Advances in Science and Technology Research Journal

Advances in Science and Technology Research Journal, 2026, 20(3), 47–59
https://doi.org/10.12913/22998624/213204
ISSN 2299-8624, License CC-BY 4.0

Received: 2025.08.27
Accepted: 2025.11.22
Published: 2026.02.01

An entropy-based framework for model aggregation in federated learning

Bartosz Sterniczuk[1*]

[1] Department of Computer Science, Lublin University of Technology, Nadbystrzycka 36B, 20-618 Lublin, Poland
E-mail: b.sterniczuk@pollub.pl

ABSTRACT

Federated learning is a machine learning technique that enables models to learn while preserving user privacy. In this approach, multiple institutions collaborate to develop a shared model without exchanging raw data. Instead, they share only the model's generated weights. In this article, a novel method for weight aggregation is proposed, based on weighted averages and entropy, within the framework of horizontal federated learning. The aggregation process begins by generating predictions on a validation set. Then, entropy is calculated for the weights from each client, reflecting the uncertainty or variability in their contributions. Finally, a weighted average is applied, and the previously computed entropies are used to determine the influence of each client's weights in the final model. The proposed algorithm has been evaluated on several datasets and compared against widely used methods such as FedAvg, FedProx, and FedOpt. The results indicate that the new approach increased mean accuracy by about 2 percentage points compared to FedAvg. The most significant improvement was observed on the Iris dataset, where accuracy increased by about 6 percentage points.

Keywords: classification, entropy, federated learning, federated averaging, machine learning, neural network.

INTRODUCTION

Nowadays, privacy has become of significant importance to our society. With the growth of digital services, both industry and science have had to find a balance between processing and protecting data. Many countries have introduced specific data protection regulations such as the General Data Protection Regulation (GDPR) in the European Union [1], or the Personal Information Protection and Electronic Documents Act (PIPEDA) in Canada [2]. These regulations aim to safe-guard personal data, while still allowing for innovation in areas such as artificial intelligence (AI) and machine learning.

One example of the challenges posed by AI and privacy is the increasing accuracy in recognising handwritten characters. Recent research shows that the proposed models can correctly classify writing with accuracy as high as 96% [3]. These systems open up possibilities for applications, such as digital archives, but on the other hand, they pose a privacy risk, particularly in behaviour identification. Such use raises some concerns, especially when happening beyond legislative control or when their results fall into the wrong hands.

Many devices that use artificial intelligence work in a client-server architecture, which poses a risk of losing control over data. The samples have to be sent to the server for analysis. In many companies, transferring data outside the premises is prohibited and may violate regulations. This is especially true when handling sensitive data, such as medical, financial or tax information [4]. At the same time, a single device does not have enough computing power to make a prediction or build a new model. Furthermore, single clients typically have a limited dataset of cases registered in their own companies. Building one large, shared model seems to be more effective than building models by each company individually. To achieve this while maintaining data privacy, federated learning offers a promising solution [5]. However, the method cited above presents several challenges related to aggregating weights and ensuring the accuracy of trained models. One of the main issues with weight aggregation is the inherent diversity in the data distribution across different clients. Each client possesses a local dataset which can vary in the number of samples, size, quality and representativeness. These factors indicate a risk that the final model might be skewed, biased or inaccurate when applied to real-world data [6].

Nowadays, aggregation methods are one of the most popular research topics. One example is using the F1-Score metrics in ensemble methods to aggregate different models [7]. Similarly, in the field of federated learning, there is a lot of research focused on the various methods for weights aggregation. One of them is Federated Averaging, which includes several samples of each client. Consequently, expanding the client's training datasets enhances their involvement with the training global model. However, this process might skew the global model, which leads to a failure of achieving established accuracy [8]. The case described above prompts the pursuit of new solutions. Thus, the primary objective of this article was to propose a novel approach to weights aggregation that leverages entropy

This paper is organised as follows: firstly, the state-of-the-art works are discussed. Section 3 describes the material used during research as well as shows the basic concepts and types of federated learning. Further, the proposed algorithm is described. Subsequently, Section 4 introduces the course of the experiment, showing the breakdown of the dataset, the methods used and the equipment. In Section 5, the obtained results are presented. Finally, Section 6 contains a discussion of the results achieved. The last paragraph includes a summary of the research.

RELATED WORKS

Although federated learning is a relatively new approach to machine learning, there is already extensive literature on the subject. This section provides a state-of-the-art review of publications related to the research focus of this article.

One of the interesting uses of federated learning is shown in the article [9] where the authors aimed to improve the accuracy of predictions for the mobile keyboard used in the Android operating system (Gboard). This approach was an ideal solution, as the data entered by the user might be highly confidential, such as credit card credentials or a PIN for a mobile bank applications. During their research, they used federated learning to calculate the local model on the client's devices. Additionally, the authors have taken an interesting approach as training the models was possible only when the device was plugged into the charger and at night. The above method was intended to minimise the load on the device during normal operation. As a measure of evaluation, the authors chose click-through rate (CTR), which measures the ratio of clicks to impressions. Of the three proposed learning rates, the best one achieved a CTR increase of about 42% for the training set, while for the test set it was about 34%.

In article [10] by Reyes et al. a novel method for aggregating weights was proposed based on precision. The primary aim of this method is to modify the weighting coefficient. Instead of relying on the number of samples, the authors utilised the inverse of the variance. This approach ensures that the clients with less dispersion have a greater influence on the global weights. To determine the variance, the authors employed an estimate of the raw second moment (uncentred variance) derived from the Adam optimiser, which approximates the diagonal of the Fisher information matrix. Their experiments demonstrated that the algorithm achieves a 9% improvement in prediction accuracy with MNIST, an 18% improvement with Fashion-MNIST, and a 5% improvement with CIFAR-10 in non-IID settings.

Another intriguing approach was presented in article [11] by Ling et al., who introduce a new type of aggregation based on entropy. The authors argued that not all models should contribute to the creation of a common model. They proposed an algorithm that categorises models into two distinct groups: positive and negative devices. This process occurs in two steps. The first one involves collecting soft labels from clients, which indicate the probability of belonging to a particular class. Next, the server calculates the entropy based on these values to assess the diversity of the data and determine whether to reject a given client. In the second phase, the entropy for the global model is calculated. One client is temporarily removed, and the entropy is recalculated. If the new value is higher than the previous one, the client is classified into the positive set; otherwise, in the negative one. As a result, weights are aggregated only from the positive set.

In article [12] the authors proposed a novel aggregation method that analyses the deviation of client parameters from the Gaussian distribution. To achieve this, they utilised statistical measures, such as divergence and higher-order statistics. Divergence, including its Kullback-Leibler divergence and Renyi varieties, is employed to assess how different the client weight parameters are from the Gaussian distribution. The aggregation of weights involves assigning weights to the parameters of the client loss, allowing for their effective combination in the global model. Finally, the weights are normalised, and the server parameters are calculated from a weighted average.

The Choquet integral is one of the most state-of-the-art approaches used in the fuzzy set classification. The authors of article [13] applied it to aggregate weights coming from different models. Their investigation focuses on two datasets: balanced and unbalanced. The Choquet integral was compared to traditional methods, such as the Sugeno integral and arithmetic averaging. The results indicate that the Choquet integral raises the accuracy, precision, and sensitivity, especially with unbalanced data. Choquet's integral allows inputs from various clients, which will be combined efficiently. This helps in dealing better with uncertainties and imperfect relations within the data.

MATERIAL AND METHODS

This chapter describes the datasets used in evaluating the newly proposed method. Additionally, preliminaries on federated learning and its types were also introduced. Finally, a detailed description of the proposed method was presented.

Study material

The new method was tested using four publicly available datasets. The basic characteristics of these datasets are presented in Table 1.

The Heart Disease database contains medical features such as resting blood pressure, maximum heart rate achieved, and serum cholesterol. The dataset is classified as occurrence or absence of disease. It is also worth emphasising that there are missing values, which were removed before starting the tests [14].

The Iris dataset is one of the most popular sets on the issue of machine learning. Originally, it was introduced by the British statistician and biologist Fisher in 1936 [18]. During this research, a set of 150 samples with 4 features and no missing values were used. The dataset consists of three classes –"Iris setosa", "Iris versicolor", and "Iris virginica" [15].

The next dataset under discussion is Pumpkin, which describes 12 morphological features of pumpkin seeds: area, perimeter, major axis length, minor axis length, convex area, equivalent diameter, eccentricity, solidity, extent, roundness, aspect ratio and compactness. The dataset classifies the seeds into two classes: "Ürgüp Sivrisi" and "Çerçevelik". This dataset is one of the largest tested in this article, containing as many as 2500 samples [19].

The last dataset utilised during experiments was Seeds, which included three types of wheat: Kama, Rosa and Canadian. This dataset includes 7 features which describe the characteristics of the seeds, such as size, weight and texture [17].

The concept of federated learning

Federated learning is one of the latest methods in machine learning. This approach assumes a decentralised approach to training models which enables the creation of a single, global model based on the data dispersed between various clients. Unlike traditional machine learning methods in which data samples are sent to central servers, in federated learning data remain on edge devices, which significantly enhances privacy and security.

Formally, it can be assumed that there are n clients: $\{k_1, \dots, k_n\}$ where $n \in N$, each with their datasets $\{d_1, \dots, d_n\}$. It is important to note that it is impossible for clients to share samples with each other, ensuring that the privacy and security

Table 1. Summary of datasets used during experiment

Name	Sample	Feature	Class
Heart disease [14]	1025	13	2
Iris [15]	150	4	3
Pumpkin [16]	2500	12	2
Seeds [17]	210	7	3

of each client's data are maintained throughout the learning process. Initially, the server initialises its own weights $\hat{w} = \{w_1, \dots, w_r\}$, where the r is the number of model parameters. These weights can either be randomly set or derived from a previously trained model, which may have been developed in earlier rounds of federated learning or trained on a different, related dataset. Alternatively, the server may request one of the clients to generate these initial weights based on its local data, establishing a starting point for the training process. A typical process includes the following steps in one round:

1) The server sends weights $\hat{\boldsymbol{w}}$ to each client.
2) Client k_i, where $i = 1, \dots, n$, trains the model on its own dataset d_i using the weights $w_i = \{w_{i,1}, \dots, w_{i,r}\}$ provided by the server. After the training is done, the model generates the established weights $\mathbf{w}_i$ and sends them back to the server.
3) Next, the server aggregates the weights received from the clients w_i for i from 1 to n, into a single global weight $\hat{\boldsymbol{w}}$. There are different types of aggregation. The basic methods consist of simple averaging, while more advanced methods enable the selection of clients with the highest accuracy [20].

This process has been illustrated and presented in Figure 1.

Types of federated learning

During the development of the shared model, a vast number of clients actively collaborate, training their individual models to significantly enhance the overall performance. However, due to the broad diversity of clients and the heterogeneity of their data, challenges can arise during the aggregation of weights, potentially affecting the convergence and stability of the model. In light of these factors, federated learning has been categorised into the following types [21]:

- horizontal federated learning,
- vertical federated learning,
- federated transfer learning.

Horizontal federated learning

Horizontal federated learning is applied when the majority of a client's feature space overlaps with that of another client. It is important to note that, in order to achieve model convergence, the sample space should be distinct. The typical data preparation process involves reconciling which features should be taken into account, while others should be discarded. This approach is commonly used when there are multiple clients from the same business, but each has different customers. By adhering to the principles of privacy, they can collaboratively develop a shared model, which yields mutual benefits [20–22]. The process is illustrated in Figure 2.

Formally, let X represent the feature space, Y represent the label space and I are the training cases. For a client i, let their data d_i be represented by $X_i \subseteq X$, $Y_i \subseteq Y$ and $I_i \subseteq I$. Then for any two clients d_i, d_j where $i \neq j$, it is holds that [22]:

$$X_i = X_j,\ Y_i = Y_j, I_i \neq I_j \tag{1}$$

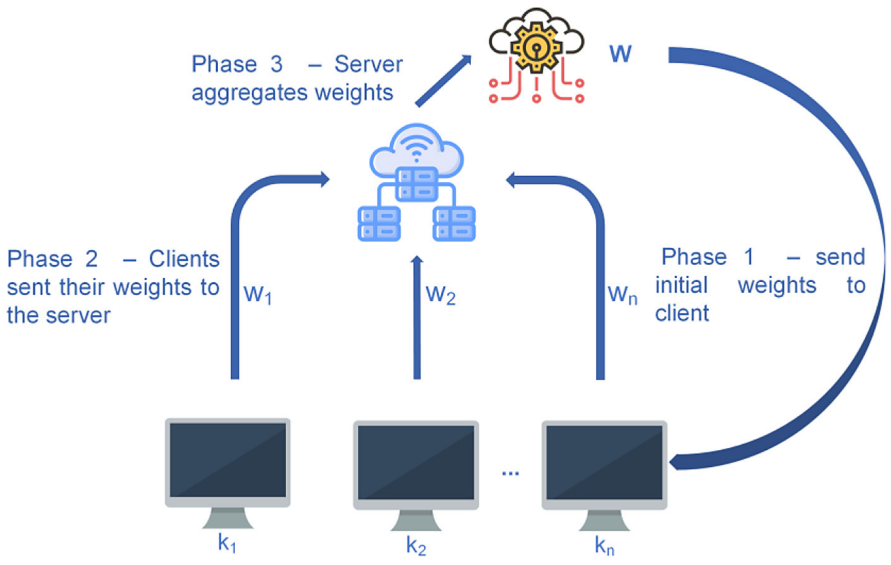
Phase 3 – Server aggregates weights
W
Phase 2 – Clients sent their weights to the server
W1
W2
Wn
Phase 1 – send initial weights to client
...
k1
k2
kn

Figure 1. The basics of federated learning

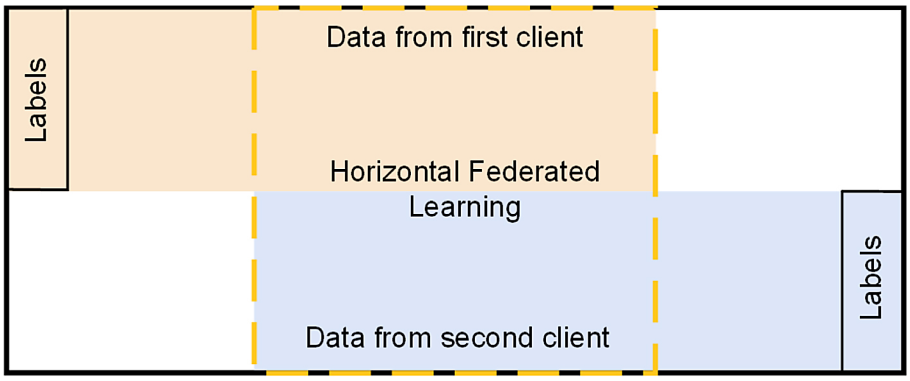

Figure 2. Horizontal learning example [23]

Vertical federated learning

This type of learning is the opposite of horizontal learning. The core idea is that while the feature space differs between clients, the identifier space remains the same. It is particularly useful when multiple clients come from different industries. They share the same customers, but the features they collect are entirely different [21, 22]. This relationship can be summarised mathematically as follows:

$$X_i \neq X_j, Y_i \neq Y_j, I_i = I_j \tag{2}$$

For example, consider specialist clinics that treat the same patients but focus on different medical conditions. Their feature space can vary significantly due to the specific procedures used for various diseases, even though they are treating the same individual. In some cases, diagnosing a condition may require a broader diagnostic perspective, which could be available at another clinic, but cannot be shared due to the sensitivity of medical data. Federated learning solves this issue by allowing the creation of a shared global model, without compromising data privacy [24]. The process outlined is illustrated in Figure 3.

Federated transfer learning

Federated transfer learning introduces greater complexity compared to other learning paradigms. In this approach, feature spaces are often distinct, though there may be slight overlap in some cases. Additionally, the sample spaces usually differ as well. The core idea is to facilitate knowledge transfer between different devices. Collaborating clients generally come from specialised fields, but at a broader level, their datasets share certain commonalities. The typical process involves training a model with one client and then applying it to another, even with minimal data overlap. The second client adapts the feature space to fit their dataset and refines the model by leveraging knowledge transfer [22, 25]. This process can be represented mathematically as follows:

$$X_i \neq X_j, Y_i \neq Y_j, I_i \neq I_j \tag{3}$$

A typical example of the described process could involve a financial institution and an ecommerce company working together. Both the bank and the online store share one common feature: the purchase transaction. However, their feature spaces are significantly different. The store processes information about the products bought, quantities, and customer behaviour on the website. In contrast, the bank records the transaction details and has access to the client's past financial history. Using a model developed by the store, the bank can refine its own model to predict the future spending patterns and make decisions about granting loans to clients [26]. The process depicted is shown in Figure 4.

Model aggregation techniques

Aggregation is one of the most crucial elements in the federated learning algorithm. The main idea behind this technique is to combine local weights trained on local data into one global model. In the literature, many algorithms differ in complexity, as well as in how they deal with the problems arising from data heterogeneity or device diversity. In this section, some of the most popular methods were presented.

FedAvg is one of the earliest weight aggregation algorithms, introduced by a team of researchers from Google. Its main concept revolves around weighting the client's weights and averaging them, where the weight is determined by the

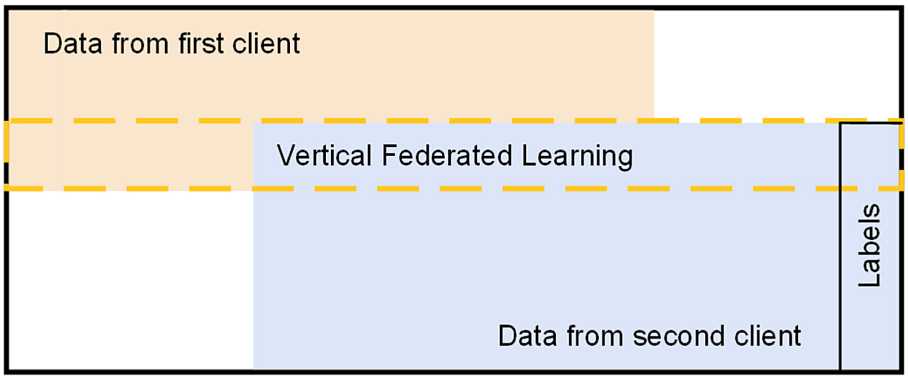

Figure 3. Vertical learning example [23]

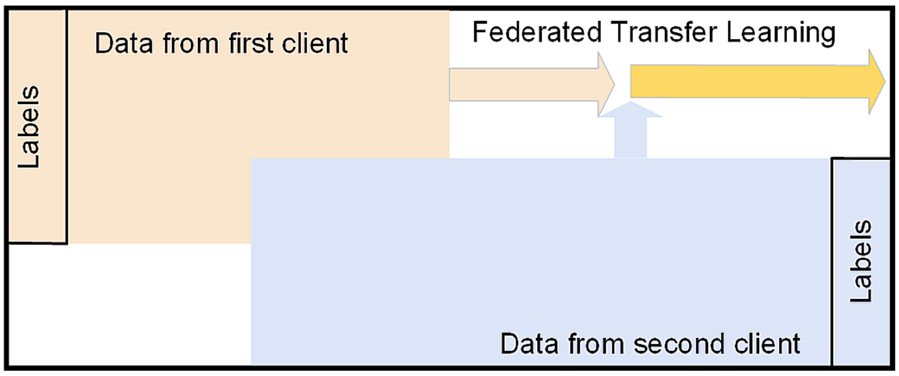

Figure 4. Federated transfer learning example [24]

number of samples from each client. In the *t*-th round, the weight is determined as follows:

$$\widehat{\mathbf{w}}^{(\mathbf{t+1})} = \sum_{i=1}^{n} \frac{m_i}{m} \mathbf{w}_{\mathbf{i}}^{(\mathbf{t})} \tag{4}$$

where: m_i represents the number of samples for the *i*-th client and *m* is the total number of samples. The resulting value from Equation 4 will be used to initialise the client's weights in the next round or, if training has concluded, will serve as the final global weights [27].

Scaffold is a more sophisticated algorithm than FedAvg and tries to eliminate a problem with heterogeneous data by introducing two control parameters ***c*** for a server, and $\boldsymbol{c}_i$ for each model. In each interaction, a client obtains a parameter ***c*** and estimates their weights according to the following formula:

$$\mathbf{w_i} := \mathbf{w_i} - \eta \begin{pmatrix} g_i(\boldsymbol{w_i}) + \\ + \boldsymbol{c} - \boldsymbol{c_i} \end{pmatrix} \tag{5}$$

where η is a learning rate and g_i is the gradient. Further, a client updates their local parameter c_i, where *K* is the number of local update steps:

$$\boldsymbol{c}_i^+ := \boldsymbol{c_i} - \boldsymbol{c} + \frac{1}{K\eta}(\widehat{\boldsymbol{w}} - \boldsymbol{w_i}) \tag{6}$$

In the end, the server aggregates the weights, and updates the *c* parameter [28]:

$$\begin{gathered} \widehat{\mathbf{w}} := \widehat{\mathbf{w}} + \frac{\eta}{m} \sum_{i=1}^{n} (\mathbf{w_i} - \widehat{\mathbf{w}}) \\ \mathbf{c} := \mathbf{c} + \frac{1}{n} \sum_{i=1}^{n} (\mathbf{c}_\mathbf{i}^+ - \mathbf{c_i}) \end{gathered} \tag{7}$$

FedProx is a novel technique proposed by Li et al. Their research showed that local training focuses on its own data bypassing weights received from global models, especially in the models using stochastic gradient descent (SGD). As a result, each model tries to achieve a local minimum rather than the global one in the shared model. To eliminate this discrepancy, they introduced a proximal term, which is included in the objective function and it might be expressed as follows:

$$\frac{\mu}{2}|\mathbf{w}_i^t - \hat{\mathbf{w}}^t|^2 \quad (8)$$

where $\mu \geq 0$ is known as the penalty coefficient of the proximal term. It is also worth noting that if the μ is set to 0, this method is exactly the same as FedAvg. Conversely, the larger the value of μ, the stronger the regularisation, resulting in smaller deviations of local models from the global model. On the other hand, smaller values of μ ead to weaker regularisation and greater flexibility for local models to adapt to their specific data [29].

Proposed method

In this chapter, the proposed novel algorithm was introduced. Traditional aggregation methods often overlook client quality, focusing solely on model validity based on the volume of training samples which assumes that the models trained on larger datasets produce more accurate predictions. To address these limitations, the proposed algorithm incorporates entropy as a measure of predictive uncertainty drawn from information theory. Entropy is a term introduced by Shannon and it is a certain set of uncertainties related to the set of possible outcomes [30].

In practice, the models with lower entropy on the validation set will have a higher inverse entropy value, thereby receiving greater weight in the aggregation process. Conversely, the models with higher entropy, indicating lower confidence in their predictions, will contribute less to the final ensemble model. This approach helps prevent weaker local models from diminishing the overall quality of the global model.

Algorithm 1 presents a comprehensive overview of the proposed algorithm, specifically focusing on a single round of weight aggregation. The key distinction from other methods is the requirement for a validation dataset to be maintained on the server side. This dataset should not be sourced from clients due to privacy concerns, nor should it duplicate any learning events from individual clients. By implementing this approach, the server can evaluate each client effectively and assign an appropriate coefficient based on the value each client contributes to the core model.

Algorithm 1: W*eighted average method based on entropy*
Description:
$\hat{\boldsymbol{w}}$ – a set of global weights
$\mathbf{w}_i$ – a set of weights for the client k_i
V – a validation set
$P_i = \{p_{ij}\}$ – the set of probabilities generated by the client k_i on the validation set
c – number of classes

Outcome: aggregated weights
1. Collect model weights for each customer $\mathbf{w}_i$ for $i = 1,2, ..., n$.
2. For each w_i from $i = 1,2, ..., n$ create a new model M_i and set its weight to w_i.
3. For each model M_i conduct training on the validation set V: $P_i = M_i(V)$.
4. For each set of probabilities compute the entropy: $H_i = \sum_{j=1}^{c} p_{ij}\, log_2\, p_{ij}$.
5. Calculate the inverse of entropy for each model: $E_i = \frac{1}{H_i}$.
6. Aggregate weights: $\sum_{i=1}^{n} w_i\, E_i / \sum_{i=1}^{n} E_i$.

Return the global weights: $\hat{\boldsymbol{w}}$

EXPERIMENTS

Federated learning can be applied to a variety of models, but during this study, it will be based on a neural network classification mechanism. The architecture of the model was selected according to the characteristics of the input data - such as the number of features, dataset size, and overall task complexity – as well as the evaluation of training accuracy and validation learning curves. Using this procedure allowed for the generalization ability to be evaluated alongside recognition of overfitting, both of which played important roles in the architecture design for both datasets.

In the federated learning approach, a key challenge arises in generating learning curves due to the decentralized nature of the data. Since the training data remains on client devices, conventional real-time monitoring of performance metrics is limited. Nevertheless, there is a method to collect training statistics locally and aggregate them centrally, allowing for the generation of representative learning curves. This process is outlined as follows:
1. Collect the final training loss and accuracy from each client at the end of their last local epoch.

2. Send these metrics to the central server.
3. On the server, compute the average training loss and accuracy across all clients – this gives the overall training performance.
4. Run a validation step on the server using a separate validation dataset to obtain validation loss and accuracy.
5. Use the aggregated training and validation results to plot learning curves.

When fine-tuning the hyperparameters, the standard FedAvg aggregation method was used to ground the convergence of the model. Once convergence was achieved, further experiments were performed by applying alternative aggregation methods. The architectures of the models used for each dataset are detailed in Tables 2 to 5.

The study that formulated horizontal federated learning consisted of a certain central server and three customers. The dataset was cut into training, validation and testing subsets. In addition, the training set was already evenly distributed among the three clients to make sure all clients have the same contribution to the model. The hyperparameters for each subset are presented in Table 6.

The new approach is evaluated through a process that starts with training separate models for each client and then constructing a global model through the FedAvg, FedOpt, and FedProx methods. During this step, the model parameters are combined across clients to build a holistic representation of learned patterns. To evaluate the performance of the proposed approach, the same process was followed; this time, however, using a novel weight aggregation method to optimise model performance and ensure balanced contributions from various clients. This comparison allows for assessing the improvements provided by the new aggregation method in comparison to the currently known data aggregation methods, particularly the standard FedAvg algorithm.

Table 2. Summary of neural network architecture for heart disease dataset

Layer	Output shape	Parameters	Activation
Flatten	(,13)	0	-
Dense	(,256)	3 584	Relu
Dropout	(,256)	0	-
Dense	(,128)	32 896	Relu
Dropout	(,128)	0	-
Dense	(,2)	258	Softmax

Table 3. Summary of neural network architecture for Iris dataset

Layer	Output shape	Parameters	Activation
Flatten	(,4)	0	-
Dense	(,32)	160	Relu
Dropout	(,32)	0	-
Dense	(,16)	16	Relu
Dense	(,3)	3	Softmax

Table 4. Summary of neural network architecture for Pumpkin dataset

Layer	Output shape	Param	Activation
Flatten	(,12)	0	-
Dense	(,32)	416	Relu
Dropout	(,32)	0	-
Dense	(,16)	528	Relu
Dense	(,2)	34	Softmax

Table 5. Summary of neural network architecture for Seeds dataset

Layer	Output shape	Param	Activation
Flatten	(,7)	0	-
Dense	(,32)	256	Relu
Dropout	(,32)	0	-
Dense	(,16)	528	Relu
Dense	(,3)	51	Softmax

Table 6. Summary of hyperparameters for each dataset

Dataset	Client epochs	Server epochs	Learning rate
Heart disease	5	20	$2 \cdot 10^{-10}$
Iris	10	10	$2 \cdot 10^{-10}$
Pumpkin	10	10	$2 \cdot 10^{-10}$
Seeds	10	8	$2 \cdot 10^{-10}$

Datasets

Table 7 provides an overview of the datasets used to test the new algorithm and their division into training, validation, and test sets. The first column lists the dataset names while the second column displays the total number of training samples, which were later divided among three clients. Finally, the last two columns show the number of samples in the test and validation sets, respectively. The dataset was split into training, validation, and test sets in a 6:2:2 ratio.

Selecting appropriate datasets ensures an accurate evaluation of the proposed method by including datasets with varying sizes of client training data, with sample sizes ranging from 90 to 1500.

Computing environment

During the research, the Python programming language version 3.10.15 was used, together with one of the most popular libraries targeting federated learning – Flower version 1.10.0 [31], [32]. The process of implementation involved leveraging inheritance to extend this library's "FedAvg" class. Utilising polymorphism, the newly created weight aggregation strategy was used in the server class. Additionally, the following libraries were used:

- Tensorflow-gpu: 2.17.0,
- Keras: 3.5.0,
- Numpy: 1.26.4.

The research was conducted on a computer with the following specifications:

- Operating System: Ubuntu 22.04,
- GPU: Nvidia RTX 5000,
- CPU: Intel(R) Xeon(R) Silver 4208 CPU @ 2.10GHz,
- RAM: 64 GB.

RESULTS

This chapter presents the results obtained from the conducted experiments. The evaluation includes confusion matrices, test set performance metrics, validation and training learning curves, which were used to assess model convergence. These elements provide a comprehensive overview of the model's behaviour, its generalisation capabilities, and the effectiveness of the applied aggregation methods in the federated learning setting.

Figure 5 illustrates the learning curves for both accuracy and loss, tracked throughout the training process. These curves are presented for both the training and validation datasets, allowing for a comprehensive assessment of the model's convergence behaviour. The graphs shown were produced on the server side, following the earlier algorithm using the FedAvg aggregation method.

Table 7. Division of the dataset into training, validation and testing

Dataset	Train	Test	Validation
Heart disease [14]	615	205	205
Iris [15]	90	30	30
Pumpkin [19]	1500	500	500
Seeds [17]	126	42	42

Table 8 provides a comprehensive summary of the test set results for each dataset, comparing the performance of four aggregation methods: FedAvg, Entropy, FedOpt, and FedProx. Each table presents the results for key performance metrics, including accuracy, precision, recall, and F1-score. For each dataset, the first row shows the results for the FedAvg method, while the second, third, and fourth rows display the performance for the Entropy, FedOpt, and FedProx methods, respectively.

In order to give an accurate representation of the behaviour of the different aggregation methods, the confusion matrices for each dataset are presented in Figure 6. Each row contains the aggregation methods in the following order: FedAvg, Entropy, FedOpt, and FedProx.

DISCUSSION

The results presented in Table 8 clearly demonstrate that introducing an aggregation method based on entropy into federated learning brings tangible benefits compared to the classic FedAvg, often outperforms FedOpt and FedProx as well. Although the differences are minimal for the Pumpkin dataset, a noticeable improvement in accuracy is observed across most of metrics for datasets, such as Heart Disease, Iris or Seeds.

Firstly, for the Heart Disease dataset, the entropy-based method increases the classification accuracy of sick patients from 92.66% (FedAvg) to 95.41%, while simultaneously reducing the misclassification rate from 7.34% to 4.59%, which is crucial for patient safety in a medical context. The newly proposed method achieved the best results across all metrics and increased overall accuracy by about 1.46 percentage points compared to the FedAvg method.

In the Iris dataset, the entropy-based method improved the classification accuracy for the second class from 61.54% (FedAvg) to 76.92% (Entropy). A similar result was also achieved by the FedProx method. This method showed the largest increase in accuracy over FedAvg across all datasets, with an improvement of 15.38 percentage points. Among all the aggregation methods, both the entropy-based method and FedProx achieved the highest accuracy values, both reaching exactly 90%.

The results obtained on the Pumpkin dataset show little difference between the aggregation methods. The highest accuracy was achieved by

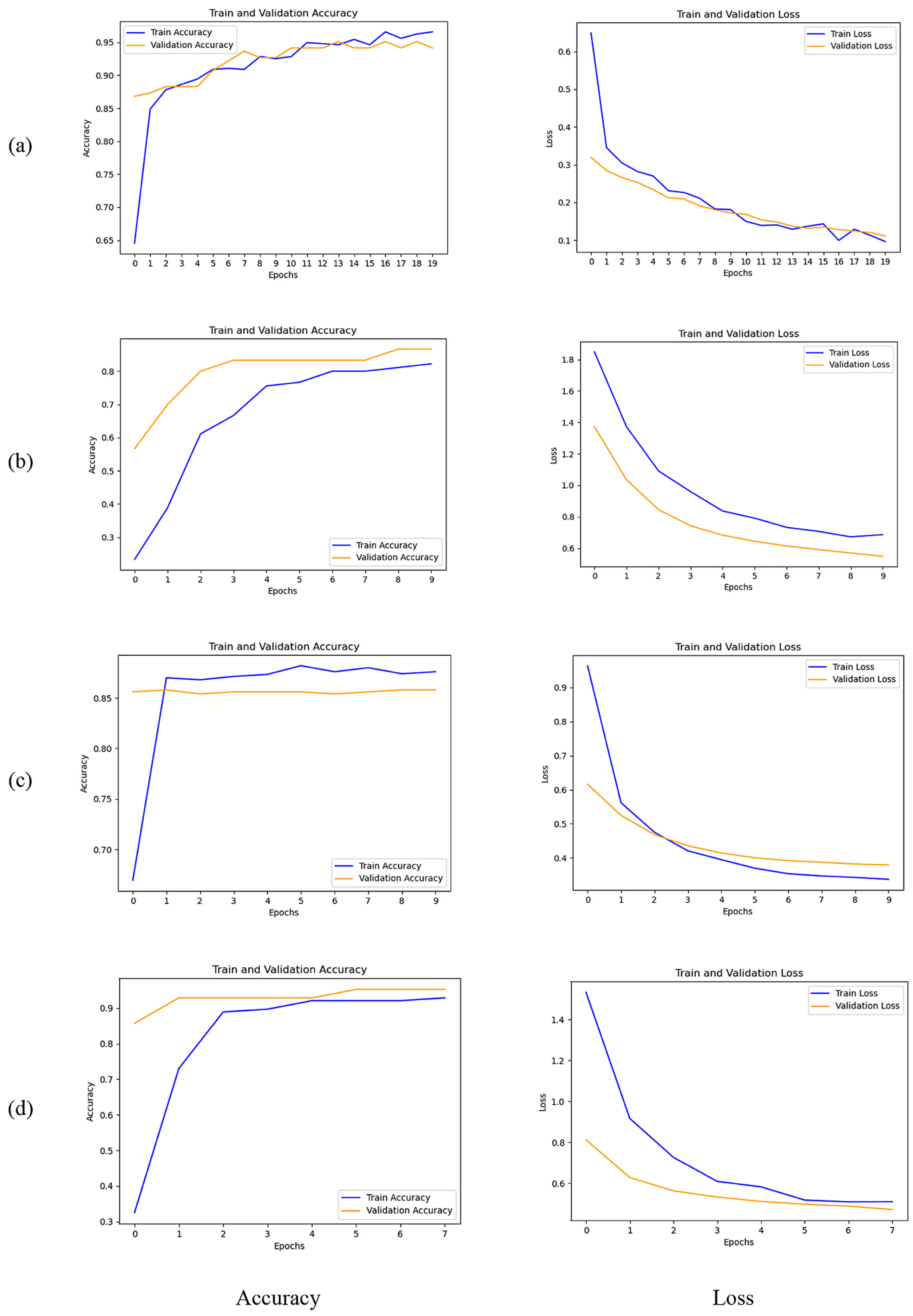

Figure 5. Test set results for each dataset: (a) heart disease, (b) iris, (c) pumpkin, (d) seeds

Entropy and FedProx, with values of 88% and 87.80%, respectively. The confusion matrix indicates that the Entropy method slightly improved the accuracy for class 1, with an increase of 0.4 percentage points.

For the Seeds dataset, the largest overall increase in accuracy was observed compared to FedProx, with an improvement of 2.39 percentage points. A similar increase was also noted when compared to FedAvg. The confusion matrix reveals that the entropy-based method achieves the most significant improvement in class 1, with an increase of approximately 13.34 percentage points compared to FedProx. However, for class

Table 8. Presents the test set results (%) for each dataset across different aggregation methods: FedAvg, Entropy, FedOpt, and FedProx

Dataset	Methods	Accuracy	Precision	Recall	F1-Score
Heart disease	FedAvg	93.66	93.71	93.66	93.66
	Entropy	**95.12**	**95.12**	**95.12**	**95.12**
	FedOpt	93.66	93.67	93.66	93.66
	FedProx	94.15	94.23	94.15	94.15
Iris	FedAvg	83.33	90.91	83.33	83.80
	Entropy	**90.00**	**93.33**	**90.00**	**90.35**
	FedOpt	86.67	92.00	86.67	87.12
	FedProx	90.00	93.33	90.00	90.35
Pumpkin	FedAvg	87.60	87.61	87.60	87.60
	Entropy	**88.00**	**88.00**	**88.00**	**88.00**
	FedOpt	87.80	87.82	87.80	87.80
	FedProx	87.80	87.81	87.80	87.80
Seeds	FedAvg	85.71	86.49	85.71	85.90
	Entropy	**88.10**	**88.54**	**88.10**	**88.24**
	FedOpt	83.33	84.65	83.33	83.20
	FedProx	85.71	86.57	85.71	85.69

Iris

Methods		Setosa	Versicolor	Virginica
FedAvg	Setosa	100.0	0.0	0.0
Entropy		**100.0**	**0.0**	**0.0**
FedOpt		100.0	0.0	0.0
FedProx		100.0	0.0	0.0
FedAvg	Versicolor	0.0	61.54	38.46
Entropy		**0.0**	**76.92**	**23.08**
FedOpt		0.0	69.23	30.77
FedProx		0.0	76.92	23.08
FedAvg	Virginica	0.0	0.0	100.0
Entropy		**0.0**	**0.0**	**100.0**
FedOpt		0.0	0.0	100.0
FedProx		0.0	0.0	100.0

Seeds

Methods		Kama	Rosa	Canadian
FedAvg	Kama	86.67	0.0	13.33
Entropy		**86.67**	**0.0**	**0.0**
FedOpt		66.67	0.0	33.33
FedProx		73.33	0.0	26.67
FedAvg	Rosa	8.33	91.67	0.0
Entropy		**8.33**	**91.67**	**0.0**
FedOpt		8.33	91.67	0.0
FedProx		8.33	91.67	0.0
FedAvg	Canadian	20.0	0.0	80.0
Entropy		**13.33**	**0.0**	**86.67**
FedOpt		6.67	0.0	93.33
FedProx		6.67	0.0	93.33

Heart Disease

Methods		Helathy	Sick
FedAvg	Helathy	94.79	5.21
Entropy		**94.79**	**5.21**
FedOpt		93.75	6.25
FedProx		95.83	4.17
FedAvg	Sick	7.34	92.66
Entropy		**4.59**	**95.41**
FedOpt		6.42	93.58
FedProx		7.34	92.66

Pumpkin

Methods		Çerçevelik	Ürgüp Sivrisi
FedAvg	Çerçevelik	88.40	11.60
Entropy		**88.40**	**11.60**
FedOpt		88.80	11.20
FedProx		88.40	11.60
FedAvg	Ürgüp Sivrisi	13.20	86.80
Entropy		**12.40**	**87.60**
FedOpt		13.20	86.80
FedProx		12.80	87.20

Figure 6. Presents the confusion matrices (%) for each dataset, comparing the classification performance across different aggregation methods: FedAvg, Entropy, FedOpt, and FedProx

3, there is a decrease of 6.66 percentage points. Despite this, the method still demonstrates a notable increase in overall accuracy.

The analysis of the presented results shows that the entropy-based weight aggregation method for federated learning yields measurable benefits compared to classical methods such as FedAvg, and in many cases also outperforms the FedOpt method. In all tested datasets, the newly proposed method achieved the highest scores or consistently high results, regardless of the dataset. The average increase in accuracy over the FedAvg method was 2.73 percentage points.

CONCLUSIONS

This article introduced a new aggregation method based on entropy which was empirically tested on various datasets. The results show that in all tested cases, the proposed method outperforms the classical approach such as FedProx or FedOpt. The present achievement confirms the thesis put forward at the beginning of this article, providing strong evidence that the novel method offers significant benefits and advantages in Federated Learning. The presented algorithm opens a new approach to using entropy in the context of Federated Learning. The author's future work will involve examining the proposed method on other types of data, especially in the datasets containing image datasets.

REFERENCES

1. European union general data protection regulation (GDPR) [Internet]. [cited 2024 Nov 24]. Available from: https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:02016R0679-20160504
2. Personal information protection and electronic documents ac. [Internet]. [cited 2024 Aug 3]. Available from: https://laws-lois.justice.gc.ca/PDF/P-8.6.pdf
3. Lukasik E, Charytanowicz M, Milosz M, Tokovarov M, Kaczorowska M, Czerwinski D, et al. Recognition of handwritten Latin characters with diacritics using CNN. Bulletin of the Polish Academy of Sciences Technical Sciences. 2021 Jan 6;136210–136210.
4. Spiekermann S, Cranor LF. Engineering Privacy. IEEE Transactions on Software Engineering. 2009 January;35(1):67–82.
5. Zhu X, Wang D, Pedrycz W, Li Z. Privacy-Preserving Realization of Fuzzy Clustering and Fuzzy Modeling Through Vertical Federated Learning. IEEE Trans Syst Man Cybern Syst. 2024 February; 54(2):915–24.
6. Qi P, Chiaro D, Guzzo A, Ianni M, Fortino G, Piccialli F. Model aggregation techniques in federated learning: A comprehensive survey. Future Generation Computer Systems. 2024 January;150:272–93.
7. Sterniczuk B, Charytanowicz M. An ensemble transfer learning model for brain tumors classification using convolutional neural networks. Advances in Science and Technology Research Journal. 2024 December 1;18(8):204–16.
8. Huang W, Li T, Wang D, Du S, Zhang J, Huang T. Fairness and accuracy in horizontal federated learning. Inf Sci (N Y). 2022 Apr;589:170–85.
9. Yang T, Andrew G, Eichner H, Sun H, Li W, Kong N, et al. Applied Federated Learning: Improving Google Keyboard Query Suggestions. 2018 December 6.
10. Reyes J, Di Jorio L, Low-Kam C, Kersten-Oertel M. Precision-Weighted Federated Learning. 2021 July 20.
11. Ling Z, Yue Z, Xia J, Hu M, Wang T, Chen M. FedEntropy: Efficient Device Grouping for Federated Learning Using Maximum Entropy Judgment. 2022 May 24.
12. Chatzikonstantinou C, Konstantinidis D, Dimitropoulos K, Daras P. Federated Learning Aggregation based on Weight Distribution Analysis. In: 2023 IEEE International Conference on Imaging Systems and Techniques (IST). IEEE; 2023; 1–6.
13. Pękala B, Wilbik A, Szkoła J, Dyczkowski K, Żywica P. Federated Learning with the Choquet Integral as Aggregation Method. In: 2024 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE). IEEE; 2024; 1–8.
14. Janosi A, Steinbrunn W, Pfisterer M, Detrano R. Heart Disease [Internet]. 1989 [cited 2024 Aug 15]. Available from: https://doi.org/10.24432/C52P4X
15. Fisher R. Iris [Internet]. UCI Machine Learning Repository. 1936 [cited 2024 Aug 20]. Available from: https://doi.org/10.24432/C56C76
16. Koklu M, Sarigil S, Ozbek O. The use of machine learning methods in classification of pumpkin seeds (Cucurbita pepo L.). Genet Resour Crop Evol. 2021 Oct 25;68(7):2713–26.
17. Charytanowicz M, Niewczas J, Kulczycki P, Kowalski P. Seeds [Internet]. UCI Machine Learning. 2010 [cited 2024 Aug 15]. Available from: https://doi.org/10.24432/C5H30K
18. Fisher RA. The use of multiple measurements in taxonomic problems. Ann Eugen. 1936 September 29;7(2):179–88.
19. Pumpkin dataset [Internet]. [cited 2024 Aug 5]. Available from: https://www.muratkoklu.com/datasets/
20. Zhang C, Xie Y, Bai H, Yu B, Li W, Gao Y. A survey on federated learning. Knowl Based Syst. 2021 Mar;216:106775.
21. Li L, Fan Y, Tse M, Lin KY. A review of applications in federated learning. Comput Ind Eng. 2020 November;149:106854.
22. Yang Q, Liu Y, Chen T, Tong Y. Federated Machine Learning. ACM Trans Intell Syst Technol. 2019 March 31;10(2):1–19.
23. Rahman KMJ, Ahmed F, Akhter N, Hasan M, Amin R, Aziz KE, et al. Challenges, applications and design aspects of federated learning: A survey. IEEE Access. 2021;9:124682–700.
24. Dhade P, Shirke P. Federated learning for healthcare: A comprehensive review. In: RAiSE-2023. Basel Switzerland: MDPI; 2024; 230.

25. Saha S, Ahmad T. Federated transfer learning: Concept and applications. Intelligenza Artificiale: The international journal of the AIxIA. 2021 Jul 28;15(1):35–44.
26. Wang Z, Xiao J, Wang L, Yao J. A novel federated learning approach with knowledge transfer for credit scoring. Decis Support Syst. 2024 Feb;177:114084.
27. H. B. McMahan, E. Moore, D. Ramage, S. Hampson, B. A. y Arcas. Communication-efficient learning of deep networks from decentralized data. Artificial intelligence and statistics. 2017;1273–82.
28. Karimireddy SP, Kale S, Mohri M, Reddi SJ, Stich SU, Suresh AT. SCAFFOLD: Stochastic Controlled Averaging for Federated Learning. 2020.
29. Li T, Sahu AK, Zaheer M, Sanjabi M, Talwalkar A, Smith V. Federated Optimization in Heterogeneous Networks. 2018 December 14.
30. Shannon CE. A Mathematical Theory of Communication. Bell System Technical Journal. 1948 Jul;27(3):379–423.
31. Framework used during experiments [Internet]. [cited 2024 August 5]. Available from: https://flower.ai/
32. Beutel DJ, Topal T, Mathur A, Qiu X, Fernandez-Marques J, Gao Y, et al. Flower: A Friendly Federated Learning Research Framework. 2020 July 28.