/CCNC49032.2021.9369513.

[13] A. Hammoud, H. Otrok, A. Mourad, Z. Dziong, On demand fog federations for horizontal federated learning in iov, IEEE Transactions on Network and Service Management 19 (3) (2022) 3062–3075. doi:10.1109/TNSM.2022.3172370.

[14] S. Augenstein, H. Eichner, D. Ramage, G. Llc, M. View, Federated learning for mobile keyboard prediction arxiv:1811.03604v2, Google LLC (2019).

[15] S. Abdulrahman, H. Tout, A. Mourad, C. Talhi, Fedmccs: Multicriteria client selection model for optimal iot federated learning, IEEE Internet of Things Journal 8 (2021). doi:10.1109/JIOT.2020.3028742.

[16] S. A. Rahman, H. Tout, C. Talhi, A. Mourad, Internet of things intrusion detection: Centralized, on-device, or federated learning?, IEEE Network 34 (2020). doi:10.1109/MNET.011.2000286.

[17] S. Arisdakessian, O. A. Wahab, A. Mourad, H. Otrok, M. Guizani, A survey on iot intrusion detection: Federated learning, game theory, social psychology and explainable ai as future directions, IEEE Internet of Things Journal (2022) 1–1doi:10.1109/JIOT.2022.3203249.

[18] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, V. Chandra, Federated learning with non-iid data, arXiv preprint arXiv:1806.00582 (2018).

[19] C. Briggs, Z. Fan, P. Andras, Federated learning with hierarchical clustering of local updates to improve training on non-iid data, in: Proceedings of the International Joint Conference on Neural Networks, 2020. doi:10.1109/IJCNN48605.2020.9207469.

[20] A. Reisizadeh, F. Farnia, R. Pedarsani, A. Jadbabaie, Robust federated learning: The case of affine distribution shifts, in: Advances in Neural Information Processing Systems, Vol. 2020-December, 2020.

[21] M. Wazzeh, H. Ould-Slimane, C. Talhi, A. Mourad, M. Guizani, Privacy-preserving continuous authentication for mobile and iot systems using warmup-based federated learning, IEEE Network (2022) 1–7doi:10.1109/MNET.121.2200099.

[22] B. McMahan, E. Moore, D. Ramage, S. Hampson, B. A. y. Arcas, Communication-Efficient Learning of Deep Networks from Decentralized

Data, in: A. Singh, J. Zhu (Eds.), Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, Vol. 54 of Proceedings of Machine Learning Research, PMLR, 2017, pp. 1273–1282.
URL https://proceedings.mlr.press/v54/mcmahan17a.html

[23] G. Wang, C. X. Dang, Z. Zhou, Measure contribution of participants in federated learning, in: Proceedings - 2019 IEEE International Conference on Big Data, Big Data 2019, 2019. doi:10.1109/BigData47090.2019.9006179.

[24] H. Wang, Z. Kaplan, D. Niu, B. Li, Optimizing federated learning on non-iid data with reinforcement learning, in: Proceedings - IEEE INFOCOM, Vol. 2020-July, 2020. doi:10.1109/INFOCOM41043.2020.9155494.

[25] S. P. Karimireddy, S. Kale, M. Mohri, S. J. Reddi, S. U. Stich, A. T. Suresh, SCAFFOLD: stochastic controlled averaging for on-device federated learning, CoRR abs/1910.06378 (2019). arXiv:1910.06378.
URL http://arxiv.org/abs/1910.06378

[26] H. Wang, M. Yurochkin, Y. Sun, D. Papailiopoulos, Y. Khazaeni, Federated learning with matched averaging, in: International Conference on Learning Representations, 2020.
URL https://openreview.net/forum?id=BkluqlSFDS

[27] M. Duan, D. Liu, X. Ji, R. Liu, L. Liang, X. Chen, Y. Tan, Fedgroup: Ternary cosine similarity-based clustered federated learning framework toward high accuracy in heterogeneous data, CoRR abs/2010.06870 (2020). arXiv:2010.06870.
URL https://arxiv.org/abs/2010.06870

[28] W. Zhang, X. Wang, P. Zhou, W. Wu, X. Zhang, Client selection for federated learning with non-iid data in mobile edge computing, IEEE Access 9 (2021). doi:10.1109/ACCESS.2021.3056919.

[29] A. Ghosh, J. Chung, D. Yin, K. Ramchandran, An efficient framework for clustered federated learning, IEEE Transactions on Information Theory (2022) 1–1doi:10.1109/TIT.2022.3192506.

[30] J. Kang, Z. Xiong, D. Niyato, H. Yu, Y. C. Liang, D. I. Kim, Incentive design for efficient federated learning in mobile networks: A contract theory approach, in: Proceedings - 2019 IEEE VTS Asia Pacific Wireless Communications Symposium, APWCS 2019, 2019. doi:10.1109/VTS-APWCS.2019.8851649.

[31] L. Wang, W. Wang, B. Li, Cmfl: Mitigating communication overhead for federated learning, in: Proceedings - International Conference on Distributed Computing Systems, Vol. 2019-July, 2019. doi:10.1109/ICDCS.2019.00099.

[32] Z. Chai, A. Ali, S. Zawad, S. Truex, A. Anwar, N. Baracaldo, Y. Zhou, H. Ludwig, F. Yan, Y. Cheng, Tifl: A tier-based federated learning system, in: HPDC 2020 - Proceedings of the 29th International Symposium on High-Performance Parallel and Distributed Computing, 2020. doi:10.1145/3369583.3392686.

[33] M. Duan, D. Liu, X. Chen, R. Liu, Y. Tan, L. Liang, Self-balancing federated learning with global imbalanced data in mobile systems, IEEE Transactions on Parallel and Distributed Systems 32 (2021). doi:10.1109/TPDS.2020.3009406.

[34] O. A. Wahab, A. Mourad, H. Otrok, T. Taleb, Federated machine learning: Survey, multi-level classification, desirable criteria and future directions in communication and networking systems, IEEE Communications Surveys and Tutorials 23 (2021). doi:10.1109/COMST.2021.3058573.

[35] J. B. MacQueen, Some methods for classification and analysis of multivariate observations, in: L. M. L. Cam, J. Neyman (Eds.), Proc. of the fifth Berkeley Symposium on Mathematical Statistics and Probability, Vol. 1, University of California Press, 1967, pp. 281–297.

[36] A. Hammoud, A. Mourad, H. Otrok, O. A. Wahab, H. Harmanani, Cloud federation formation using genetic and evolutionary game theoretical models, Future Generation Computer Systems 104 (2020). doi:10.1016/j.future.2019.10.008.

[37] A. Hammoud, H. Otrok, A. Mourad, Z. Dziong, Stable federated fog formation: An evolutionary game theoretical approach, Future Generation Computer Systems 124 (2021). doi:10.1016/j.future.2021.05.021.

[38] L. Deng, The mnist database of handwritten digit images for machine learning research, IEEE Signal Processing Magazine 29 (2012). doi:10.1109/MSP.2012.2211477.

[39] A. Krizhevsky, Learning multiple layers of features from tiny images, Technical Report TR-2009, University of Toronto, Toronto. (2009). doi:10.1.1.222.9220.

[40] J. Schmidhuber, Deep learning in neural networks: An overview, Neural Networks 61 (2015) 85–117. doi:https://doi.org/10.1016/j.neunet.2014.09.003.
URL https://www.sciencedirect.com/science/article/pii/S0893608014002135

Biographies

Mohamad Arafeh

Is currently pursuing the Ph.D. degree with École de Technologie Supérieure, Montreal, QC, Canada. He is working in the area of federated learning.

Hakima Ould-Slimane

Obtained her Ph.D. degree in Computer Science from Laval University, Quebec, Canada. She is currently a professor at the department of mathematics and computer science at Universite de Quebec a Trois-Rivieres (UQTR, Trois-Rivieres, Canada). Her research interests include mainly: information security, cyber resilience, homomorphic encryption, federated learning, preserving data privacy in smart environments, machine learning based intrusion detection, access control, optimization of security mechanisms and security of social networks.

Hadi Otrok

Received the Ph.D. degree in computer science and software engineering from Laval University, Canada, in 2005. He is a Professor with Concordia Institute for Information Systems Engineering, Concordia University, Canada. From 2005 to 2006, he was a Postdoctoral Fellow with Laval University, and then NSERC Postdoc-

toral Fellow at Simon Fraser University, Canada. He is an NSERC Co-Chair for Discovery Grant for Computer Science (2016-2018). His research interests include the areas of computational logics, model checking, multi-agent systems, services computing, game theory, and deep learning.

Azzam Mourad

Received his M.Sc. in CS from Laval University, Canada (2003) and Ph.D. in ECE from Concordia University, Canada (2008). He is currently Professor of Computer Science and Founding Director of the Cyber Security Systems and Applied AI Research Center with the Lebanese American University, Visiting Professor of Computer Science with New York University Abu Dhabi and Affiliate Professor with the Software Engineering and IT Department, Ecole de Technologie Superieure (ETS), Montreal, Canada. His research interests include Cyber Security, Federated Machine Learning, Network and Service Optimization and Management targeting IoT and IoV, Cloud/Fog/Edge Computing, and Vehicular and Mobile Networks. He has served/serves as an associate editor for IEEE Transactions on Services Computing, IEEE Transactions on Network and Service Management, IEEE Network, IEEE Open Journal of the Communications Society, IET Quantum Communication, and IEEE Communications Letters, the General Chair of IWCMC2020, the General Co-Chair of WiMob2016, and the Track Chair, a TPC member, and a reviewer for several prestigious journals and conferences. He is an IEEE senior member.

Chamseddine Talhi

Received the Ph.D. degree in computer science from Laval University, Quebec, QC, Canada, in 2007. He is a Professor with the Department of Software Engineering and IT, ÉTS, University of Quebec, Montreal, QC, Canada. He is leading a research group that investigates smartphone, embedded systems, and IoT security.

His research interests include cloud security and secure sharing of embedded systems.

Ernesto Damiani

is currently a Full Professor at the Department of Computer Science, Universita degli Studi di Milano, where he leads the Secure Service-oriented Architectures Research (SESAR) Laboratory. He is also the Founding Director of the Center for Cyber–Physical Systems, Khalifa University, United Arab Emirates. He received an Honorary Doctorate from Institute National des Sciences Appliquees de Lyon, France, in 2017, for his contributions to research and teaching on big data analytics. He is the Principal Investigator of the H2020 TOREADOR project on Big Data as a Service. He serves as Editor in Chief for IEEE Transactions on Services Computing. His research interests include cybersecurity, big data, and cloud/edge processing, and he has published over 600 peer-reviewed articles and books. He is a Distinguished Scientist of ACM and was a recipient of the 2017 Stephen Yau Award.