Journal of the Brazilian Computer Society, 2025, 31:1, doi: 10.5753/jbcs.2025.4324

A Practical Evaluation of a Federated Learning Application in IoT Device and Cloud Environment

Guilherme Nunes Nasseh Barbosa [**Universidade Federal Fluminense** | ***gbarbosa@midiacom.uff.br***]
Diogo Menezes Ferrazani Mattos [**Universidade Federal Fluminense** | ***menezes@midiacom.uff.br***]

Universidade Federal Fluminense, Av. Passo da Pátria, 156 Bloco E, Sala 408 - São Domingos, Niterói, RJ, Brazil.

Received: 29 March 2024 • **Accepted:** 19 October 2024 • **Published:** 17 January 2025

Abstract Internet of Things (IoT) devices have grown exponentially in recent years, resulting in valuable data for machine learning applications. Traditionally, machine learning models require centralized data collection and processing, which is not feasible in the IoT landscape due to high density and growing data privacy concerns. Federated Learning is a trend in this scenario, as it allows collaborative training of models on IoT devices, distributed and without the need to share data. This paper examines a federated learning framework for IoT devices, employing a parameter server topology in a benign node environment without considering strategies for optimizing model performance. The evaluation is conducted in two distinct scenarios: (i) a testbed of IoT devices equipped with ARM processors and limited to 2GB of RAM, and (ii) a virtualized cloud environment with a mixture of resource-constrained virtual machines. The experiments use non-identically distributed (non-IID) datasets—MNIST for the IoT testbed and CIFAR-10 for the cloud environment—evaluated under various client configurations and aggregation strategies. In the IoT device scenario, the framework achieved an accuracy of up to 0.6 after ten rounds of global aggregation, while the cloud environment attained a maximum accuracy of 0.4. These results demonstrate the feasibility of applying FL in resource-constrained IoT environments, with scalability and accuracy influenced by the number of clients and local training epochs.

Keywords: Federated Learning, Internet of Things, Machine Learning

1 Introduction

The rise of the Internet of Things (IoT) provides ubiquitous sensor and computing capabilities to connect a wide range of devices to the Internet. Machine learning techniques have been widely explored to train predictive models, such as intrusion detection, healthcare, transportation, and intelligent cities [Medeiros *et al.*, 2019], aiming to glean valuable insights from data generated by IoT devices. Traditionally, data processing is done on a remote server, focused on learning and data modeling. This approach incurs critical limitations due to the sheer volume of data generated by the IoT device [Neto *et al.*, 2023; Cunha Neto *et al.*, 2024]. According to Cisco, about 850 ZB of data is generated by all the people, machines, and things at the edge of the network [Alli and Alam, 2019]. By contrast, the overall traffic in data centers is only 20.6 ZB [1]. In a recent report, Equinix predicts that service providers will consume 62% of interconnection bandwidth, representing approximately 20.670 Tbps by 2026 [2]. As data volumes at the edge of networks continue to grow, sending all data to remote servers may become unfeasible due to network resources and latency. The use of third-party servers for training machine learning models also raises privacy concerns, such as breaches and information leakage, as training data may contain sensitive information, such as addresses or personal preferences of users [Neto *et al.*, 2023]. Therefore, developing collaborative and efficient machine learning models is necessary to ensure privacy during the training of IoT applications.

Federated learning enables collaborative training in a decentralized way. It allows multiple participants to train a model with the help of a central server, without data sharing [Lim *et al.*, 2020]. An intrusion detection system aimed at IoT devices, each device monitors a local network and acts as a participant in the collaborative training using the data collected from the network as a dataset [Neto *et al.*, 2022]. IoT devices must communicate with an aggregator server to perform collaborative training. First, the server creates an initial model with random parameters, and each participant will start training using that model. Participants receive the initial model and update it using their local dataset. Then, each participant submits their updated model to the aggregator server. The server combines the updates from each local model and creates a global model. It should be noted that the server aggregates the local models of only a subset of randomly selected participants. Finally, participants receive the aggregated global model and compute the updates with their local data again [Nguyen *et al.*, 2021].

In this paper we present a federated learning framework aimed at the Internet of Things, designed for the practical performance assessment of federated learning applications. The proposed framework adopts the parameter server topol-

[1]Available at `https://www.cisco.com/c/en/us/solutions/executive-perspectives/annual-internet-report/index.html`.
[2]Available at `https://www.equinix.com/gxi-report`.

ogy, where several clients join the federation. Local models are trained on clients and then aggregated on the parameter server. Two test environments were deployed to evaluate the framework: (i) IoT devices equipped with ARM processors and 2GB of RAM memory; and (ii) a virtualized test environment in a private cloud, with virtual machines ranging from 2 to 4 CPU cores and 2 to 8GB of RAM. For evaluation in the IoT device scenario, we use the MNIST [3] dataset, containing 60,000 training samples and 10,000 handwritten digit test samples. In the cloud environment scenario, the CIFAR-10 [4] was used, consisting of 60,000 color images divided in 10 classes, with 6,000 images per class, 50,000 training images and 10,000 test images. The framework was implemented in Python, using the TensorFlow library [5]. The test results demonstrate that, even with limited computational resources, the federated model achieves an accuracy of 0.6 in just 10 rounds of global aggregation, regardless of the number of training epochs of the local models.

Previous studies on federated learning in IoT devices typically evaluate application performance using simulators [Ciftler *et al.*, 2020; Neto *et al.*, 2022] or devices with higher computational power than real devices [Zhang *et al.*, 2021; Mills *et al.*, 2019]. Analyzing and assessing the challenges of federated learning for IoT devices in simulated environments through high-performance computers is complex. Similarly, using devices such as *Raspberry*, which have superior computational performance compared to traditional IoT devices, is also challenging. Thus, there is a need for a validation environment with limited computational resources to analyze and evaluate the main proposals of federated learning in real Internet of Things environments. While federated learning holds promise for improving privacy and efficiency in distributed environments, this study is limited to examining its behavior within a benign node environment. This choice allows for a focused evaluation of federated learning under typical, non-adversarial conditions. As such, optimization strategies for model performance, including dealing with adversarial nodes or network heterogeneity, are beyond the scope of this work and will be considered in future studies.

The reminder of the paper is organized as follows. Section 2 discusses the related work. Federated learning and challenges of IoT devices are described in Section 3. Section 4 describe the evaluation scenarios and the employed datasets. Section 5 evaluates the accuracy obtained using different approaches. Finally, Section 6 concludes the paper.

2 Related Work

Federated learning is an enabling technology for deploying machine learning-based applications on Internet of Things devices. Nguyen *et al.* [2021] highlight that federated learning improves data privacy, reduces network communication latencies, and improves learning quality on constrained devices in the Internet of Things processing [Nguyen *et al.*, 2021]. Recent work focuses on developing new aggregation and participant selection techniques for enhancing learning efficiency on resource-constrained devices [Lai *et al.*, 2021; Neto *et al.*, 2022; Mills *et al.*, 2019]. Also, it focuses on developing new smart applications based on a federation of models [Yu *et al.*, 2018; Wang *et al.*, 2020; Md. Fadlullah and Kato, 2022].

In a previous work, Neto *et al.* [2022] propose the federated optimization of hyperparameters of the Federated Average (FedAvg) algorithm [Neto *et al.*, 2022]. The authors argue that the optimized selection of participants tends to improve the accuracy and decrease the loss of the federated global model. To this end, the authors propose the Federated Simulated Annealing (FedSA), an extension of the simulated annealing (SA) meta-heuristic to a distributed execution scenario where the objective function tends to undergo modifications with each new round. The results show that the proposal can achieve the same accuracy as aggregation with FedAvg [Hard *et al.*, 2018] but with fewer rounds of aggregation and fewer participants. Also aiming to improve the performance of the federated learning algorithm, Mills *et al.* [2019] propose to adapt the FedAvg to use a distributed variation of Adam optimization algorithm, reducing the number of rounds for convergence, along with the new compression techniques, to make the FedAvg efficient in the transmission of weights [Mills *et al.*, 2019]. Lai *et al.* [2021] propose the aggregation method called Oort [Lai *et al.*, 2021]. The Oort proposal prioritizes selecting participants with data that offers the most significant utility in improving the model's accuracy and who can perform the training quickly. The Oort proposal imposes requirements on the distribution of participant data, reducing the execution time of federated learning.

To avoid privacy breaches due to frequent data transfer between edge and cloud, federated learning employs edge computing and uploads updated models from the edge server to the central server for aggregation, rather than directly transferring data. However, a malicious Edge Server can infer the update of others from the aggregate model, and the update can still expose some characteristics of the data of other servers or even Byzantine poisoning attacks. Liu *et al.* [2022] propose a privacy-preserving FL scheme with robust aggregation in edge computing called FL-RAEC [Liu *et al.*, 2022]. By utilizing a hybrid privacy-preserving mechanism, data integrity and privacy are ensured when the model is sent to the cloud server. During the aggregation stage of the model, the authors propose a phased aggregation strategy that incorporates anomaly detection and anonymous trust checking. The trustworthiness is estimated based on the results of an autoencoder-based anomaly detection. Local differential privacy consists of adding noise to the weight update to satisfy the privacy requirement, as measured by the Hamming code distance. Similarly, Lu *et al.* [2020] integrate a local differential privacy mechanism to preserve the privacy of local updates with descending gradient training to enable secure and robust sharing in federated learning. To reduce communication costs, the authors propose a decentralized federated learning model that allows aggregating updates of participant models on distributed servers [Lu *et al.*, 2020].

The reliance on a remote cloud server for federated learning operations can result in long communication latency. Therefore, Zhou *et al.* [2020] present an optimization to co-

[3] Available at http://yann.lecun.com/.
[4] Available at https://www.cs.toronto.edu/ kriz/cifar.html
[5] The source code is available upon request from the authors.

ordinate edge and cloud devices while minimizing communication latency. Shared data scheduling, admission control, and accuracy tuning are optimized together. The simulation results verify the feasibility of the proposed algorithm with reduced latency and increased privacy in various network configurations [Zhou *et al.*, 2020]. Chiu *et al.* [2020] note that, in real-world applications, the data on the end devices is non-independent and is not identically distributed (non-IID) [Chiu *et al.*, 2020]. The distribution of the data can cause weight divergence during training and result in a considerable decrease in the performance of the federated model. Thus, the authors propose an operation called Federated Swap (FedSwap) to replace partial federated learning operations based on some data shared during federated training. A semi-supervised learning scheme is adopted to predict objects for video analytics applications between edge devices.

Savazzi *et al.* [2020] propose a fully distributed learning approach, with no parameter aggregation server [Savazzi *et al.*, 2020]. The concept behind federated learning algorithm proposal is to promote device collaboration by alternating between local computations and mutual interactions through consensus mechanisms. The experimental datasets collected within an Industrial IoT environment (IIoT) assesses the proposed methodology. Kong *et al.* [2020] also focus on developing a federated learning framework for the IIoT environment [Kong *et al.*, 2020]. The authors develop a federated tensor framework for data mining in industrial IoT. The proposal integrates data from multiple sources, enabling tensor-based mining with security guarantees. Industry plants cooperate to get into tensor mining by sharing their data, which has been encrypted using a homomorphic encryption technique, with a centralized server. Thus, the server only collects the encrypted data and federates it on a tensor, while the raw data is kept in the local plants, keeping data privacy. While eavesdroppers can attack the centralized server to compromise the aggregated ciphertext, and attackers can read the ciphertext on communication channels, they cannot obtain the key for the decryption of the data. While these works aim to develop federated applications for massive environments on IoT devices, the proposals do not consider the devices' limited capabilities.

This study distinguishes itself by conducting real-world federated learning experiments on actual low-power IoT devices equipped with ARM processors and 2GB of RAM, closely resembling the constraints of many real-world IoT deployments. By incorporating a private cloud environment, we evaluate the scalability of our proposed framework in a heterogeneous setting and simulate the common scenario of cloud-based computational offloading. This dual-environment approach ensures our framework's adaptability to isolated IoT setups and hybrid configurations leveraging cloud support.

The key innovation lies in its ability to effectively implement federated learning on highly constrained, real-world IoT devices while seamlessly operating in hybrid environments. We employ a parameter server topology with a lightweight client-side implementation optimized for devices with minimal processing power and memory. To minimize communication overhead and ensure compatibility with constrained IoT networks, we utilize HTTP-based communication with base64-encoded messages. Unlike previous work that often relied on simulated conditions or higher-powered hardware, our framework provides a realistic testbed for assessing FL performance in IoT deployments.

By offering insights into the performance of federated learning on low-power IoT devices and cloud environments, this study fills a void in the literature where prior research has been limited to simulations or more powerful hardware setups. Our dual-environment testing provides a comprehensive understanding of how FL can be adapted for various IoT network configurations, ensuring its practical applicability and scalability.

3 Federated Learning on IoT Devices

Federated learning is a type of machine learning that allows multiple devices or clients to collaboratively train a shared model while keeping their data stored locally and private. Unlike traditional machine learning, where a centralized server is responsible for collecting and processing all the data for model training, in federated learning, each device contains private data and participates in the model training process by sending its local updates to the central server, which aggregates the updates to create an improved global model. Federated learning is a specialization of decentralized learning about a unit-height tree topology [Neto *et al.*, 2023]. In decentralized learning, nodes are organized in a hierarchical tree structure, where each node communicates only with its parent and child nodes. In the case of federated learning, the root node is a parameter server responsible for aggregating the local models sent by leaf nodes. In a federated learning scenario, data is always processed on the devices that have it, ensuring data privacy and decreasing the need for bandwidth for data transmission. When training a model with data from multiple sources, federated learning tends to improve the accuracy and generalization of the model. The fundamental federated learning algorithm can be summarized in the following steps [Neto *et al.*, 2023]:

1. **Bootstrap**. The centralized parameter server initializes a machine-learning model and shares it with participant clients;
2. **Participant Selection**. The server selects a subset of participants from the total set of participant clients available for model training. The participant selection can follow different criteria or be random [Neto *et al.*, 2022];
3. **Local Model Training**. Each selected participant receives the current model from the server and performs the training locally using its data. In turn, the participant sends the updated model parameters (not the raw data) back to the server;
4. **Model Aggregation**. The server receives the local model updates from all selected participants and aggregates the models to create a new global model. The main aggregation algorithm is Federated Averaging (FedAvg);
5. **Update Local Models**. After the aggregation of a new global model, the server distributes the new model to all

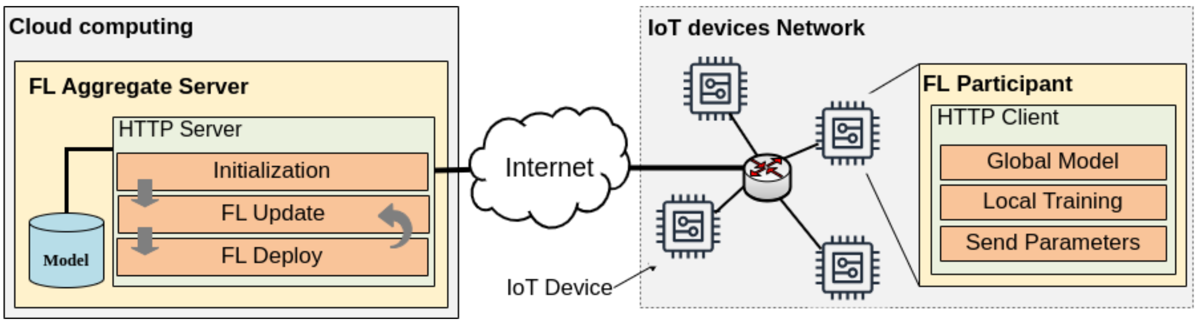


Figure 1. Federated learning framework for IoT devices. The parameter server runs on a cloud computing platform and implements microservices on top of an HTTP server. The federated learning client runs an HTTP client that fetches the global model from the server, trains the local model, and sends the parameters to the server.

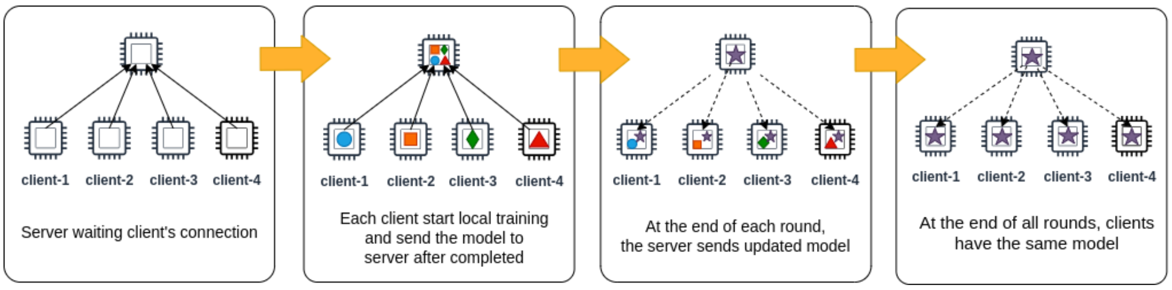

client-1
client-2
client-3
client-4
Server waiting client's connection
Each client start local training and send the model to server after completed
At the end of each round, the server sends updated model
At the end of all rounds, clients have the same model

Figure 2. Experimental scenario adopted. Each client owns a portion of the dataset. The data are non-identically distributed and non-independent (non-IID). The data is local to the clients and is not shared across the network. The exchange of messages boils down to the parameters of the models.

clients participating in the training. Clients then start to use the new model proposed by the server as the basis for training the following rounds;

6. **Repeat**. Steps 2 through 5 constitute a round of model training execution. Therefore, they are repeated for several rounds until the global model meets a predefined stop criterion. The stop criterion is the maximum number of rounds, expected accuracy, or any other model quality metric.

Federated Average (FedAvg) is a commonly used aggregation method for updating the model received from multiple participant clients [Neto *et al.*, 2023]. The technique is widely used in neural network-based learning models, as the model training can be expressed by weight vectors representing each layer of the trained neural network. The key idea of Federated Average is to calculate a weighted average of the local model updates received from each participant client to create a more up-to-date global model. Thus, given a set of N participant clients, each i client performs local training on its data and computes an update vector δw_i. Upon receiving the update vectors from all clients, the server calculates the new global model by taking a weighted average of the updates:

$$w_{t+1} = \frac{\sum_i^n w_{t,i} * n_i}{\sum_i^n n_i}, \tag{1}$$

where $w_{t,i}$ is the vector of client i model weights in round t and n_i is the number of data samples used by client i for local training. The new global model with w_{t+1} weight vector is then used by all clients to train the next round of federated learning. It should be noted that the weight vector contains the weights of all the layers for the neural network-based learning model.

Figure 1 presents the proposal for implementing federated learning in a network of IoT devices. Because IoT devices are limited in processing and memory, the proposal considers that the parameter server runs on a cloud computing platform while the participants are IoT devices. Communication between the participants and the parameter server is carried out using the HTTP protocol. The model and parameter vectors are serialized into a binary message and then encoded in *base64*. The message encoded in *base64* is transmitted as the contents of HTTP messages. The adopted architecture allows the code on the participant to be simple enough to not compromise its capacity. Simultaneously to local model training, the participant client performs only HTTP call actions and encodes messages in base64.

4 Experimental Scenario

Training and testing for the IoT devices' scenario were performed using machine learning algorithms on the MNIST dataset, which comprises several handwritten digit images. The dataset consists of 60,000 samples for training and 10,000 for validation, making it a widely used dataset for image processing tasks. In this proposed scenario, we chose MNIST because of its versatility and small size, which is suitable for devices with limited storage capacity. To carry out the distributed training, IoT devices known as *TV Box*[6] were used as clients. These devices are equipped with ARM Cortex-A53 processors clocked at 1.2GHz and 2GB of RAM. They run the Ubuntu 22.04 LTS operating system with Linux

[6]The term *TV Box* refers to low-cost equipment usually marketed as *set top boxes* for IPTV services. The Federal Revenue Service of Brazil donated the equipment used in this work to the University for recharacterization of the product and its use for social purposes.

kernel version 5.9.0-arm-64, which supports multiple processing cores. The aggregation server runs on an Ubuntu 22.10 virtual machine with Linux kernel version 5.15.0-69-generic in an OpenStack [7] environment.

For the cloud-based heterogeneous environment, we employed the CIFAR-10 dataset for training and testing. CIFAR-10 comprises 60,000 color images, each sized 32x32 pixels, spanning ten distinct classes. The dataset is split into 50,000 training images and 10,000 test images. Notably, the training batches are meticulously balanced across all classes, while the test batch incorporates 1,000 randomly chosen images from each class. In contrast to the prior setup, the cloud-based heterogeneous scenario involves virtual machines operating atop an OpenStack private cloud. These virtual machines boast diverse resource configurations, with virtual CPUs ranging from 2 to 4 cores and RAM ranging from 2 to 8 GB. The CIFAR-10 dataset was chosen for this setting due to its complexness, demanding substantial hardware resources to handle the uneven data distribution effectively.

Python 3.7 [8] was the programming language used, along with the Sklearn[9] and TensorFlow[10] libraries. The architecture of the evaluated IoT scenario is presented in Figure 2.

The advantage of simplification for conducting tests lies in the fact that both the parameter server running on top of OpenStack and all clients are either connected by a local network or sharing a virtual network environment. Every client is initialized with a specific dataset partition. However, it's important to note that the data partition among clients is inherently unbalanced, characterized by a non-identical distribution and non-independence (non-IID) of the data.

The training model adopted in the experimental scenario is the convolutional neural network. For this purpose, the *LeNet-5* network is used, a simple convolutional neural network, as shown in Figure 3. Convolutional neural networks are commonly used in large-scale image processing, as they perform satisfactorily in bitmap segmentation [Andreoni and Mattos, 2021]. The model takes as input a bitmap of dimension 28×28 with one channel, for the IoT device scenario, and a bitmap of dimension 32×32 with three channels, for the cloud-based scenario. The activation function used is the hyperbolic tangent. However, in the last dense layer, the *softmax* activation function is used. The regularization layers apply the average to the regularization window (*AveragePooling2D*).

5 Experimental Results

The datasets were partitioned randomly, resulting in an unbalanced, non-independent, and non-identically distributed pattern. Six scenarios were evaluated for each dataset.The first scenario involved training with two clients, ten rounds, and ten local epochs. The second and third scenarios were evaluated with three and four clients, respectively, while maintaining the same number of local rounds and epochs as the first. The fourth scenario also included training with two clients, maintaining the ten aggregation rounds but using twenty local model training epochs. The fifth and sixth scenarios also used three and four clients, respectively, with twenty local epochs and ten global aggregation rounds.

The first step of the proposal assessment focuses on the MNIST dataset processed by low-cost IoT devices. Figure 4 presents the three scenarios evaluated using ten local epochs, varying the number of clients selected for training the model. Figure 5 shows the training using twenty local epochs for the MNIST dataset.

In scenarios in which only two clients were used, the global model showed minimal accuracy improvement in each training round.It suggests that these clients contributed little to the training process since their combined training data was insufficient. Figures 4(b) and 5(b) depict the scenarios with three clients, with ten and twenty local epochs being utilized, respectively. These scenarios showed an improvement in the accuracy of the aggregation server. In the case of ten local epochs, a significant improvement in server accuracy was observed, with an increase of approximately 40% between the second and third rounds. In the scenario with twenty local epochs, an increase in accuracy of approximately 14% was observed. However, it is unclear whether this improvement is attributable to the specific variation in training or the distribution of the dataset.

Additionally, Figures 4(c) and 5(c) show the scenarios that used four clients, with ten and twenty local epochs, respectively. These scenarios indicate a significant improvement in the accuracy of the aggregation server compared to training with two clients, with an increase of approximately 52% for ten epochs and 34% for twenty epochs. However, the variation between the three and four client scenarios was minor, showing only a 7% improvement in the case of ten local epochs. There was no significant percentage improvement for twenty epochs. Notably, fewer clients can directly influence local accuracy, making it constant. On the other hand, some clients may have data that improves the global model but does not influence the accuracy of the local model.

It is possible to notice a slight difference in accuracy when comparing the scenarios with three and four clients in the aggregation server. However, we found a significant improvement in accuracy when utilizing four clients for training. This indicates that client-4 has a large amount of data crucial for training the global model. Around 40% of the original dataset is allocated to client-4. As the data set is distributed unevenly, such variations are expected. The impact of client-4 on global training can be observed by the fact that when this client starts training with more information, accuracy significantly improves for all other clients.

Regarding the CIFAR-10 dataset, the proposal assessment was conducted on running the federated learning framework in a cloud-based heterogeneous environment. The results show that, similarly to the MNIST dataset results, the number of epochs had little influence on the accuracy during the runs. However, it was possible to observe a new scenario in which a client with a significantly larger sample of data than the other had a greater accuracy than the aggregation server itself. Figures 7(a) show this scenario. It can also be seen that although this event took place, all participants had an improvement in accuracy at the end of the ten rounds. On the

[7]Available at `https://www.openstack.org/`
[8]Available at `https://www.python.org/`
[9]Available at `https://scikit-learn.org/stable/`
[10]Available at `https://www.tensorflow.org/`

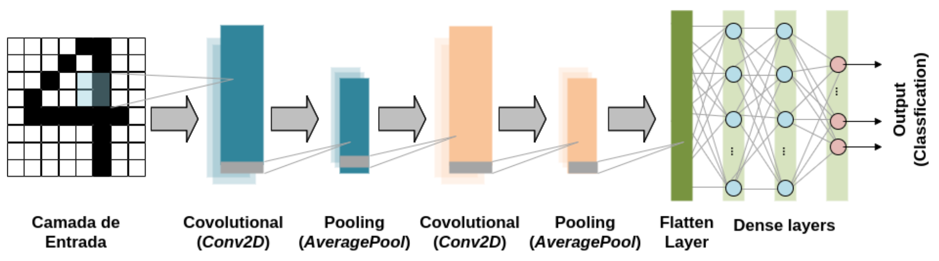
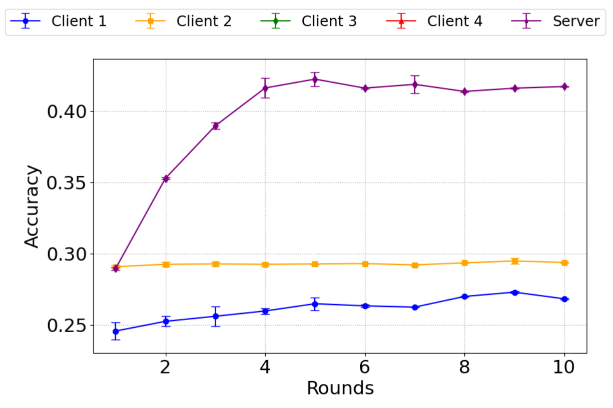
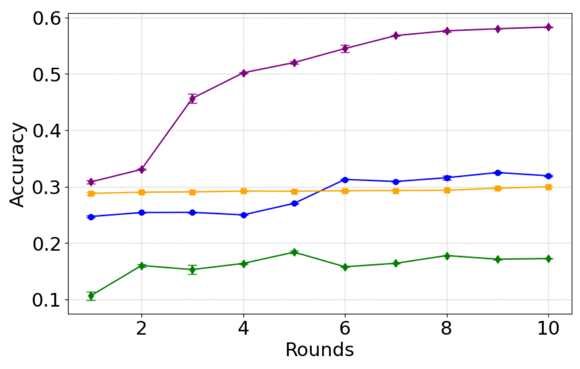


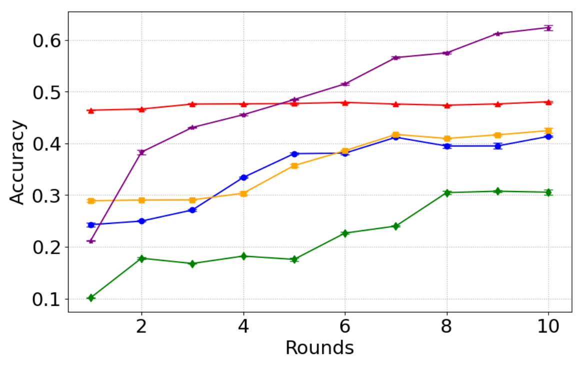
Figure 3. LeNet-5 neural network model trained on federated learning. The model is feed-forward with eight layers, two convolutional layers interspersed with regularization layers, and, at the end, a planning layer precedes three dense layers.



(a) 2 clients



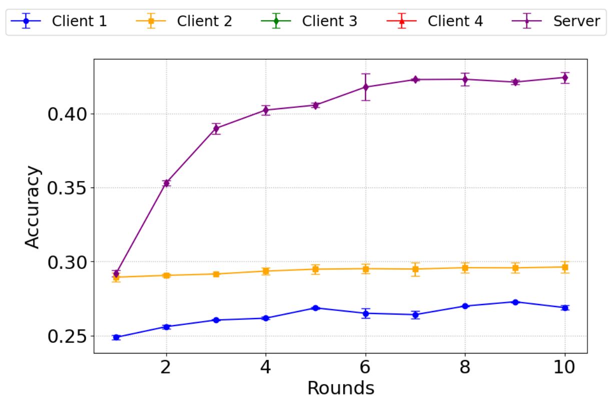
(b) 3 clients



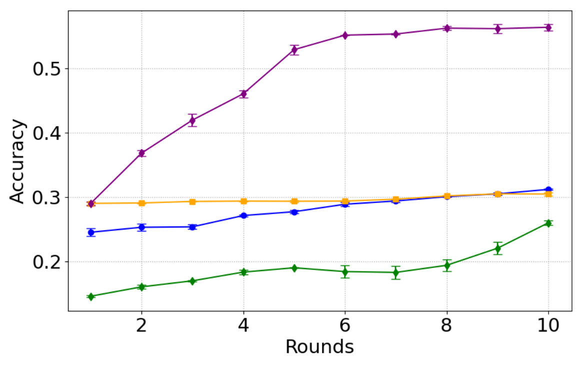
(c) 4 clients

Figure 4. Distributed training using an unbalanced MNIST dataset with ten rounds of global aggregation and ten epochs of local training. a) Training using two clients. b) Training using three clients. c) Training using four clients.

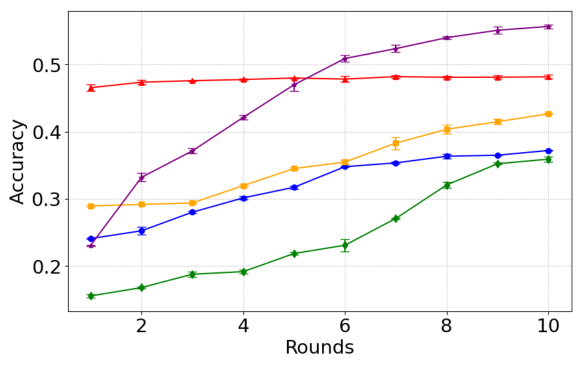
other hand, Figure 6(a) presents a scenario in which clients have an equivalent data sample, and the aggregation server



(a) 2 clients



(b) 3 clients

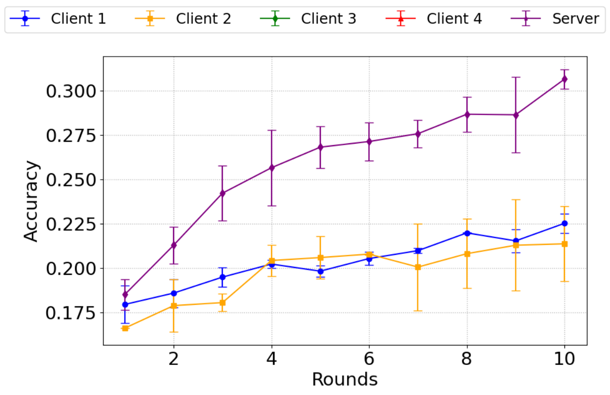


(c) 4 clients

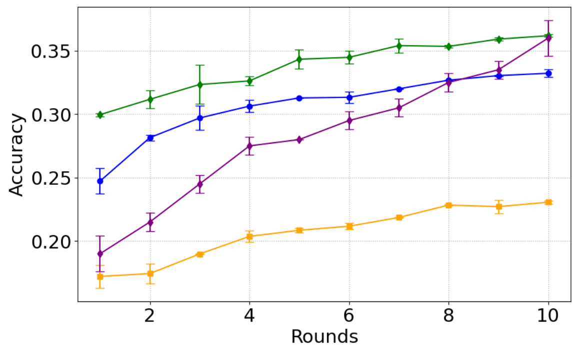
Figure 5. Distributed training using an unbalanced MNIST dataset with twenty rounds of global aggregation and ten epochs of local training. a) Training using two clients. b) Training using three clients. c) Training using four clients.

presents a better overall accuracy.

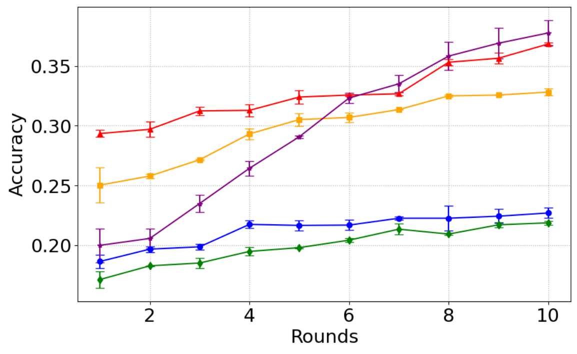
Figures 6(b) and 7(b) present the accuracy using three



(a) 2 clients



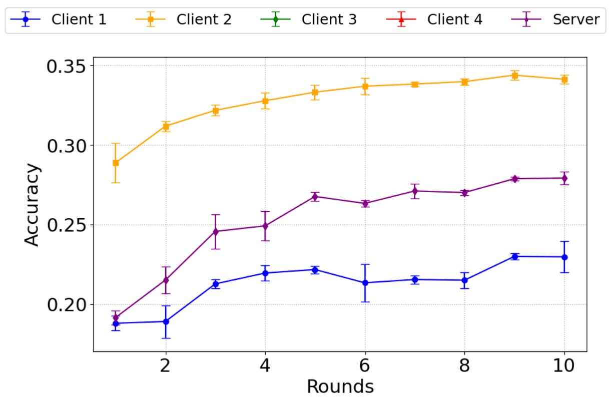
(b) 3 clients



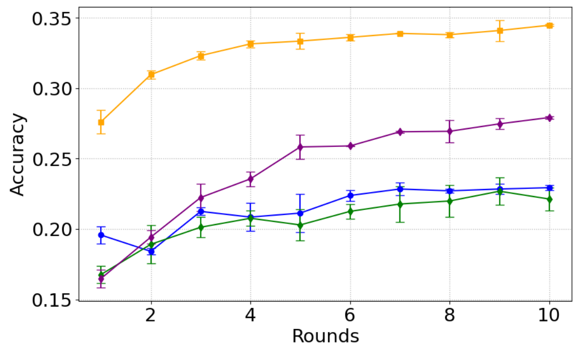
(c) 4 clients

Figure 6. Distributed training using an unbalanced CIFAR-10 dataset with ten rounds of global aggregation and ten epochs of local training. a) Training using two clients. b) Training using three clients. c) Training using four clients.

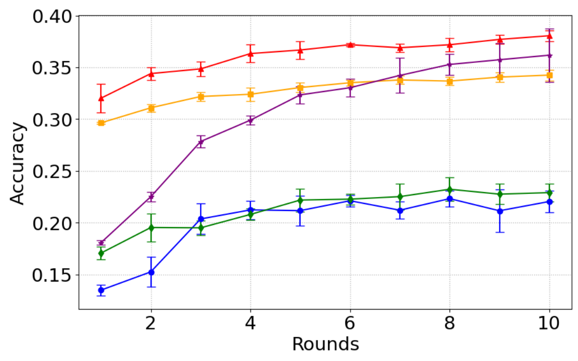
clients over ten rounds, with ten and twenty epochs, respectively. It was possible to observe the same scenario, in which clients with more data stand out in final accuracy, including in front of the aggregation server. However, when the difference in the amount of data between clients decreases, the aggregation server significantly increases, as shown in Figure 7(b). Figures 6(c) and 7(c) demonstrate the accuracy results achieved using four clients for ten and twenty epochs, respectively. The experiment was conducted similarly to the MNIST dataset, where the global accuracy increases with the number of clients. Although the achieved accuracies were relatively low, around 0.35, each round showed a noticeable improvement. It shows that federated learning is a feasible approach from a privacy perspective, as each client has their data, ensuring increased model quality for all participants. Additionally, the results highlight the influence of each client on the training process. Clients with a more significant portion of information have little effect on the final accuracy, indicating that training needs to be continuous and always with new data.



(a) 2 clients



(b) 3 clients



(c) 4 clients

Figure 7. Distributed training using an unbalanced CIFAR-10 dataset with ten rounds of global aggregation and twenty epochs of local training. a) Training using two clients. b) Training using three clients. c) Training using four clients.

6 Conclusion

The upsurge of Internet of Things (IoT) devices has ushered in a new era, embedding processing capabilities into everyday objects. However, these devices strugle with inherent limitations in processing power and memory, often idling during operation. This paper presented and assessed a framework designed to harness federated learning among IoT devices. Federated learning facilitates collaborative training of a global model by diverse clients, each contributing with their local model's weight vector. This approach empowers individual devices to conduct local training on reduced datasets privately, minimizing their computing burden. Through cooperative efforts, the collective array of devices trains a learning model with performance similar to traditional machine learning algorithms and methods. The paper deployed a federated learning prototype on low-cost IoT devices characterized by constrained computing power, evaluating the global model's efficiency on the MNIST and CIFAR-10 datasets. Findings indicated that augmenting the number of participants impacts the models' quality while escalating local training epochs yields negligible enhancements in final classification quality. According to the results, the accuracy achieved for the MNIST dataset was up to 0.6, whereas for the CIFAR-10 dataset, it was up to 0.4. Although this study identifies that increasing data and rounds can enhance model performance, this was not the primary focus of the current research. Instead, future work will focus on deploying the framework in a hybrid environment and evaluating the computational resource efficiency of IoT devices. The expansion of rounds and data is less of a concern because, in a hybrid setting, we expect that cloud resources will offset the limitations seen in purely IoT environments, thus alleviating the need for extensive rounds in such constrained devices.

This paper presented the results obtained through a survey answered by 90 professionals from Brazilian software startups about the UX work in these companies. More precisely, we investigated the UX attitudes and the perception of these professionals regarding the usefulness of UX methods and techniques (RQ1), the moments of Long-Term UX in which these attitudes are held for collecting and evaluating UX data (RQ2), and challenges regarding product, market, team, and the impact of UX attitudes on challenges faced by startups (RQ3).

From the quantitative descriptive and inferential analysis for RQ1, we found that although the most used method, the interview was not considered the most useful in any of the different phases. Usability testing was the most useful for professionals of software startups in stage 1, competitive analysis for professionals of startups in stage 2, and high-fidelity prototyping for professionals of software startups in stages 3 and 4. From this result, we conclude that the perceived usefulness of methods varies according to the stage of the software startup that the professional is inserted. This conclusion suggests caution when recommending or identifying methods to be used in these companies. On the other hand, the methods considered least useful, not surprisingly, are the least used: storyboard and card sorting.

Concerning the results of RQ2, we realized that software startups have been collecting and evaluating UX data in all moments of Long-Term UX. However, interventions that involve more than one moment of Long-Term UX to evaluate user interaction with the product or service received fewer responses. Unlike other literature, Momentary UX is the moment of Long-Term UX at which software startups most often collect and evaluate UX data. Regarding the usefulness of UX data, software startups rate the currently obtained UX data as very useful for software development. However, the amount of professionals who agree with this usefulness decreases according to the stage of the software startup. Finally, improving product/service quality and producing what the user wants to consume are the main motivations for software startups to adopt UX attitudes and methods. These motivations confirm recent findings in the literature on the same topic.

According to the results of RQ3, our study reveals that Brazilian software startups face challenges in applying UX research practices, particularly in team training and defining UX strategies. These challenges, supported by tight development schedules, impact the quality of UX work and ultimately affect customer satisfaction and business success. Financial challenges are also linked to optimizing research and evaluation processes, vital to balancing costs and improving product quality. Furthermore, while startups recognize the importance of analyzing data to generate insights, there is a need to better implement collaborative artifacts to expedite the development of minimum viable products (MVPs). By prioritizing UX as a core component of their strategic planning, startups can overcome these challenges, improve product development, and strengthen their market position.

Options for future works are visualized. From the knowledge of the importance of all UX attitudes, we emphasize that more significant efforts should be made to verify how to motivate and prepare professionals to use UX attitudes in these software startups, aiming to generate a competitive advantage. Our research also opens room to investigate why the interview is the most used UX method but not the most useful in any of the stages. We understand it is useful in future work to investigate which UX attitudes and methods software startups apply for each Long-Term UX moment. We believe it is important to understand what UX data is collected and evaluated by software startups (i.e., user behaviors, user context, or user emotions) and how useful each UX data is for software development. These findings can guide the development of proposals that help software startups professionals to work with UX attitudes, methods, and Long-Term UX in an efficient and lean way.

Declaration

Acknowledgements

The authors thank the Brazilian Federal Revenue for donating equipment utilized in Internet of Things devices.

Funding

This research was funded by CNPq, CAPES, FAPERJ, RNP, and Niterói City Hall/FEC/UFF (PDPA 2020).

Authors' Contributions

All authors were involved in every stage of the work and contributed equally to its completion.

Competing interests

The authors declare no competing interests.

Availability of data and materials

All relevant data and materials are available and can be provided upon reasonable request.

References

Alli, A. A. and Alam, M. M. (2019). SecOFF-FCIoT: Machine learning based secure offloading in fog-cloud of things for smart city applications. *Internet of Things*, 7:100070. DOI: https://doi.org/10.1016/j.iot.2019.100070.

Andreoni, M. and Mattos, D. M. F. (2021). Resumo de grandes volumes de dados com filtro de bloom: Uma abordagem eficiente para aprendizado profundo com redes neurais convolucionais em fluxos de rede. In *Anais do XXXIX Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*, pages 532–545, Porto Alegre, RS, Brasil. SBC. DOI: 10.5753/sbrc.2021.16745.

Chiu, T.-C., Shih, Y.-Y., Pang, A.-C., Wang, C.-S., Weng, W., and Chou, C.-T. (2020). Semisupervised distributed learning with non-IID data for AIoT service platform. *IEEE Internet of Things Journal*, 7(10):9266–9277. DOI: 10.1109/JIOT.2020.2995162.

Ciftler, B. S., Albaseer, A., Lasla, N., and Abdallah, M. (2020). Federated learning for RSS fingerprint-based localization: A privacy-preserving crowdsourcing method. In *2020 International Wireless Communications and Mobile Computing (IWCMC)*, pages 2112–2117. DOI: 10.1109/IWCMC48107.2020.9148111.

Cunha Neto, H. N., Hribar, J., Dusparic, I., Fernandes, N. C., and Mattos, D. M. (2024). Fedsbs: Federated-learning participant-selection method for intrusion detection systems. *Computer Networks*, 244:110351. DOI: https://doi.org/10.1016/j.comnet.2024.110351.

Hard, A., Rao, K., Mathews, R., Ramaswamy, S., Beaufays, F., Augenstein, S., Eichner, H., Kiddon, C., and Ramage, D. (2018). Federated learning for mobile keyboard prediction. *arXiv preprint arXiv:1811.03604*. DOI: 0.48550/arXiv.1811.03604.

Kong, L., Liu, X.-Y., Sheng, H., Zeng, P., and Chen, G. (2020). Federated tensor mining for secure industrial internet of things. *IEEE Transactions on Industrial Informatics*, 16(3):2144–2153. DOI: 10.1109/TII.2019.2937876.

Lai, F., Zhu, X., Madhyastha, H. V., and Chowdhury, M. (2021). Oort: Efficient federated learning via guided participant selection. In *USENIX Symposium on Operating Systems Design and Implementation (OSDI)*. Available at: `https://www.usenix.org/conference/osdi21/presentation/lai`.

Lim, W. Y. B., Luong, N. C., Hoang, D. T., Jiao, Y., Liang, Y. C., Yang, Q., Niyato, D., and Miao, C. (2020). Federated learning in mobile edge networks: A comprehensive survey. *IEEE Communications Surveys & Tutorials*. DOI: 0.1109/COMST.2020.2986024.

Liu, W., Xu, X., Li, D., Qi, L., Dai, F., Dou, W., and Ni, Q. (2022). Privacy preservation for federated learning with robust aggregation in edge computing. *IEEE Internet of Things Journal*. DOI: 10.1109/JIOT.2022.3229122.

Lu, Y., Huang, X., Dai, Y., Maharjan, S., and Zhang, Y. (2020). Differentially private asynchronous federated learning for mobile edge computing in urban informatics. *IEEE Transactions on Industrial Informatics*, 16(3):2134–2143. DOI: 10.1109/TII.2019.2942179.

Md. Fadlullah, Z. and Kato, N. (2022). HCP: Heterogeneous computing platform for federated learning based collaborative content caching towards 6G networks. *IEEE Transactions on Emerging Topics in Computing*, 10(1):112–123. DOI: 10.1109/TETC.2020.2986238.

Medeiros, D. S. V., Cunha Neto, H. N., Andreoni Lopez, M., Magalhães, L. C. S., Silva, E. F., Vieira, A. B., Fernandes, N. C., and Mattos, D. M. F. (2019). Análise de dados em redes sem fio de grande porte: Processamento em fluxo em tempo real, tendências e desafios. *Minicursos do Simpósio Brasileiro de Redes de Computadores-SBRC*, 2019:142–195. Available at: `https://books-sol.sbc.org.br/index.php/sbc/catalog/download/65/287/536?inline=1`.

Mills, J., Hu, J., and Min, G. (2019). Communication-efficient federated learning for wireless edge intelligence in IoT. *IEEE Internet of Things Journal*, 7(7):5986–5994. DOI: 10.1109/JIOT.2019.2956615.

Neto, H. N. C., Dusparic, I., Mattos, D. M. F., and Fernande, N. C. (2022). FedSA: Accelerating intrusion detection in collaborative environments with federated simulated annealing. In *2022 IEEE 8th International Conference on Network Softwarization (NetSoft)*, pages 420–428. DOI: 10.1109/NetSoft54395.2022.9844024.

Neto, H. N. C., Hribar, J., Dusparic, I., Mattos, D. M. F., and Fernandes, N. C. (2023). A survey on securing federated learning: Analysis of applications, attacks, challenges, and trends. *IEEE Access*, 11:41928–41953. DOI: 10.1109/ACCESS.2023.3269980.

Nguyen, D. C., Ding, M., Pathirana, P. N., Seneviratne, A., Li, J., and Vincent Poor, H. (2021). Federated learning for internet of things: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, 23(3):1622–1658. DOI: 10.1109/COMST.2021.3075439.

Savazzi, S., Nicoli, M., and Rampa, V. (2020). Federated learning with cooperating devices: A consensus approach for massive IoT networks. *IEEE Internet of Things Journal*, 7(5):4641–4654. DOI: 10.1109/JIOT.2020.2964162.

Wang, X., Wang, C., Li, X., Leung, V. C. M., and Taleb, T. (2020). Federated deep reinforcement learning for internet of things with decentralized cooperative edge caching. *IEEE Internet of Things Journal*, 7(10):9441–9455. DOI: 10.1109/JIOT.2020.2986803.

Yu, Z., Hu, J., Min, G., Lu, H., Zhao, Z., Wang, H., and Georgalas, N. (2018). Federated learning based proactive content caching in edge computing. In *2018 IEEE Global Communications Conference (GLOBECOM)*, pages 1–6. DOI: 10.1109/GLOCOM.2018.8647616.

Zhang, T., He, C., Ma, T., Gao, L., Ma, M., and Avestimehr, S. (2021). Federated learning for internet of things. In *Proceedings of the 19th ACM Conference on Embedded Networked Sensor Systems*, pages 413–419. DOI: 10.1145/3485730.3493444.

Zhou, Z., Yang, S., Pu, L., and Yu, S. (2020). CEFL: Online admission control, data scheduling, and accuracy tuning for cost-efficient federated learning across edge nodes. *IEEE Internet of Things Journal*, 7(10):9341–9356. DOI: 10.1109/JIOT.2020.2984332.