NIFL: A Statistical Measures-Based Method for Client Selection in Federated Learning

MOHAMED M’HAOUACH1, ANASS HOUDOU2, HAMZA ALAMI3, KHALID FARDOUSSE1, and ISMAIL BERRADA3

1Sidi Mohamed Ben Abdellah University (USMBA), Fez 30000, Morocco
2International School of Public Health, Mohammed VI University of Health Sciences, Casablanca 82403, Morocco
3School of Computer Science, Mohammed VI Polytechnic University, Ben Guerir 43150, Morocco

Corresponding author: Mohamed M’haouach (mohamed.mhaouach@usmba.ac.ma)

ABSTRACT Federated learning (FL) has been proposed as a machine learning approach to collaboratively learn a shared prediction model. Although, during FL training, only a subset of workers participate in each round, existing approaches introduce model bias when considering the average of local model parameters of heterogeneous workers, which degrades the accuracy of the learned global model. In this paper, we introduce NIFL, a new strategy for worker selection that handles the statistical challenges of FL when local data is Non-Independent and Identically Distributed (N-IID). In NIFL, the server starts sending the signal to the workers that react by sending the number of their samples. The server then selects a percentage of workers with the highest number of samples and requests data statistics such as mean and standard deviation. After that, the server calculates our proposed N-IID index, based on the statistical information collected from the workers without having access to their data, and uses this index as a criterion for worker selection. Finally, the server broadcasts the global model to the selected workers. NIFL takes into account the disparity in the distribution of workers’ data in order to improve the performance of the model in heterogeneous data environment. We have performed several experiments with N-IID data. The obtained results show that both the convergence of our method and the test accuracy increased considerably comparing to the other techniques while keeping a reasonable computation and communication costs.

INDEX TERMS Federated learning, worker selection strategies, privacy-preserving, N-IID.

I. INTRODUCTION The issue of data privacy is now getting a lot of attention and users are more aware of data security and privacy. Additionally, it is becoming more and more difficult to obtain reliable, trustworthy, and accurate data. Large-scale machine learning models were primarily trained in data center environments using potent processing nodes, fast inter-node communication links, and huge training datasets that are centrally accessible. If no real-world data sets are observed, these models cannot be applied. Moving both data gathering and model training to edge devices is essential for machine learning’s future. In this context, Federated Learning (FL), a machine learning paradigm, is proposed to preserve data privacy by enabling edge devices to collaboratively optimize a shared prediction model while keeping all the training data private (data are not shared). Thus, the potential exposure of data and the size of the data shared by users are reduced.

In terms of physical composition, the FL system consists of a central server and multiple distributed nodes. This structure enables each individual device or local worker to train the model on its own data and in its own environment. Computationally, the FL pipeline consists of the following steps. First, the server initializes the system parameters and sends the signal to the participating workers. Then, a subset of workers is chosen based on statistical measures. Next, the aggregation system receives the calculated model parameters from multiple data holders, aggregates these calculated values to...
update the global model, and resends the updated model to the workers involved in the modelling.

Although the total number of workers participating in FL can reach several millions, not all these workers can be available for simultaneous training at the same time due to the limited bandwidth, the processing capacity, the energy cost, and the Non-Independent and Identically Distributed (N-IID) dataset. Therefore, worker selection techniques play a crucial role in FL. These methods aim at selecting a subset of workers based on the above mentioned constraints in order to optimize the global model. The process of worker selection impacts the model’s training time, the convergence speed, the stability of training, as well as, the final accuracy obtained. However, although the convergence of the model is guaranteed when all workers participate with arbitrarily heterogeneous data, the convergence of the scheme with partial worker participation is difficult and strongly dependent on the selection approach.

In this paper, we consider the case where the workers which usually participate in collaborative learning processes are mobile devices consuming computing resources and having restricted power. We also assume that workers’ data distributions can be N-IID in several ways. The main challenges of worker selection in these FL settings are the model bias during the training and possible accuracy degradation. In fact, since the distribution of each worker’s local data set is very different from the global distribution, the local objective of each part is incompatible with the global optima. Thus, there is a drift in local updates, i.e., local models can be updated to local optima, which may be different from global optima. The averaged model may also be far from the global optima especially when the local updates are large [6], [12](e.g., a large number of local epochs). Furthermore, FL relies on stochastic gradient descent (SGD) which is widely used in training deep networks with good empirical performances [13], [14], [15]. The IID sampling of the training data is important to ensure that the stochastic gradient is an unbiased estimate of the full gradient [15], [16], [17]. Nevertheless, such property can hardly be guaranteed in FL.

In this paper, we introduce NIFL (N-IID Index method in FL), a new strategy for worker selection in the context of FL. The main idea of NIFL is to eliminate the high heterogeneous workers that affect the model aggregation. Based on the disparity in the distributions of the workers dataset, NIFL selects workers that are close to each other. Consequently, the collected model weights will not diverge from each other. Therefore, the convergence speed of the global model improves. NIFL processes as follows. First, the server requests and gets the number of samples from each worker. Next, it selects half of the workers without replacement according to their fraction of data and requests for workers’ dataset statistics, namely the means and the standard deviations. Then, in the server side, NIFL computes the N-IID index that represents the disparity between two datasets distributions of two workers. The N-IID index distance allows us to compute and describe the variance between the data distribution of two workers without accessing to their datasets. Finally, based on the computed distances, the server selects the workers which will participate in the training by selecting the workers which are close to each other. For the implementation part, we conduct a set of experiments on FMNIST dataset with various data heterogeneity levels and participation rates. We generate N-IID data following two regimes: random split and label split. Since participants are associated with different environments, the labels distribution of local data are quite different from each other. We keep the training data balanced, so each worker holds the same amount of data. The evaluation results demonstrate that the proposed algorithm outperforms existing worker selection methods in terms of reducing communication round, convergence speed, stability, and accuracy in the most cases of various heterogeneous data distribution properties. Therefore, the main contributions of this paper are listed as follows:

- We introduce NIFL, a new method for worker selection in FL supporting highly unbalanced and N-IID data. The proposed method selects the workers according to the disparity of their dataset distributions, i.e., selecting the workers that are close to each other in terms of dataset distribution. Thus, the model weights do not diverge. Consequently, the accuracy, stability, communication and convergence speed of the collaborative learning model significantly improve.
- We simulate NIFL with extensive experiments using Pyfed framework [18]. We have apply four different partitioning strategies to generate comprehensive N-IID data distribution cases, and we have evaluate our model in terms of the ability to achieve efficiency, accuracy, and parameter sensitivity. The experimental results demonstrate that NIFL can improve model performances under different environment settings. Moreover, the convergence speed can also be improved.

The remainder of this paper is structured as follows. Section 2 describes an overview of related works on worker selection methods. Section 3 presents the necessary preliminaries. Section 4 introduces NIFL. The experiments and our findings are discussed in section 5. Finally, section 6 concludes the paper by summarizing our contributions and draws future research directions.

II. RELATED WORK

In order to guarantee the convergence of the global model in FL, most approaches [5], [6], [7] perform a strategy that selects a set of workers $S^{(t)}$ at iteration $t$ by sampling $m$ workers at random (with replacement) such that the worker $k$ is selected with probability $P_k$. However, recent works propose advanced sampling techniques to accelerate the convergence of federated optimization, i.e., to obtain a better model with higher performance.

Active Federated Learning (AFL): [8] proposes an Active Federated Learning Strategy in which workers are not randomly selected at each round. They are selected using
computed losses of each worker model. The server collects these losses and converts them to probabilities which are used to select the next cohort of workers to train. Losses are computed based on worker model parameters. However, since model transmission is expensive, only fresh losses are obtained from users during an iteration. AFL steps can be summarised as follows: Firstly, the server sends the global model \( \overline{w}^{(t-1)} \) to each worker in the set \( S^{(t)} \). Each worker then trains this model using local data and produces an updated model parameter values \( w_k^{(t)} \), and a loss \( F_k^{(t)} \). Once evaluated, each worker returns a corresponding valuation \( F_k^{(t)} \) to the server, which is used to calculate probability distribution of workers. Finally, the server selects the workers set \( S^{(t+1)} \) using these workers probabilities for the next training iteration.

Power-of-Choice Worker Selection Strategy (POW-d): [9] proposes to select the \( m \) workers with the highest local loss \( F_k(\overline{w}^{(t)}) \) by the current global model at the worker level. Compared to the random selection, POW-d delivers faster convergence to the global minimum. The disadvantage of this selection strategy is that it requires sending the current global model to all \( K \) workers, asking them to evaluate \( F_k \), and requesting their feedback in order to determine the current local loss \( F_k(\overline{w}^{(t)}) \). The fact that frequently there is a large number of workers \( K \) and that these workers have constrained communication and computation capabilities makes the additional communication and computation costs unaffordable.

As a result, the power-of-choice worker selection strategy has been suggested. In this strategy, the central server selects the active worker set \( S^{(t)} \) in the manner described below:

1) Sample the candidate worker set. In order to choose the worker \( k \) with probability \( p_k \), the percentage of data at the \( k^{th} \) worker for \( k = 1, \ldots, K \), the central server samples a candidate set \( A \) of \( d \) (\( m \leq d \leq K \)) workers without replacement.

2) Estimate the local losses. The workers in the set \( A \) receive the server’s most recent global model \( \overline{w}^{(t)} \). Furthermore, these workers calculate their local loss \( F_k(\overline{w}^{(t)}) \) and send it back to the central server.

3) Select workers with the highest loss. The central server creates the active worker set \( S^{(t)} \) from the candidate set \( A \) by choosing \( m = \max([C.K], 1) \) workers with the highest values \( F_k(\overline{w}^{(t)}) \), with ties broken at random. These \( S^{(t)} \) workers take part in the training during the subsequent round, which consists of iterations \( t + 1, t + 2, \ldots, t + \tau \).

After selecting the set \( S^{(t)} \), the training process of the selected workers immediately starts. The paper also proposes two variants of POW-d:

1) Computation efficient Variant of Power-of-Choice Worker Selection Strategy (CPOW-d). To accommodate practical considerations, the three POW-d phases can be easily modified. For instance, instead of computing the values \( F_k(\overline{w}^{(t)}) = \sum_{i \in B_k} f(\overline{w}^{(t)}, e_i) \), by accessing every local dataset \( B_k \), the authors use an estimate \( \frac{1}{|B_k|} \sum_{i \in B_k} f(\overline{w}^{(t)}, e_i) \), where \( e_i \) is the mini-batch of \( b \) samples, sampled uniformly at random from \( B_k \).

2) Communication and Computation efficient Variant of Power-of-Choice Worker Selection Strategy (RPOW-d). Additionally, the chosen workers for each round submit their averaged cumulative loss over local iterations to reduce local calculation and communication costs, i.e., \( \frac{1}{\tau} \sum_{t=-\tau+1}^{l} \sum_{i \in \xi} f(\overline{w}^{(t)}, e_i) \) as soon as they transmit their local models to the server.

Bandit-based Communication-Efficient Worker Selection Strategies (UCB-CS) [10]: The observed worker’s local loss values are used more appropriately by UCB-CS, which is communication-efficient and does not use stale values. The authors create a discounted UCB index for each worker \( k \in [K] \), at communication round \( t \), and select the \( m \) workers with the largest discounted UCB indices. The objective is to calculate the local customer loss estimate in a reliable manner that avoids noise in the most recent evaluation and discounts the stale values computed in the past. Regardless of the local loss values of recently unselected workers, the exploration term \( U_1(\gamma, k) \) increases by a factor of \( \sigma^2_x \). Consequently, UCB-CS is forced to look for other potential customers with a lower operating value. This can, for example, allow the algorithm to avoid generating the error floor by only taking customers with larger than expected local losses. It also reinforces fairness based on the number of times the worker is explored [9].

III. PRELIMINARY

In this section, we introduce a brief description of FL and data assumption concepts used in the rest of the paper.

A. FEDERATED LEARNING

FL as a machine learning paradigm is composed of three major components: data sources, FL systems, and workers [24], [25], [26]. Figure 1 illustrates the general architecture of a FL framework.
From a functionality point of view, the FL pipeline is defined as follows:
1) Initialization: the server selects a proportion of the workers among all the workers to participate in the FL process. Then, the global model is shared among all the selected workers.
2) Local training: each worker trains its model locally using its own part of data, and then sends the updated model back to the server.
3) Server aggregation: the server aggregates the models trained by the workers into one global model which is then sent back to the workers involved in the next round.

These three steps are repeated until the model converges as follows:

M. M’haouach et al.: NIFL: A Statistical Measures-Based Method for Client Selection in Federated Learning

X

concepts related to IID and N-IID data.

how to handle Non-IID (N-IID) data, we review the main assumptions made in this assumption may not be realistic [3]. Before discussing this assumption is not realistic in a FL environment. Providing a method to compute this index without accessing user data is possible to access the local data of each worker. However, computing the NI index between two workers assumes that it is possible to access the local data of each worker. However, this assumption is not realistic in a FL environment. Providing a method to compute this index without accessing user data could be a good metric that describes how user datasets differ from each other.

B. DATA ASSUMPTIONS

One of the most common assumptions in many machine learning and data analysis tasks is that the given data points are realizations of Independent and Identically Distributed (IID) random variables. However, in practical applications, this assumption may not be realistic [3]. Before discussing how to handle Non-IID (N-IID) data, we review the main concepts related to IID and N-IID data.

1) IID AND N-IID DATA

Let us consider two real-valued random variables X and Y taking values in an interval I ⊆ R. Then, as usual:

- The cumulative distribution functions of X and Y are defined by: F_X(x) = P(X ≤ x) and F_Y(y) = P(Y ≤ y), respectively.
- The joint cumulative distribution function is defined by: F_{X,Y}(x,y) = P(X ≤ x, Y ≤ y).

X and Y are Independent and Identically Distributed (IID), if and only if, the following two conditions are met:

1) X and Y are identically distributed, i.e. F_X(x) = F_Y(x) for all x ∈ I.
2) X and Y are independent, i.e. F_{X,Y}(x,y) = F_X(x)F_Y(y) for all (x,y) ∈ I^2.

In real world scenarios [3], [4], data can be N-IID in several ways. In fact, let us assume that, for a learning task T, the feature set is x, the label set is y, and the data distribution of the worker i is P_i. Thus, the taxonomy of the N-IID data regimes is defined as follows:

- Covariate shift: the marginal distributions P_i(x) may vary across workers, even if P_j(y|x) = P_j(y|x), for all workers i and j.
- Prior probability shift: the marginal distributions P_i(y) may vary across workers, even if P_i(x|y) = P_j(x|y), for all workers i and j.
- Concept shift:

1) Same labels, different features: the conditional distribution P_i(x|y) may vary across workers even if P_j(x) = P_j(x), for all workers i and j.
2) Same features, different labels: the conditional distribution P(y|x) may vary across workers even if P_j(y) = P_j(y), for all workers i and j.

- Unbalancedness: the local data amounts may differ considerably from one worker to another.
- Inter and intra-partition correlation: data across multiple worker devices may be correlated, or different partitions of worker’s data are dependent.

To address the challenge of N-IID data, one way is to consider balancing data between data holders (cell phones, cameras, cars, ...). We can also take into account the amount of worker data to weight the learning models. Zhao et al [11] have proposed the strategy of adding a global dataset shared between the different workers. Their idea consists in pushing a small, uniform and distributed dataset to the participating devices. By creating a small subset of data which is globally shared between all devices, each device holds its private data and the shared one. Thus the data sets look a bit alike and the weights of the trained models will be almost similar.

2) N-IID INDEX (NI) EQUATION

N-IID index [22], [23] is a new measure to describe distributions between two datasets X_i and X_j, defined by the following equation:

\[
NI = \frac{\bar{X}_i - \bar{X}_j}{\sigma(X)}
\]

where, 

- X = X_i ∪ X_j,
- \(\bar{X}\) represents the first order moment,
- \(\sigma(.)\) is the standard deviation (std) used to normalize the scale of features, and
- \(\| . \|\) represents the 2-norm (the Euclidean distance of a vector).

Computing the NI index between two workers assumes that it is possible to access the local data of each worker. However, this assumption is not realistic in a FL environment. Providing a method to compute this index without accessing user data could be a good metric that describes how user datasets differ from each other.
**IV. NIFL: A N-IID INDEX BASED APPROACH FOR FEDERATING LEARNING**

In this section, we introduce NIFL, a new approach to handle N-IID data in a FL settings. NIFL uses four main steps for the distributed training as depicted in Figure 2, namely:

1) Worker selection based on samples.
2) Statistical measures gathering.
3) N-IID Index (NI) computation.
4) Worker selection based on NI.

The NIFL algorithm is depicted in Algorithm 1, and explained bellow.

**A. WORKER SELECTION BASED ON SAMPLES**

In this step, the server first collects the number of samples used in each worker. Then, a list of workers ordered according to the number of samples is computed. Finally, the server selects from this list, the first $\lceil K/2 \rceil$ workers where $K$ is the total number of workers. This process is illustrated in Figure 2, step 1. Selecting workers with a large number of samples will provide meaningful statistical measures and will reduce duplicate information since workers with a large number of samples may contain information similar to information of workers with a small number of samples. Thus, the server decreases the computation and communication cost since $K$ is usually very large.

**B. STATISTICAL MEASURES GATHERING**

After the first selection, the server requests the statistical measures from each selected worker, namely the mean and standard deviation of each feature within the collected data. These measurements are mainly used to compute the N-IID index. The step 2 in Figure 2 shows the process of collecting the statistical measures.

**C. N-IID INDEX COMPUTATION**

Inspired by the idea proposed in [11], we build a method to select workers based on their statistical features without accessing the worker data. The selected subset will participate in the training phase of the global model. The general idea is to select a subset of workers where the distance between the elements of the subset is minimum. In order to find the distance between workers, we use statistical information, namely the mean and the standard deviation received from each worker in the previous steps. Let’s assume that we have a network consisting of a server and two workers which use different datasets but share at least one feature. Let $X_1$ and $X_2$ be the same features provided by the first and second worker, respectively, and $X = X_1 \cup X_2$. To measure the distribution shift between the two datasets, we apply the N-IID index defined as:

$$NI = \frac{\|X_1 - X_2\|_2}{\sigma(X)}$$

(2)

It is worth mentioning that the N-IID index can be calculated in a distributed manner without accessing the worker data. The challenge then is to compute $\sigma(X)$ directly without accessing the worker data. Therefore, we propose a method to compute it in a distributed way.

Let $n_1$ (resp. $n_2$) be the size of $X_1$ (resp. $X_2$). We can simply calculate $X_1$ and $X_2$ by getting the averages directly from the workers as follows:

$$X_1 = \frac{1}{n_1} \sum_{i=1}^{n_1} x_{1i} \quad \text{and} \quad X_2 = \frac{1}{n_2} \sum_{i=1}^{n_2} x_{2i}$$

where $x_{1i}$ and $x_{2i}$ are the $i$-th sample of $X_1$ and $X_2$ respectively. Then, the standard deviation of $X$ can be calculated as:

$$\sigma(X) = \sqrt{\frac{1}{n_1 + n_2} \sum_{i=1}^{n_1 + n_2} (x_i - \bar{X})^2}$$

where $x_i$ is the $i - th$ sample of $X$ and $\bar{X} = \frac{\sum_{i=1}^{n_1 + n_2} x_i}{n_1 + n_2}$. Since $X$ is the concatenation of the two datasets $X_1$ and $X_2$, then:

$$\bar{X} = \frac{\sum_{i=1}^{n_1 + n_2} x_i}{n_1 + n_2} = \frac{\sum_{i=1}^{n_1} x_{1i} + \sum_{i=1}^{n_2} x_{2i}}{n_1 + n_2} = \frac{n_1 \bar{X}_1 + n_2 \bar{X}_2}{n_1 + n_2}$$
Let $m = \overline{X}$, then:

$$\sigma^2(X) = \frac{1}{n_1 + n_2} \left( \sum_{i=1}^{n_1} (x_{1i} - m)^2 + \sum_{i=1}^{n_2} (x_{2i} - m)^2 \right)$$

$$= \frac{1}{n_1 + n_2} \left( \sum_{i=1}^{n_1} (x_{1i}^2 + m^2 - 2x_{1i}m) + \sum_{i=1}^{n_2} (x_{2i}^2 + m^2 - 2x_{2i}m) \right)$$

$$+ \sum_{i=1}^{n_2} (x_{2i}^2 + m^2 - 2x_{2i}m)$$

$$= \frac{1}{n_1 + n_2} \left( \sum_{i=1}^{n_1} x_{1i}^2 + m^2 \sum_{i=1}^{n_1} 1 - 2m \sum_{i=1}^{n_1} x_{1i} \right)$$

$$+ \sum_{i=1}^{n_2} x_{2i}^2 + m^2 \sum_{i=1}^{n_2} 1 - 2m \sum_{i=1}^{n_2} x_{2i}$$

$$= \frac{1}{n_1 + n_2} \left( \sum_{i=1}^{n_1} x_{1i}^2 + n_1m^2 - 2m \sum_{i=1}^{n_1} x_{1i} \right)$$

$$+ \sum_{i=1}^{n_2} x_{2i}^2 + n_2m^2 - 2m \sum_{i=1}^{n_2} x_{2i}$$

Let $\sigma_1$ and $\sigma_2$ be the standard deviation of $X_1$ and $X_2$ respectively, and $m_1 = \overline{X_1}$ and $m_2 = \overline{X_2}$. Then, we have:

$$\sum_{i=1}^{n_1} x_{1i} = n_1m_1 \text{ and } \sum_{i=1}^{n_2} x_{2i} = n_2m_2.$$

Knowing that:

$$\sigma_1^2 = \frac{1}{n_1} \sum_{i=1}^{n_1} (x_{1i} - m_1)^2$$

$$= \frac{1}{n_1} \left( \sum_{i=1}^{n_1} x_{1i}^2 + m_1^2 \sum_{i=1}^{n_1} 1 - 2m_1 \sum_{i=1}^{n_1} x_{1i} \right)$$

$$+ \sum_{i=1}^{n_2} x_{2i}^2 + m_1^2 \sum_{i=1}^{n_2} 1 - 2m_1 \sum_{i=1}^{n_2} x_{2i}$$

$$= \frac{1}{n_1} \sum_{i=1}^{n_1} x_{1i}^2 + m_1^2 - 2m_1^2$$

$$= \frac{1}{n_1} \sum_{i=1}^{n_1} x_{1i}^2 - m_1^2$$

Therefore:

$$\sum_{i=1}^{n_1} x_{1i}^2 = n_1(\sigma_1^2 + m_1^2)$$

In the same way, we have:

$$\sum_{i=1}^{n_2} x_{2i}^2 = n_2(\sigma_2^2 + m_2^2)$$

By replacing each term in $\sigma^2(X)$ by its expression:

$$\sigma^2(X) = \frac{1}{n_1 + n_2} \left( n_1(\sigma_1^2 + m_1^2) + n_1m_1^2 - 2n_1mm_1 \right.$$

$$+ n_2(\sigma_2^2 + m_2^2) + n_2m_2^2 - 2n_2mm_2 \right)$$

$$= \frac{1}{n_1 + n_2} \left( n_1(\sigma_1^2 + m_1^2 + m_2^2 - 2mm_1) \right.$$  

$$+ n_2(\sigma_2^2 + m_2^2 + m_2^2 - 2mm_2) \right)$$

$$= \frac{1}{n_1 + n_2} \left( n_1(\sigma_1^2 + (m_1 - m)^2) \right.$$  

$$+ n_2(\sigma_2^2 + (m_2 - m)^2) \right)$$

Thus, in order to estimate $\sigma(X)$ and therefore to compute the N-IID index as defined in Equation (2), we can use the statistical features $n_1$, $n_2$, $m_1$, $m_2$, $\sigma_1$ and $\sigma_2$ as follows (Equation 3):

$$NI = \left\| \left[ \frac{m_1 - m_2}{\sigma_1} \ldots \frac{m_p - \overline{m_p}}{\sigma_p} \right] \right\|_2 \quad (3)$$

with

$$m = \frac{n_1m_1 + n_2m_2}{n_1 + n_2}$$

Now, let’s assume that both workers share the same $p$ features, where $p \geq 1$, and let $m_1^i$ and $\sigma_1^i$ (resp. $m_2^i$ and $\sigma_2^i$) be the mean and the standard deviation of the worker 1 (resp. worker 2) at the $i^{th}$ feature. Then, the N-IID index is given as follows:

$$NI = \left\| \left[ \frac{m_1^1 - m_2^1}{\sigma_1^1} \ldots \frac{m_1^p - m_2^p}{\sigma_1^p} \right] \right\|_2 \quad (4)$$

where for $i \in \{1 \ldots p\}$, $\sigma^i$ is given by:

$$\sigma^1 = \frac{1}{\sqrt{n_1 + n_2}} \left( n_1(\sigma_1^2 + (m_1^1 - m_1^2)^2) + n_2(\sigma_2^2 + (m_2^1 - m_1^2)^2) \right)$$

$$\ldots$$

$$\sigma^p = \frac{1}{\sqrt{n_1 + n_2}} \left( n_1(\sigma_1^p + (m_1^p - m_1^p)^2) + n_2(\sigma_2^p + (m_2^p - m_1^p)^2) \right)$$

and,

$$m_1^i = \frac{n_1 \times m_1^i + n_2 \times m_2^i}{n_1 + n_2}$$

$$\ldots$$

$$m_2^p = \frac{n_1 \times m_1^p + n_2 \times m_2^p}{n_1 + n_2}$$

More generally, for a network having $K$ workers, if we compute the distances NI between each two workers using Equation (4), then the number of computations is equal to $\frac{k(k-1)}{2}$.

**D. WORKER SELECTION BASED ON THE N-IID INDEX**

After computing the distances between workers using the N-IID index, we select the workers that will participate in the training. First, we sort the distances from the smallest to the largest and we group the two closest workers. Then, we add the next closest one of them until having $C$ workers,
where $C \in [0, 0.5]$ is the fraction of the considered workers. For example, assuming that a network contains 10 workers. Then, after the first selection based on the dataset size, we pick five workers $\{1, 2, 3, 4, 5\}$. The aim is to select only three ($C = 0.3$) workers from this set using the N-IID index.

V. EXPERIMENTAL RESULTS

In this section, we present all the required details about our experiments, including the dataset, experimental settings, and performance evaluation. In order to evaluate NIFL, we built a network that contains one server and 100 workers. We allocate data to $K$ with two hidden layers of 64 units and 30 units, respectively. We implement a deep multi-layer perceptron neural network using Fashion-MNIST dataset [1]. It consists of 60,000 grayscale image, associated with a label from 10 classes. The dataset is balanced i.e. the number of samples of each class is 6000. Fashion-MNIST is designed to serve as a direct drop-in replacement for the original MNIST dataset [2] for benchmarking machine learning algorithms. It shares the same image size and structure of training and testing splits.

In order to generate the N-IID version of Fashion-MNIST dataset for FL training, we simulate 4 split types, classified into two categories, namely the randomized split and the label split:

- Randomized split: we randomly distribute samples of the train set on workers.
- Label split: we define 3 types according to how workers with the same classes will share samples:
  - Type 1: The workers having the same labels also share the same samples.
  - Type 2: The workers share the samples of the same class randomly.
  - Type 3: The workers do not share any samples of the same class, i.e. even if two workers have the same class, they will never share the same samples.

B. EXPERIMENTAL SETTINGS

We implement a deep multi-layer perceptron neural network with two hidden layers of 64 units and 30 units, respectively. We allocate data to $K = 100$ workers, and set the participation fraction $C$ to 0.03, the mini-batch size to $b = 64$, and the batch size is to 10. An SGD optimizer with $\eta = 0.005$ is used, where $\eta$ is decayed by half for every 150, 300 rounds. The number of local epochs is set to $\tau = 30$.

For the implementation of FL, we use PyFed [18] as a benchmarking framework to simulate and run all our

Algorithm 1 Worker Selection Strategy (NIFL)

1: **Input**: $K$, $p$, and $C$
   - $K$: the number of workers.
   - $p$: the number of shared features.
   - $C$: the fraction of the considered workers.
   - $S$: the set of worker’s indices.
   - $N_f$: the set of NI distances
2: **Initialize**: Set $S = \emptyset$ and $N_f = \emptyset$.
3: /* Step 1: Computing the initial set of candidates */
4: Receiving the number of samples from each worker, $n_1, \ldots, n_K$.
5: for each worker $k = 1, \ldots, K$ do
6:   $P_k = \frac{n_k}{\sum_{i=1}^{K} n_i}$
7: end for
8: Set $A = \{[K/2]$ work’s indices sampled without replacement from $\{1, \ldots, K\}$ by $P_k$}
9: /* Step 2: Gathering the statistical measures */
10: Receiving the means $[m_1, \ldots, m_p]^T$, $\ldots$, $[m_{[K/2]}^1, \ldots, m_{[K/2]}^p]^T$
11: and the standard deviations $[(\sigma_1^1, \ldots, \sigma_p^1), \ldots, (\sigma_{[K/2]}^1, \ldots, \sigma_{[K/2]}^p)]$
12: from $[K/2]$ workers in the set $A$.
13: /* Step 3: Computing the N-IID index list */
14: for each worker $i = 1, \ldots, [K/2] - 1$ of the set $A$ do
15:   for each feature index $l = 1, \ldots, p$ do
16:     $m_l' = \frac{n_i m_l^i + n_j m_l^j}{n_i + n_j}$
17:     $\sigma_l' = \sqrt{\frac{1}{n_i n_j} (n_i (\sigma_l^i)^2 + (m_l^i - m_l')^2) + n_j (\sigma_l^j)^2 + (m_l^j - m_l')^2)}$
18: end for
19: $N_I = N_I \cup \{ N_I_{ij} = \left\| \frac{m_l^i - m_l'}{\sigma_l'} \ldots \frac{m_l^j - m_l'}{\sigma_l'} \right\|_2 \}$
20: end for
21: /* Step 4: Selecting C workers */
22: Sort distances $N_I$ from the smallest to the largest.
23: for each distance $d$ in $N_I$ do
24:   $i, j \leftarrow$ indices of the two workers, separated by the distance $d$
25:   if $S = \emptyset$ then
26:     $S \leftarrow S \cup \{i, j\}$
27:   else if $S \cap \{i, j\} \neq \emptyset$ then
28:     $S \leftarrow S \cup \{i, j\}$
29: end if
30: if $\text{len}(S) = [C \times K]$ then
31:   break
32: end if
33: end for
34: return $S$
experiments. PyFed relies on a number of Python libraries: PySyft and PyTorch, supports IID and N-IID data distributions and establishes a methodology for model evaluation and result reproduction.

Finally, all experiments are performed on an Ubuntu 16 Server, having 2 CPU Intel Xeon E5-2603 v4 1.7GHz 15M Cache 6.4GT/s QPI 6C/6T, and 512 GB of RAM, GPU Nvidia Tesla P100 - 16 Go.

C. PERFORMANCE EVALUATION
We perform various experiments to compare NIFL to the existing techniques in terms of test loss and test accuracy with different N-IID splits and in terms of communication and computations costs. NIFL is tested against the following selection strategies using different split techniques:

1) AFL: Active Federated Learning [8].
2) POW-d: Power-of-Choice worker Selection Strategy [9].
3) UCB-CS: Bandit-based Communication-Efficient worker Selection Strategies [10].
4) RAND: Random worker Selection.

1) RANDOMIZED SPLIT
Figure 4 shows the variation of the test loss and the test accuracy according to the communication round number. As we can see, NIFL (niid-index) performs better than AFL and RAND in terms of test accuracy. However, NIFL achieves less scores compared to POW-d and UCB-CS methods. The observations on the training loss are consistent with the test accuracy results.

2) LABEL SPLIT TYPE 0
Figure 5 measures the performance of different techniques assuming that workers which share the same labels have also the same samples. NIFL surpasses the UCB-CS technique; yet, it scores lower than AFL and POW-d. It is worth noting that our method converges rapidly compared to other methods, in this simulation.

3) LABEL SPLIT TYPE 1
Figure 6 illustrates the performance of different methods when the workers share the samples of the same class randomly. Our method is more successful than random selection of workers and UCB-CS techniques in terms of test accuracy.

4) LABEL SPLIT TYPE 2
In the last Figure 7, we plot the evaluation measures considering that the workers do not share any samples of the
same class. It shows that our proposed technique scores better than POW-d, UCB-CS, and random worker selection.

5) COMMUNICATION AND COMPUTATION COSTS
We evaluate the communication and computation efficiency of random worker selection, AFL, UCB-CS, NIFL and POW-d in terms of:

1) **RouAcc**: the number of communication rounds required to reach an accuracy of Acc%. We use Rou70 to compare various strategies.

2) **Tcomp**: the average computation time (in seconds) spent per round. The computation time takes into account the time needed by the central server to choose the workers (as well as the time needed by the d workers to compute their local loss values) and the time needed by the chosen workers to perform local updates.

Table 1 presents the obtained results for the communication and computation evaluations using a randomized split. Our method needs 30 rounds to achieve an accuracy of 70.46%, which is better than random worker selection and worse than other techniques such as POW-d which achieves an accuracy of 70.14% in 19 rounds. However, our method consumes only 32.63 seconds to score 70.14%, which is faster compared to POW-d that requires 63.03 seconds.

| TABLE 1. Comparison of Rou70, Tcomp(sec), and Test accuracy (%) for different sampling strategies, when applying randomized split. |
|-----------------|--------|---------|--------|--------|--------|
| Rand            | 78     | 19      | 70.01% | 70.47% | 70.14% |
| POW-d           | 29     | 63.03%  | 70.84% | 70.46% |
| UCB-CS          | 20     | 31.17%  | 70.14% |        |
| NIFL            | 30     | 32.63%  |        |        |

We also compare the performance of different worker selection techniques applying label splits of type 1, type 2, and type 3. Tables 2, 3, and 4 show the obtained results in terms of Rou70, Tcomp, and accuracy with different data split regimes. In all these evaluations, NIFL requires the fewest number of communication rounds to achieve 70% accuracy compared to other worker selection methods. The results also show that our method scores better than POW-d and UCB-CS in terms of computation time, but slightly worse than random worker selection and AFL.

| TABLE 2. Comparison of Rou70, Tcomp(sec), and Test accuracy (%) for different sampling strategies, when applying label split type 1. |
|-----------------|--------|---------|--------|--------|--------|
| Rand            | 30     | 26      | 73.37% | 71.21% | 72.49% |
| POW-d           | 23     | 36.98%  | 67.16% | 70.34% |
| UCB-CS          | 500    | 18.17%  | 67.16% |        |
| NIFL            | 18     | 73.37%  |        |        |

| TABLE 3. Comparison of Rou70, Tcomp(sec), and Test accuracy (%) for different sampling strategies, when applying label split type 2. |
|-----------------|--------|---------|--------|--------|--------|
| Rand            | 18.17% | 73.37%  | 71.21% | 72.49% | 67.16% |
| POW-d           | 36.98% | 47.37%  | 67.16% | 70.34% |
| UCB-CS          | 19.84% | 36.98%  |        |        |
| NIFL            | 73.37% | 71.21%  |        |        |

| TABLE 4. Comparison of Rou70, Tcomp(sec), and Test accuracy (%) for different sampling strategies, when applying label split type 3. |
|-----------------|--------|---------|--------|--------|--------|
| Rand            | 18.17% | 73.37%  | 71.21% | 72.49% | 67.16% |
| POW-d           | 36.98% | 47.37%  | 67.16% | 70.34% |
| UCB-CS          | 19.84% | 36.98%  |        |        |
| NIFL            | 73.37% | 71.21%  |        |        |
Unlike other methods like AFL, UCB-CS and RAND, our method NIFL performs well in all cases in terms of accuracy, communication, and stability. The stability and rapid convergence of our method is due to the fact that NIFL only selects and focuses on the workers which are close to each other in terms of the distribution of their datasets. This explains why NIFL needs few communication rounds to reach 70% of accuracy in most cases. The other methods do not take into consideration the diversity in the data distribution of the workers. As a result, they need more time to converge and to be stable (see Figure 7).

VI. CONCLUSION

In this paper, we introduce NIFL, a new worker selection algorithm to improve the learning accuracy and the convergence speed of the global model in the case of federated learning. The main idea of our method is to compute distances between workers using statistical measures (the mean and the standard deviation of workers datasets) without accessing the workers data. We select customers who are close to each other based on the disparity of the distributions of the workers’ dataset and eliminate heterogeneous workers that impact the model aggregation. We perform several experiments with N-IID data and the public available FMNIST dataset. The results show that the convergence of our method and the test accuracy significantly increase compared to the other techniques. Moreover NIFL method can improve model performances under different environment settings.

For further works, we intend to add another step in the process of selecting workers according to their loss functions. We will also explore the development of a new metric that provides a description of the data set distribution in a federated learning context and that requires less information from workers.

REFERENCES

[1] H. Xiao, K. Rasul, and R. Vollgraf, “Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms,” 2017, arXiv:1708.07747.
[2] L. Deng, “The MNIST database of handwritten digit images for machine learning research [best of the web],” IEEE Signal Process. Mag., vol. 29, no. 6, pp. 141–142, Nov. 2012.
[3] P. Kairouz et al., “Advances and open problems in federated learning,” Found. Trends Mach. Learn., vol. 14, nos. 1–2, pp. 1–210, Jun. 2021.
[4] L. Liu, F. Zhang, J. Xiao, and C. Wu, “Evaluation framework for large-scale federated learning,” 2020, arXiv:2003.01575.
[5] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, “On the convergence of FedAvg on non-IID data,” in Proc. Int. Conf. Learn. Represent. (ICLR), Jul. 2020, pp. 1–26.
[6] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh, “SCAFFOLD: Stochastic controlled averaging for federated learning,” in Proc. Int. Conf. Mach. Learning. (PMLR), 2020, pp. 5132–5143.
[7] J. Wang, Q. Liu, H. Liang, G. Joshi, and H. V. Poor, “Tackling the objective inconsistency problem in heterogeneous federated optimization,” in Proc. Adv. Neural Inf. Process. Syst., vol. 33, 2020, pp. 7611–7623.
[8] J. Goetz, K. Malik, D. Bui, S. Moon, H. Liu, and A. Kumar, “Active federated learning,” 2019, arXiv:1909.12641.
[9] Y. J. Cho, S. Gupta, G. Joshi, and O. Yagan, “Bandit-based communication-efficient worker selection strategies for federated learning,” in Proc. 54th Asilomar Conf. Signals, Syst., Comput., Nov. 2020, pp. 1066–1069.
[10] Y. Jee Cho, J. Wang, and G. Joshi, “Client selection in federated learning: Convergence analysis and power-of-choice selection strategies,” 2020, arXiv:2010.01243.
[11] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, “Federated learning with non-IID data,” 2018, arXiv:1806.00582.
[12] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, “Federated optimization in heterogeneous networks,” in Proc. Mach. Learn. Syst., vol. 2, 2020, pp. 429–450.
[13] J. Duchi, E. Hazan, and Y. Singer, “Adaptive subgradient methods for online learning and stochastic optimization,” J. Mach. Learn. Res., vol. 12, no. 7, pp. 1–39, 2011.
[14] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” 2014, arXiv:1412.6980.
[15] L. Bottou, “Large-scale machine learning with stochastic gradient descent,” in Proc. COMPSTAT, 2010, pp. 177–186.

[16] A. Rakhlin, O. Shamir, and K. Sridharan, “Making gradient descent optimal for strongly convex stochastic optimization,” 2011, arXiv:1109.5647.

[17] S. Ghadimi and G. H. Lan, “Stochastic first-and zeroth-order methods for nonconvex stochastic programming,” SIAM J. Optim., vol. 23, no. 4, pp. 2341–2368, 2013.

[18] H. Bouraqadi, A. Berrag, M. Mhaouach, A. Bouhoute, K. Fardousse, and I. Berrada, PyFed: Extending PySyft With N-IID Federated Learning Benchmark.

[19] T. Ryffel, A. Trask, M. Dahl, B. Wagner, J. Mancuso, D. Rueckert, and J. Passerat-Palmbach, “A generic framework for privacy preserving deep learning,” 2018, arXiv:1811.04017.

[20] A. Hard, K. Rao, R. Mathews, S. Ramaswamy, F. Beaufays, S. Augenstein, H. Eichner, C. Kiddon, and D. Ramage, “Federated learning for mobile keyboard prediction,” 2018, arXiv:1811.03604.

[21] J. Rincon, V. Julian, and C. Carrascosa, “FLaMAS: Federated learning based on a SPADE MAS,” Appl. Sci., vol. 12, no. 7, p. 3701, Apr. 2022.

[22] Y. He, Z. Shen, and P. Cui, “Towards non-IID image classification: A dataset and baselines,” Pattern Recognit., vol. 110, Feb. 2021, Art. no. 107383.

[23] W. Casteels and P. Hellinckx, “Exploiting non-IID data towards more robust machine learning algorithms,” 2020, arXiv:2010.03429.

[24] C. Zhang, Y. Xie, H. Bai, B. Yu, W. Li, and Y. Gao, “A survey on federated learning,” Knowl.-Based Syst., vol. 216, Mar. 2021, Art. no. 106775.

[25] M. Aledhari, R. Razzak, R. M. Parizi, and F. Saeed, “Federated learning: A survey on enabling technologies, protocols, and applications,” IEEE Access, vol. 8, pp. 140699–140725, 2020.

[26] S. A. Rahman, H. Tout, H. Ould-Slimane, A. Mourad, C. Talhi, and M. Guizani, “A survey on federated learning: The journey from centralized to distributed on-site learning and beyond,” IEEE Internet Things J., vol. 8, no. 7, pp. 5476–5497, Apr. 2021.

MOHAMED M’HAOUACH received the bachelor’s degree in computer science and mathematics and the master’s degree in big data analytics and smart systems (BDSaS) from USMBA, Fez, Morocco, in 2015 and 2017, respectively. He is currently pursuing the Ph.D. degree in computer science and mathematics and the master’s degree in big data analytics and smart systems (BDSaS) from USMBA, Fez, Morocco, in 2015 and 2017, respectively. His research interests include computer vision approaches, business intelligence, advanced driving assistance systems (ADAS), and intelligent transportation systems.

ANASS HOUDOU received the bachelor’s degree in computer science and mathematics and the master’s degree in big data analytics and smart systems (BDSaS) from USMBA, in 2019 and 2021, respectively. He is currently pursuing the Ph.D. degree with the Mohammed VI University of Health Sciences (UM6SS), Casablanca, Morocco. His research interest includes developing intelligent systems to predict air pollution using machine learning.

HAMZA ALAMI received the Ph.D. degree in computer science from Ibn Tofail University, Morocco, in 2021. He is currently a Postdoctoral Research Associate at Mohammed VI Polytechnic University (UM6P), SCCS, Ben Guerir, Morocco. His research interests include natural language processing, Arabic text processing, question answering, and information extraction.

KHALID FARDOUSSE received the master’s degree in computer science and decision making and the Ph.D. degree in computer science from the Faculty of Science, USMBA, Fez, Morocco, in 2002 and 2010, respectively. He is currently a Computer Science Professor at USMBA. His research interests include the application of computer vision, natural scene pattern recognition, and the driving behavior modeling/analysis in intelligent transportation systems.

ISMAIL BERRADA received the Ph.D. degree in computer science from the University of Bordeaux 1, France, in 2005. He is currently an Associate Professor in computer science at Mohammed VI Polytechnic University (UM6P), SCCS, Ben Guerir, Morocco. His research interests include artificial intelligence’s (AI) applications in multiple domains, such as cognitive radio networks, radio resource management, vehicular ad-hoc networks, road safety, software testing, and verification. He had also worked for five years as an Assistant Professor with the University of La Rochelle, France, and for ten years with USMBA, Fez, Morocco.

***