ISSN 1870-4069

Understanding the Limits of Average Aggregation in Federated Learning Scenarios

Sergio Pérez-Picazo, Hiram Galeana-Zapién[1],
Edwyn Aldana-Bobadilla

Instituto Politécnico Nacional,
Centro de Investigación y de Estudios Avanzados,
México

{sergio.perez.picazo, hiram.galeana,
edwyn.aldana}@cinvestav.mx

Abstract. The prevalent design process nowadays for producing machine learning models requires large datasets that contain different types of data (numerical, categorical, ordinal) depending on the use case or application interest. However, increased Internet users' concerns regarding data privacy might significantly impact the data collection process in specific machine applications, particularly those related to sensitive data like healthcare services. In this regard, the feasibility of federated learning has been investigated, commonly using the seminal Federated Averaging (FedAvg) algorithm to produce a generalized model from the individual models shared by the users through an aggregation process. In this context, this paper aims to contribute to understanding, to a certain extent, the scope of model averaging (used by FedAvg) to identify the cases in which its application is feasible. For this, we adopt a generic federated scenario in which we abstract the selection and configuration stages to focus on the aggregation process. Due to the diversity of possible architectures to perform the aggregation process, we consider three generic architecture families. Through simulation, we quantify the precision achieved by each method under different conditions, such as the number of participants, the type of data used in local learning, etc.

Keywords: Federated learning, model aggregation, neural networks.

1 Introduction

In the last decade, Internet cloud services have allowed users to store and process large amounts of data that can be accessed anytime, anywhere, from personal computers to mobile devices such as smartphones, wearables, home assistants, etc. By gathering data in the cloud, new business models based on information extraction using machine learning (ML) algorithms can be created, transforming decision-making into knowledge-based decision processes. The design process of machine learning algorithms involves training a learning model using a dataset stored in cloud computing facilities. While this is the commonly followed approach nowadays, a growing concern exists regarding data privacy, where users might struggle to share their data for machine learning purposes with a centralized entity on the Internet. This could be the case in specific scenarios dealing with sensitive data, such as healthcare.

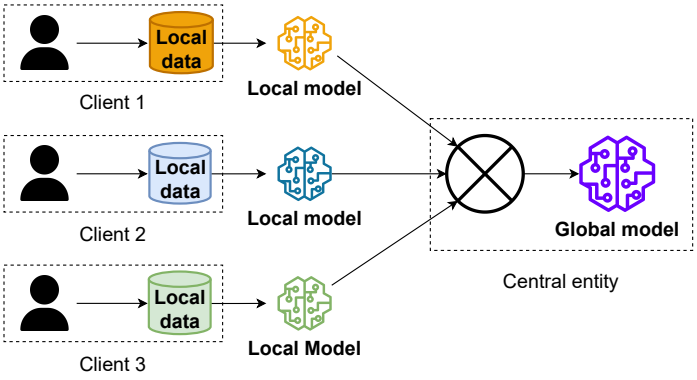


Fig. 1. Federated learning system.

Based on this, Google introduced the term federated learning (FL) in 2017, a brand-new paradigm for machine learning that tackles a few concerns in terms of data privacy, network bandwidth restrictions, and device capabilities (computing, energy, and communication resources). FL emerged as a solution where various users (devices, organizations, etc.) can train a common global model jointly. The training process in federated learning incorporates two stages:

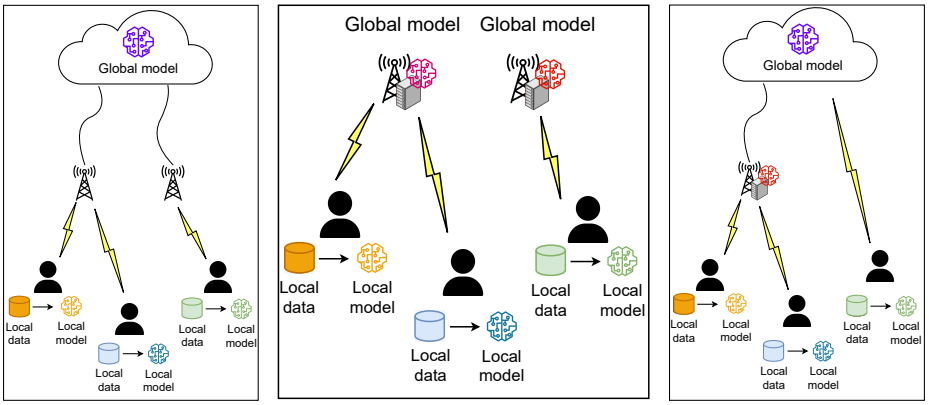
– Each client (device or organization) uses the collected data to train a local model, which is sent to a central entity.

– A centralized entity receives all the local models from clients and executes an aggregation process to create a global model.

At the same time, Google unveiled the first federated learning algorithm called Federated Averaging (FedAvg) [2] based on a progressive training process where model averaging is applied through a three-stage structure deployed in a cloud architecture. For each learning round: 1) a set of clients is selected to participate; 2) each selected client computes an update to the current global model by the server, and only the update is transmitted; and 3) the cloud server aggregates all received local models.

Meanwhile, for the common global model, FedAvg established an aggregation method based on average. The parameters of local models are averaged element-wise with weights proportional to the sizes of the client datasets. The model averaging method emerged as the standard aggregation method used most frequently in academia because of FedAvg's performance on numerous classification and prediction problem tasks. In addition, the development of FL algorithms, mostly FedAvg variants, indicated room for improvement in several areas.

For instance, certain works, such as [5, 4], have investigated how to optimize wireless networks for FL implementation and address issues with wireless communication such as orientation, mobility, and intrusion detection [7]. Other FL algorithms are focused on enhancing the FedAvg structure. For instance, improving the selection stage through better decision-making about which clients could be considered participants in the aggregation process.



(a) Cloud-based FL. (b) Edge-based FL. (c) Hybrid FL.

Fig. 2. Deployment strategies for a FL system.

Meanwhile, the configuration stage is refined in the model training process, specifically by optimizing the clients' computational resources. However, aggregation is barely explored because many FL algorithms have opted for the classic averaging method for model aggregation. In this sense, we identify a need to explore other aggregation methods besides averaging and determine under which conditions an aggregation method is better than others.

In this paper, we provide a first approximation of the aggregation of learning models in the FL paradigm using neural networks with different learning processes (error-based and memory-based). Moreover, we develop a federated learning system to simulate the clients through an on-device learning process, and the central entity that receives the models to start the aggregation process. Specifically, we formulate a scenario like the FedAvg three-stage structure; however, the main goal is to explore how the aggregation process works in different neural networks and compare them under different configurations through performance metrics.

Attending to the arguments, this paper gives an analysis to understand the aggregation process, explore the model aggregation techniques, and show how they can be integrated into neural networks with different learning processes. The rest of the paper is organized as follows. Section 3 presents the related work. Section 4 presents the system model and implementation of the FL system. Section 6 presents the results of some experiments between the different aggregation types. Section 7 details the envisioned contributions and concludes the paper.

2 Preliminaries

In section 2.1, we present a brief description of key concepts related to federated learning, including the deployment strategies (cloud, edge, hybrid) and learning approaches (horizontal and vertical). Then, section 2.2 presents an overview of neural networks, including the most commonly used learning processes (error-based and memory-based).

Table 1. Model aggregation techniques.

Reference	Aggregation technique	Description
[2, 21]	Complete Averaging (CA)	The central entity averages local models of all participating clients.
[21]	Best Half Averaging (BHA)	The central entity averages local models of the best half of participating clients.
[21]	Best Model Selection (BMS)	After receiving all local models, the central entity chooses the best model.

2.1 Overview of Federated Learning

Federated learning is conceived of as a new distributed learning paradigm that has received a lot of interest lately since it addresses some of the privacy and data security concerns identified in the conventional paradigm. A basic federated learning system consists of a group of clients who train a local model based on their local data, upload the model to a central entity, and aggregate the models of each user. Figure 1 depicts a straightforward federated learning system.

Deployment techniques and learning approaches are two crucial elements that must be integrated into a federated learning system. To create a global model in FL, computing entities deployed at a facility for edge computing or cloud computing could potentially be used. Figure 2 illustrates the three deployment options for an FL system, which can be classified as cloud, edge, and hybrid.

Figure 2a shows the cloud-based FL deployment strategy, where each device creates a unique model through a training process with its data; the parameters of these models are then updated on the cloud to produce an aggregated model. Figure 2b shows that the base station's cell site's edge computing nodes will combine the base stations' separate learning models. This form of architecture is suitable from the standpoint of communication to satisfy the latency requirements related to updating the individual models toward the computing unit in charge of the aggregate.

A set of users within a specific coverage region can also acquire more representative global learning models thanks to the FL at the edge. Eventually, you can benefit from both cloud and edge computing by using the hybrid strategy presented in Figure 2c. In other words, a local model aggregation entity at the edge receives model changes from mobile devices quickly. Then, to create a global model, models that were uploaded to the edge nodes are sent to the cloud, where an aggregation process is carried out.

On the other hand, if the data properties of each client are consistent, an FL global model can produce the same modeling results as a centralized paradigm [3]. Three basic categories can be used to categorize federated learning based on the features of the data from the participating clients [27]. Each category is briefly detailed below.

– **Horizontal FL**: Also known as sample-based federated learning, this category is used when datasets share the same feature space but different sample sizes.

– **Vertical FL**: Also referred to as feature-based federated learning, this category can be used when two datasets have different feature spaces but the same sample identification space.

Table 2. Summary of related work.

Reference	FL Algorithm	Deployment Architectures			Machine Learning Algorithms					Experiments Information		Model Aggregation Type
		Cloud	Edge	Hybrid	Memory-based	Error-based				Dataset	Clients	
					RBFN	CNN	MLP	LIR	LOR			
This work, 2023	FedRBFN	✓	-	-	✓	-	✓	✓	-	Synthetic	5-50	CA BHA BMS
[20], 2022	FedBCD	✓	-	-	-	✓	-	-	✓	MIMIC III MNIST NUS-WIDE Default-Credit	-	CA
[26], 2022	FedHome	-	-	✓	-	✓	✓	-	-	MobiAct	57	CA
[14], 2021	SCAFFOLD	✓	-	-	-	-	✓	-	-	MNIST	100	CA
[6], 2020	FedHealth	✓	-	-	-	✓	✓	-	-	UCI Smartphone PCAP COVID-19	5 20 20	CA
[18], 2020	FedProx	✓	-	-	-	-	-	-	✓	MNIST FEMNIST SHAKESPEARE SENTIMENT140	1000 200 143 772	CA
[19], 2020	FedGRU	✓	-	-	-	-	✓	-	-	PeMS	20	CA
[25], 2020	FedMA	-	✓	-	-	✓	-	-	-	CIFAR-10 SHAKESPEARE	16 66	CA
[23], 2020	FedPAQ	-	✓	-	-	-	✓	-	✓	CIFAR-10 MNIST	50	CA
[12], 2020	FedAttOpt	✓	-	-	-	-	✓	-	-	Penn Treebank WikiText-2 Reddit Comments	100	CA
[22], 2019	FedCS	-	✓	-	-	✓	-	-	-	CIFAR-10 Fashion MNIST	100 1000	CA
[17], 2019	FedMD	✓	-	-	-	✓	-	-	-	MNIST FEMNIST CIFAR-10 CIFAR-1000	10	CA
[1], 2019	FedPer	✓	-	-	-	✓	-	-	-	CIFAR-10 CIFAR-100 FLICKR-AES	10 10 30	CA
[2], 2017	FedAvg	✓	-	-	-	✓	-	-	-	CIFAR-10 MNIST SHAKESPEARE	100 100 1146	CA

RBFN = Radial Basis Function Networks, MLP = Multi-Layer Perceptron, CNN = Convolutional Neural Networks, NN = Neural Networks

2.2 Overview of Neural Networks

Neural networks can learn from their surroundings and enhance their performance through learning. A neural network learns through an interactive process of adjusting synaptic weights and bias levels, and over time, performance improves by some predetermined metric. After each iteration of the learning process, the networks should ideally become more informed about their environment. A few learning methods that are useful in the context of machine learning are: 1) error-based learning, and 2) memory-based learning [9].

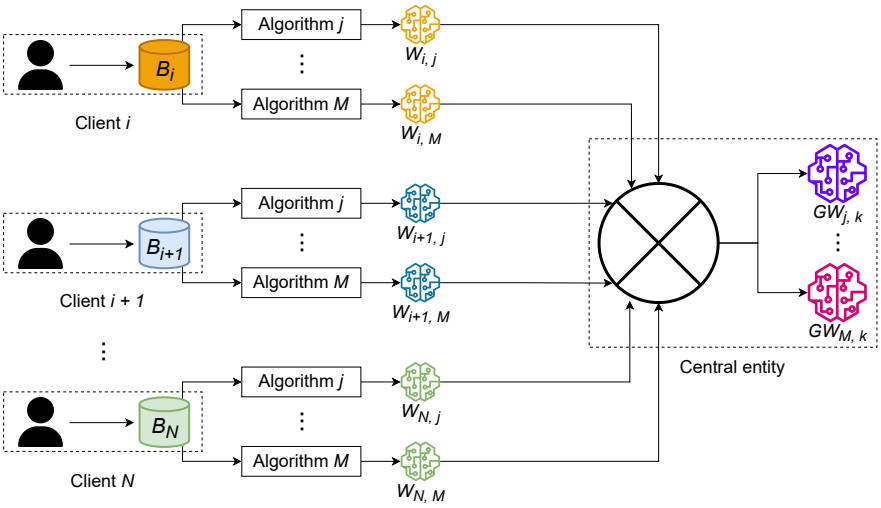


Fig. 3. System model.

In error-based learning, models adjust their parameters based on the discrepancy between their predictions and the true target values in the training data. This is achieved by optimizing a chosen loss function, which measures the error, using techniques like backpropagation. Error-based learning models that are good examples include multilayer perceptron (MLP) and linear regression (LR).

MLP, a feedforward neural network, learns complex non-linear relationships between input features and target labels through hidden layers. MLPs use backpropagation, a popular error-based learning algorithm, to update the network's weights and minimize the error between predicted and target outputs during training. Meanwhile, LR aims to predict the probability of a binary outcome by adjusting its parameters through gradient descent to minimize the error between predicted and actual target values. Both models iteratively refine their parameters to minimize prediction errors and make accurate generalizations about unseen data.

On the other hand, memory-based learning models, such as radial basis function (RBF) neural networks, store training examples explicitly in memory and make predictions based on their similarity to new inputs. RBF networks use radial basis functions to represent the data and find the closest training examples to new inputs during prediction [16]. This memory-based approach allows RBF networks to perform well in interpolation and pattern recognition tasks. Unlike error-based models, RBF does not explicitly minimize a predefined error function but instead retrieves relevant training instances from memory to make predictions.

3 Related Work

In this section, we discuss some federated learning algorithms proposed in the literature that address different concerns and the integration of use cases in several areas such as healthcare, security, business, and industry.

Algorithm 1 On-device learning pseudocode.

Require: Data batch B_i.
Ensure: All local models with structure $W_{i,j}$.

```
for j in M do
    for i in N do
        Split B_i into training and test data.
        Initialize an instance of algorithm j, specifying the parameters.
        Start the training process of the model with the training data.
        Validate through a predicted task with test data.
        Get model performance metric.
        Save local model.
        Transmit local model W_{i,j} to central entity.
    end for
end for
```

As we have mentioned before, most of the proposed solutions are based on the FedAvg algorithm, implementing the three-stage structure, the same aggregation method, and varying the deployment technique as well as the learning approach.

3.1 Aggregation Methods

Aggregation methods in federated learning are essential because of their role in updating global models, whether the messages exchanged are the models themselves, some or all of their parameters, or gradients. However, aggregation in federated machine learning goes beyond simply merging model updates. In addition, aggregation can be carried out hierarchically, aggregating local models on intermediate servers before sending them to the central server, enabling large-scale federated learning systems.

This is why the aggregation algorithm is such a fundamental concept in federated learning; it ultimately determines the success of model training and whether or not the resulting model is practical to use. Some of the aggregation methods are mentioned in Table 1, since there may be approaches other than those presented here.

In federated learning, a variety of aggregation algorithms are used depending on the goals to be achieved, such as protecting user privacy, increasing the convergence rate, and reducing the damage caused by fraudulent customers. Each of these approaches has its advantages and disadvantages, and some are better suited to certain contexts of federated learning than others.

3.2 Federated Learning Algorithms

In federated machine learning, the central entity and clients work together to train a global model. Numerous aggregation techniques have been developed in recent years as a result of the various methods used to aggregate locally trained models. Federated learning became a popular topic that attracted scholars and inspired thousands of experiments in other areas. Table 2 highlights important factors, including the machine learning algorithms utilized, client selection strategies, and model aggregation type. For instance, McMahan et al. [2] proposed a practical approach to federated learning of deep networks that uses an iterative model averaging technique.

Algorithm 2 Model aggregation pseudocode.

Require: All local models with structure $W_{i,j}$.
Ensure: Global model $GW_{j,k}$.

```
Receive all local models from participant clients.
for each algorithm j ∈ M do
    Aggregate models by complete averaging.
    Aggregate models by best half models.
    Aggregate models by best model selection.
    Evaluate and compare all aggregations.
    Save global model GW_{j,k}.
end for
```

In this approach, they carry out an empirical evaluation considering four datasets. The results of the experimentation show that the FedAvg approach trains high-quality models with relatively few rounds of communication. This can be seen in the results from a variety of model architectures, such as a multilayer perceptron and two different convolutional neural networks. Hard et al. [8] proposed an algorithm where a recurrent neural network linguistic model was trained through a distributed learning framework in the device to predict the next word on a virtual keyboard on smartphones.

In addition, a comparison was made between the local models and the global model. In particular, the authors demonstrate the feasibility and benefits of training linguistic models on devices without exporting sensitive user data to the central node. On the other hand, Zhu et al. [28] discuss the development of a non-segmented text recognition model based on a neural network for text recognition of a corpus of Chinese financial documents, which often contain personal, confidential, and critical information.

The comparison of the results showed that the use of an FL framework effectively improves the aggregate performance of the text recognition model without sharing sensitive data. In the healthcare context, some FL approaches have been proposed. Kaissis et al. [13] presented a research paper detailing the benefits of using FL algorithms to improve patient care and, above all, to simultaneously address demands for the protection and use of data.

Meanwhile, Sadilek et al. [24] work demonstrates that creating federated learning models can achieve similar accuracy, precision, and generalizability, and lead to the same interpretation as centralized models while achieving significantly greater privacy protection without significantly increasing computational costs. To do this, they use a diverse set of health studies from one and several health centers.

Nevertheless, federated learning can be applied to contexts related to finance, business, and industry. Hiessl et al. [10] introduced an FL-based industrial system that supports knowledge sharing in the evaluation and continuous updating of learning tasks with sufficient data similarity to enable optimal collaboration of trading partners on common ML problems.

It prevents negative knowledge transfer and ensures resource optimization of the edge devices involved. Kawa et al. [15] developed a method based on federated learning that allows institutions to collaboratively learn a shared prediction model, keeping all the data in the banking institutions. In this way, the ability to do machine learning would be decoupled from the need to store data in the cloud, thus eliminating centralization.

Table 3. Simulation parameter settings.

Parameters	Value			
	System	**RBFN**	**MLP**	**LR**
Number of clients	[10, 20, 30, 40, 50]	-	-	-
Number of samples	125000	-	-	-
Number of clusters	-	8	-	-
Gamma	-	0.0009	-	-
Beta	-	-	0.05	-
Number of iterations	-	-	300	1000
Learning rate	-	-	0.01	0.01

Jansson [11] proposed a framework for training a federated learning fraud detection model in which multiple entities collaborate to solve an ML problem under the coordination of a central server. In this way, financial institutions can collectively reap the benefits of a shared model, which has seen more fraud than each bank alone, without sharing the data set among themselves.

4 System Model

The system modeling is illustrated in Fig. 3. We consider a generic federated learning system with N clients denoted as $i \in \{1, 2, \ldots, N\}$; and M algorithms denoted as $j \in \{1, 2, \ldots, M\}$. Each client i has a data batch denoted as B_i with all the data gathered over a certain amount of time. Additionally, B_i serves as the input data for each algorithm j, which results in the learning model $W_{i,\,j}$.

Each participant client i is assumed to be connected to a cloud facility that provides computing and storage resources. The cloud server aggregates models with $W_{i,\,j}$ structure during each learning round to generate a global model denoted as $GW_{j,\,k}$. Finally, it is important to emphasize that the structure of our FL system is based on the FedAvg with a few modifications.

For instance, no selection criteria were considered, and all clients were eligible to participate in the aggregation process. In addition, transmission rates and backhaul resources are not considered because our analysis is focused on understanding the aggregation methods under different scenarios.

Additionally, we consider the aggregation techniques described in Table 1 because, as we have mentioned before, averaging methods are the most understandable and commonly used methods to aggregate learning models. The CA method is considered the standard method for aggregation since FedAvg, and all its variants have integrated it into their model aggregation process because it ensures that the learning models of all clients are averaged to generate a generalized model. In this sense, the global model generation through CA is expressed as:

$$GW_{j,k} = CA\left(\sum_{i=1}^{N} W_{i,\,j}\right) = \frac{W_{1,\,j} + W_{2,\,j} + \ldots + W_{i,\,j}}{N}. \tag{1}$$

Similar to this, the BHA approach takes into consideration the top half of the client-submitted models. A performance test of the models is used to inform the decision-making process. For instance, let's assume four clients create and evaluate a model, which is later transmitted to the central entity. In this case, the central entity decides through BHA which of all models is going to be aggregated. Equation 2 shows the global model generated in BHA using the same example:

$$GW_{j,k} = \text{BHA}\left(\sum_{i=1}^{N=4} W_{i,\,j}\right) = \frac{W_{1,j} + W_{4,j}}{\frac{N}{2}}. \tag{2}$$

The BMS technique, which differs from the average and involves choosing the best model from the clients, can be used to simplify the aggregation process so that it is easier to grasp. Returning to the previous example in this sense, the difference is that the central entity will now only choose the best model out of all of them. The global model generated by the BMS aggregation can be expressed as follows:

$$GW_{j,k} = \text{BMS}\left(\sum_{i=1}^{N=4} W_{i,\,j}\right) = W_{1,j}. \tag{3}$$

5 Proposed Approach

Our suggested federated learning system follows the conventional three-phase structure of all federated learning algorithms; however, we omit the participant client selection stage because we assume that clients have already been previously selected through a selection technique. In this regard, the two remaining phases of our approach are on-device training and model aggregation, which are detailed below.

5.1 On-Device Learning

At this stage, participant clients have already been selected and modeled as smart devices (smartphones). It is also assumed that the devices are distributed in a coverage area with good computational capacity and communication channel conditions, where each smartphone has enough power capacity to carry out a training process. In federating learning systems, modeling a device with a data collection process is commonly considered. For this, each device already possesses a data batch with data collected over a period of time. Each data batch acts as input for any learning algorithms the device desires to employ.

In this situation, neural networks like RBFN, MLP, and LR have been taken into consideration because the goal of our approach is to solve a problem involving linear regression and improve the results' interpretability. Once the training process ends, the output for each algorithm is a learning model, which has its own structure. Finally, the validation step involves taking the test data and calculating its performance in terms of the coefficient of determination. In Algorithm 1, the main steps of the on-device learning stage are presented.

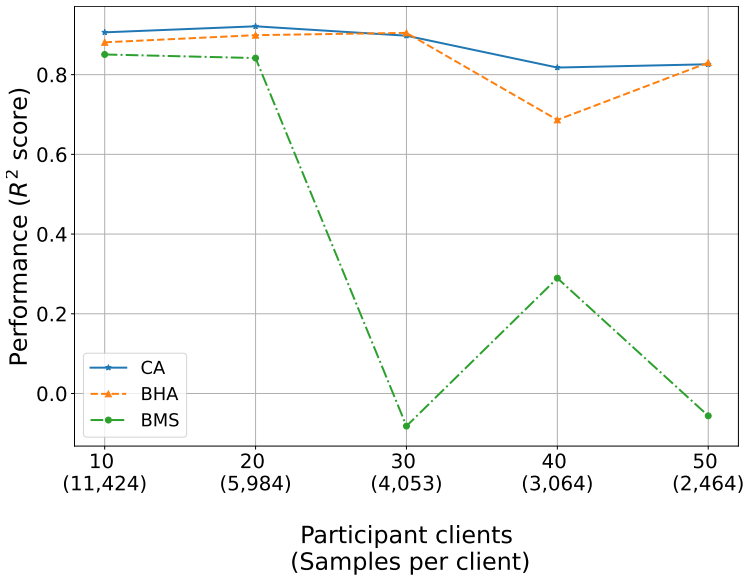


Fig. 4. RBFN performance varying the number of clients from 10 to 50.

5.2 Model Aggregation

On the other hand, Algorithm 2 shows the steps that make up our approach running on the central entity side. Initially, the central entity is modeled as a cloud server with computational and storage capacity because it is assumed that our federated learning system will be based in the cloud. The main task of the cloud server is receiving the local models from participating clients. The models are combined to generate a new generalized model using an aggregation method after all local models have been transmitted to the cloud server.

Complete averaging, best half averaging, and best model selection are regarded as aggregation methods in this work since they tend to be excellent representations of aggregation methods in federated learning. Once the unification process is complete, each algorithm j produces a global learning model that keeps the original local models' structural integrity. A validation procedure is then used to assess the new model's performance and determine whether the coefficient of determination has improved. All these steps are repeated k learning rounds.

6 Experiments

In this section, the effectiveness of the suggested federated learning system is determined using in-depth simulations. To demonstrate that model averaging isn't the best option in federated learning methods, the simulation settings will be shown first, followed by the simulation results.

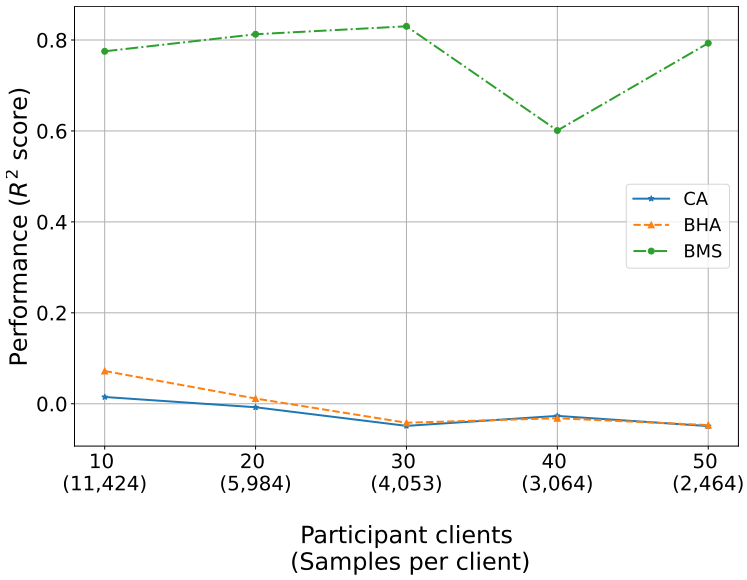


Fig. 5. MLP performance varying the number of clients from 10 to 50.

6.1 Simulation Setup

We simulate a common cloud-based federated learning system with smart devices dispersed throughout a coverage region. We chose a homogenous setting for this simulation where each device has a data batch with the same number of samples, creating the same conditions for each device to perform the training process. Additionally, we looked at a linear regression problem where the effectiveness of the aggregation process is assessed using complete averaging, best half averaging, and best model selection techniques.

We also contrast these aggregating methods across three distinct neural network designs. The first two relate to methodologies frequently utilized in error-based learning, while the third strategy makes use of memory-based learning. Table 3 shows the parameters used for the system configuration, including the neural network parameters. Specifically, the following provides information about the examined algorithms:

– Multilayer Perceptron (MLP). It is a versatile algorithm that can rupture complex relationships in data due to its multiple hidden layers and nonlinear activation functions.

– Linear Regression (LR). It is straightforward to interpret, making it a great choice for cases where the relationship between variables seems linear.

– Radial Basis Function Network (RBFN). It focuses on localized patterns within the data using radial basis functions, which allows it to capture different behaviors in different regions of the input space.

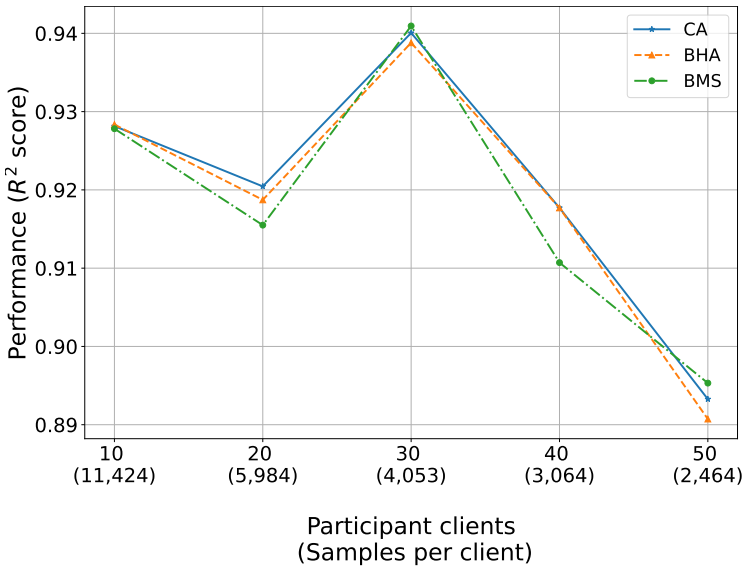


Fig. 6. LR performance varying the number of clients from 10 to 50.

6.2 Metrics

For each learning round in the system, a set of clients transmits local models to a central entity, which is in charge of the aggregation process. However, for each generated model, the following performance metrics are obtained:

– Coefficient of Determination (R^2 Score). It represents the proportion of variance in the dependent variable that is predictable from the independent variables in the model. In other words, it measures the goodness of fit of the model. In addition, it is easy to interpret. A value of 1.0 indicates that the model perfectly predicts the variations in the dependent variable, while a value of 0.0 suggests that the model's predictions do not explain any variability beyond the mean of the dependent variable.

6.3 Performance Analysis

The effectiveness of RBFN in a federated learning system is displayed in Figure 4. It demonstrates a comparison of the three model aggregating methods for five different configurations in more detail. In this experiment, we adjusted both the total number of clients and the average number of samples per client so that, while the total number of clients rose, the average number of samples fell.

Results of the experiment reveal that BHA and CA perform similarly across all client configurations, while BMS performs reasonably well for 10 and 20 clients. It is critical to emphasize that BMS suffers when the number of clients grows since it becomes more challenging to maintain a generalized model with only one client's knowledge. On the other hand, Figure 5 shows the performance of MLP, which suggests that averaging methods from model aggregation, such as CA and BHA, are not the best for this

architecture. The average strategy performs poorly for the configuration of 10 users, with a r^2 score of less than 0.2. The experiment continues with this behavior in the subsequent setups, where it even appears to get worse as the number of clients rises. Instead, the aggregation strategy for the best model selection offers a good performance even when the number of users is increasing.

For all configurations, BMS keeps its r^2 score value above 0.6; the setup with 30 clients is the only one that scores higher than 0.8. The performance of the LR algorithm in various federated learning system setups is shown in Figure 6. It demonstrates that under various configurations of client and sample numbers, the behavior of the three model aggregation strategies is nearly identical. But it is crucial to note that, except in the trial with 30 participant clients, where CA, BHA, and BMS are all above 0.93 and only the BMS exceeds the 0.94 target, the performance of LR decreases in terms of r^2 score as the number of clients increases.

7 Conclusions

In this paper, we have studied the aggregation process through different techniques in non-common scenarios. Furthermore, we discuss the importance of model aggregation to generate a generalized global model that represents all users involved in collaborative training. Simulation results have shown that the CA technique is not always the most effective way to aggregate learning models, even when it is mostly used in FedAvg algorithm variants.

However, other model averaging techniques such as BHA and BMS should be considered in more federating learning scenarios because, in some cases, they have better performance than the CA technique. In future work, we plan to integrate the selection stage, which is a fundamental part of determining the participant users, and model a communication system that allows us to play scenarios with different deployment approaches such as cloud, edge, and hybrid.

References

1. Arivazhagan, M. G., Aggarwal, V., Singh, A. K., Choudhary, S.: Federated learning with personalization layers (2019) doi: 10.48550/ARXIV.1912.00818
2. Brendan-McMahan, H., Moore, E., Ramage, D., Hampson, S., Aguera-y-Arcas, B.: Communication-efficient learning of deep networks from decentralized data. In: Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, vol. 54 (2016) doi: 10.48550/ARXIV.1602.05629
3. Chai, Z., Fayyaz, H., Fayyaz, Z., Anwar, A., Zhou, Y., Baracaldo, N., Ludwig, H., Cheng, Y.: Towards taming the resource and data heterogeneity in federated learning. In: USENIX Conference on Operational Machine Learning, pp. 19–21 (2019)
4. Chen, M., Poor, H. V., Saad, W., Cui, S.: Convergence time optimization for federated learning over wireless networks. IEEE Transactions on Wireless Communications, vol. 20, no. 4, pp. 2457–2471 (2021) doi: 10.1109/twc.2020.3042530
5. Chen, M., Yang, Z., Saad, W., Yin, C., Poor, H. V., Cui, S.: A joint learning and communications framework for federated learning over wireless networks. IEEE Transactions on Wireless Communications, vol. 20, no. 1, pp. 269–283 (2021) doi: 10.1109/twc.2020.3024629

6. Chen, Y., Qin, X., Wang, J., Yu, C., Gao, W.: Fedhealth: A federated transfer learning framework for wearable healthcare. IEEE Intelligent Systems, vol. 35, no. 4, pp. 83–93 (2020) doi: 10.1109/mis.2020.2988604
7. Ferdowsi, A., Saad, W.: Generative adversarial networks for distributed intrusion detection in the internet of things. In: IEEE Global Communications Conference, pp. 1–6 (2019) doi: 10.1109/globecom38437.2019.9014102
8. Hard, A., Rao, K., Mathews, R., Ramaswamy, S., Beaufays, F., Augenstein, S., Eichner, H., Kiddon, C., Ramage, D.: Federated learning for mobile keyboard prediction. arXiv (2018) doi: 10.48550/ARXIV.1811.03604
9. Haykin, S.: Neural networks: A comprehensive foundation. Prentice Hall PTR (1998)
10. Hiessl, T., Schall, D., Kemnitz, J., Schulte, S.: Industrial federated learning – requirements and system design. In: Highlights in Practical Applications of Agents, Multi-Agent Systems, and Trust-worthiness. The PAAMS Collection, vol. 1233, pp. 42–53 (2020) doi: 10.1007/978-3-030-51999-5_4
11. Jansson, M., Axelsson, M.: Federated learning used to detect credit card fraud. Master's Thesis, Department of Computer Science, LUND University (2020)
12. Jiang, J., Ji, S., Long, G.: Decentralized knowledge acquisition for mobile internet applications. World Wide Web, vol. 23, no. 5, pp. 2653–2669 (2020) doi: 10.1007/s11280-019-00775-w
13. Kaissis, G. A., Makowski, M. R., Rückert, D., Braren, R. F.: Secure, privacy-preserving and federated machine learning in medical imaging. Nature Machine Intelligence, vol. 2, no. 6, pp. 305–311 (2020) doi: 10.1038/s42256-020-0186-1
14. Karimireddy, S. P., Kale, S., Mohri, M., Reddi, S., Stich, S., Suresh, A. T.: Scaffold: Stochastic controlled averaging for federated learning. In: Proceedings of the 37th International Conference on Machine Learning, vol. 119, pp. 5132–5143 (2020)
15. Kawa, D., Punyani, S., Nayak, P., Karkera, A., Jyotinagar, V.: Credit risk assessment from combined bank records using federated learning. International Research Journal of Engineering and Technology, vol. 6, no. 4, pp. 1355–1358 (2019)
16. Kriesel, D.: A brief introduction to neural networks (2007) www.dkriesel.com
17. Li, D., Wang, J.: FedMD: heterogenous federated learning via model distillation. In: Proceedings of the 33rd Conference on Neural Information Processing Systems (2019) doi: 10.48550/ARXIV.1910.03581
18. Li, T., Sahu, A. K., Zaheer, M., Sanjabi, M., Talwalkar, A., Smith, V.: Federated optimization in heterogeneous networks. In: Proceedings of the 1st Adaptive and Multitask Learning Workshop (2018) doi: 10.48550/ARXIV.1812.06127
19. Liu, Y., Yu, J. J. Q., Kang, J., Niyato, D., Zhang, S.: Privacy-preserving traffic flow prediction: a federated learning approach. IEEE Internet of Things Journal, vol. 7, no. 8, pp. 7751–7763 (2020) doi: 10.1109/jiot.2020.2991401
20. Liu, Y., Zhang, X., Kang, Y., Li, L., Chen, T., Hong, M., Yang, Q.: Fedbcd: a communication-efficient collaborative learning framework for distributed features. IEEE Transactions on Signal Processing, vol. 70, pp. 4277–4290 (2022) doi: 10.1109/tsp.2022.3198176
21. Mahedi-Hasan, H. M.: Analysis of model aggregation techniques in federated learning. Master's Thesis, Faculty of Graduate Studies and Research, University of Regina (2021)
22. Nishio, T., Yonetani, R.: Client selection for federated learning with heterogeneous resources in mobile edge. In: IEEE International Conference on Communications, pp. 1–7 (2019) doi: 10.1109/icc.2019.8761315
23. Reisizadeh, A., Mokhtari, A., Hassani, H., Jadbabaie, A., Pedarsani, R.: FedPAQ: A communication-efficient federated learning method with periodic averaging and quantization. Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics (2020) doi: 10.48550/ARXIV.1909.13014

24. Sadilek, A., Liu, L., Nguyen, D., Kamruzzaman, M., Serghiou, S., Rader, B., Ingerman, A., Mellem, S., Kairouz, P., Nsoesie, E. O., MacFarlane, J., Vullikanti, A., Marathe, M., Eastham, P., Brownstein, J. S., Aguera-y-Arcas, B., Howell, M. D., Hernandez, J.: Privacy-first health research with federated learning. npj Digital Medicine, vol. 4, no. 1 (2021) doi: 10.1038/s41746-021-00489-2
25. Wang, H., Yurochkin, M., Sun, Y., Papailiopoulos, D., Khazaeni, Y.: Federated learning with matched averaging. In: Proceedings of the 8th International Conference on Learning Representations (2020) doi: 10.48550/ARXIV.2002.06440
26. Wu, Q., Chen, X., Zhou, Z., Zhang, J.: Fedhome: cloud-edge based personalized federated learning for in-home health monitoring. IEEE Transactions on Mobile Computing, vol. 21, no. 8, pp. 2818–2832 (2022) doi: 10.1109/tmc.2020.3045266
27. Yang, Q., Liu, Y., Chen, T., Tong, Y.: Federated machine learning: concept and applications. ACM Transactions on Intelligent Systems and Technology, vol. 10, no. 2, pp. 1–19 (2019) doi: 10.1145/3298981
28. Zhu, X., Wang, J., Hong, Z., Xia, T., Xiao, J.: Federated learning of unsegmented chinese text recognition model. In: Proceedings of the IEEE 31st International Conference on Tools with Artificial Intelligence, pp. 1341–1345 (2019) doi: 10.1109/ictai.2019.00186