electronics

MDPI

Article

Genetic Clustered Federated Learning for COVID-19 Detection

Dasaradharami Reddy Kandati and Thippa Reddy Gadekallu *

School of Information Technology and Engineering, Vellore Institute of Technology, Vellore 632014, India
* Correspondence: thippareddy.g@vit.ac.in

Abstract: Coronavirus (COVID-19) has caused a global disaster with adverse effects on global health and the economy. Early detection of COVID-19 symptoms will help to reduce the severity of the disease. As a result, establishing a method for the initial recognition of COVID-19 is much needed. Artificial Intelligence (AI) plays a vital role in detection of COVID-19 cases. In the process of COVID-19 detection, AI requires access to patient personal records which are sensitive. The data shared can pose a threat to the privacy of patients. This necessitates a technique that can accurately detect the COVID-19 patients in a privacy preserving manner. Federated Learning (FL) is a promising solution, which can detect the COVID-19 disease at early stages without compromising the sensitive information of the patients. In this paper, we propose a novel hybrid algorithm named genetic clustered FL (Genetic CFL), that groups edge devices based on the hypertuned parameters and modifies the parameters cluster wise genetically. The experimental results proved that the proposed Genetic CFL approach performed better than conventional AI approaches.

Keywords: COVID-19 detection; artificial intelligence; federated learning; privacy; security

check for updates

Citation: Kandati, D.R.; Gadekallu, T.R. Genetic Clustered Federated Learning for COVID-19 Detection. *Electronics* **2022**, *11*, 2714. https://doi.org/10.3390/electronics11172714

Academic Editor: Kenji Suzuki

Received: 27 July 2022
Accepted: 26 August 2022
Published: 29 August 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

The COVID-19 outbreak disturbed public health and human life [1]. The spread of COVID-19 [2] is still ongoing, and researchers are trying to find effective ways in early detection of the disease. The aim is to identify and isolate affected people, which results in limiting the spread of COVID-19. AI plays a vital role in detection of COVID-19, allowing researchers to identify it by analyzing symptoms such as throat infection, cold sweats, difficulty in breathing, and also with the assistance of a X-ray [3]. AI, with help of historical data related to COVID-19, can help in predicting and providing essential guidelines to control the spread of the COVID-19 pandemic. The historical data used by AI requires patient records, which are confidential. The patients will hesitate to share their sensitive information as their privacy can be compromised. This creates a scarcity of data required for predictions. A robust model cannot be developed due to this challenge [1]. A novel strategy is required, that allows the development of models that can provide accurate predictions without compromising the patients' personal information.

FL was introduced as an innovative ML approach by Google in 2016 [4]. The goal of FL is to create a ML method consisting of multiple datasets without gathering actual data while preserving confidentiality, privacy, transparency, and security [5,6]. In every iteration of the FL process, local system builds a classifier that uses native information and delivers parameters to a global system without transmitting actual data.

FL provides collaborative environment among different healthcare organizations in preparing a COVID-19 prediction framework while maintaining data privacy [7,8]. Researchers used FL to assess COVID-19 disease from computed tomography or X-ray pictures [1,9]. Current FL studies focus on issues related to communication costs and performance issues. In FL, communication costs increase with frequent updates in patient data to the server. FL addresses the privacy and security issues in healthcare sector by allowing data servers to classify their designs locally and distribute each other's models without compromising patient's data privacy [10].

Every iteration of an FL approach consists of client-server communication, native mentoring, and prototype clustering [11–14]. The transmission of model from the server to all the clients and vice versa can cause communication overhead. Each connectivity session involves implementation issues due to poor data usage, network congestion, and ethical concerns. Modified communication algorithms such as privacy-preserving and communication efficient scheme for federated learning (PCFL) [15,16], minimize the model dimensions and improve security with compression and encryption. The number of edge devices also influence the communication load. The implementation of communication sparsification [17] over clients is modeled to increase the convergence rate and reduce network traffic on the server. The hierarchical clustering [18] approach is also used in many models to summarize related customer strategies and minimize clustering difficulty.

In a heterogeneous environment, not only communication but also AI model training is more challenging [19]. Clients train on the server model using hyper-parameters including client ratio (e.g., choosing 100 clients), epochs per round, batch size, learning rate. Edge devices differ in computational power and data properties, making it difficult to integrate broadly developed client models. The optimization methods such as FedMA and FedAvg [20] are more focused on integrating weights of model parameters. Training and aggregation are both affected by integration rate and training intensity. There are many new techniques for model clustering, such as combining new and existing features [21] or identifying the standard client models [22] to improve the classification. Several studies use various global models such as Federated Cloning-and-Deletion (FedCD) to improve convergent analysis [23].

Researchers have mainly focused on model aggregation to make FL concepts adaptable to non-IID user information [24]. The local training model has a significant impact on determining the model's accuracy. In this study, a novel solution based on genetic algorithm is proposed for hyper-parameter tuning for improved model aggregation in a cluster, as illustrated in Figure 1. The proposed genetic CFL model involve the following steps:

- The clients are grouped based on the hyper-parameters thereby increasing the learning efficiency per training unit.
- Genetic algorithm is used to tune the hyper-parameters and better model aggregation in a cluster.

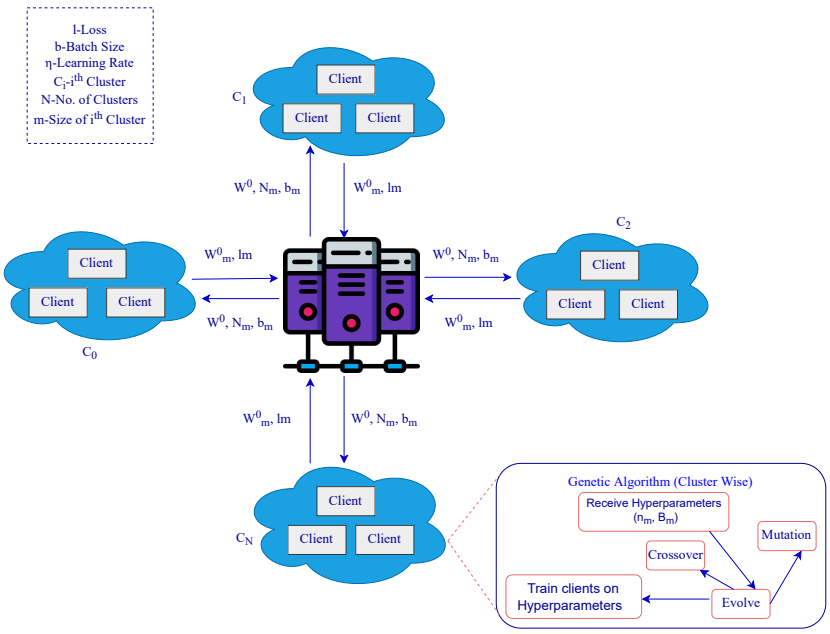

Figure 1. Genetic CFL Architecture.

The genetically optimization FL approach is a novel method for enhancing COVID-19 detection and improve the AI model efficiency and performance. In this study, we create a genetically optimization FL system architecture to detect COVID-19. When compared to basic FL's technique, the proposed technique is more accurate and ensures provacy preservation.

The rest of the article is organized as follows: Section 2 introduces the literature survey. Section 3 provides details of the proposed framework and its methodology. Section 4 discusses the experimental results. The final section of the paper presents conclusion and future research.

2. Literature Survey

This section presents a survey on the current literature on FL, clustering, and evolutionary algorithms, respectively, in order to understand their limitations.

2.1. Federated Learning

Recent studies have focused on FL as a distributed and edge AI architecture [25,26]. In a heterogeneous environment containing non-IID data, FL's decentralized nature directly contradicts traditional AI algorithms which are centralized. Many novel approaches have tried to address the aggregation of non-IID data with various aggregation algorithms, including FedMA [20], feature fusion [21], and grouping of similar client models [22] for better personalized and accurate results. The clustering process makes use of client-model similarity [27] and provides efficient communication to improve data generalization [28].

Clustering will help in optimizing the communication in FL. The convergence of the model may be significantly reduced if there are thousands of nodes in a realistic scenario. Algorithms for partitioning clusters, such as k-means clustering [29], require a predetermined number of clusters which is not feasible. Clusters based on generative adversarial networks and agglomerative hierarchical clustering [18] are examples of non-definitive clusters.

2.2. Evolutionary Algorithms

A model's hyper-parameters selection determine its ability to learn from datasets. Many researchers are working on the optimization of AI models and parameters using evolutionary algorithms [30], such as genetic algorithms [31] and whale optimization [32].

The ensemble models developed using evolutionary algorithms with deep learning techniques have become increasingly popular for optimization tasks [33].

The use of evolutionary algorithms with FL is not yet fully realised. Due to the ambiguity of data, hyper-parameter tuning is even more critical. Optimization algorithms assist with tuning these parameters beyond manual capabilities. Genetic algorithm is used to optimize learning rates and batch sizes for each of the individual end device models. Agrawal et al., in [34] showed that FL is restricted by efficiency of client training, thus involves selecting hyper-parameters effectively, model adjustment, and procedure streamlining. FedTune automatically tunes FL hyper-parameters during model training based on application training preferences [35]. To achieve diverse training preferences, it can be challenging to tune multiple hyper-parameters, particularly when several aspects of the system have to be optimized. FedEx estimates gradients based on client-side hyper-parameter distributions in federated settings [36]. This approach uses weight-sharing methods for searching neural architectures. The training process for FL must not only be aimed at high accuracy but also at reducing the training time and resource consumption in practical environments, using low-capacity computing devices [37,38]. FL uses best epoch algorithm to determine how many epochs are necessary per training round. A summary of the key findings from the above discussion can be found in Table 1.

Table 1. Summary of important surveys on FL and rvolutionary algorithms.

Ref. No	Technologies Used	Key Contributions	Limitations
[20]	Federated matched averaging (FedMA) algorithm	FedMA builds the shared global model layer by layer based on feature extraction signatures of hidden elements.	Privacy, data bias
[21]	Feature fusion mechanism	The accuracy and generalization abilities of FedFusion outperform baselines while reducing communication rounds by more than 60 percent.	The issue of high communication costs must be addressed immediately.
[22]	Iterative Federated Clustering Algorithm (IFCA)	The convergence rate of the population loss function under proper initialization ensures both convergences of the training loss and generalization to test data simultaneously.	Data heterogeneity is to be addressed.
[27]	Multi-center aggregation mechanism	The proposed objective function is optimized using the Federated Stochastic Expectation Maximization method (FeSEM).	Data heterogeneity is to be addressed.
[31]	Genetic algorithms	Convolutional neural networks can be efficiently tuned by using a variable-length genetic algorithm.	In the case of networks with fewer layers, the size could be too small for the problem, resulting in underfitting.
[32]	Whale optimization algorithm (WOA)	For training multilayer perceptrons (MLP), the WOA was applied because of its high local optimization avoidance and fast convergence speed.	Slow convergence speed and local optima stagnation are the main disadvantages of conventional training algorithms.
[39]	Clustered federated learning	It proposes a collaborative learning framework to intelligently process visual data at the edge device by developing a multi-modal ML algorithm that is capable of diagnosing COVID-19 in both X-ray and Ultrasound images.	The major challenge here is regarding the performance of CFL when the number of samples per client varies.
[13]	Clustered federated learning	It addresses the issue of suboptimal results when the local clients' data distributions diverge by separating the client population into different groups based on the pairwise cosine similarities.	The main cluster is separated from some suspicious clients after a few rounds, which poses a major challenge.
This paper	**Genetic CFL algorithm**	**Genetic algorithms are used to optimize the hyper-parameters such as batch size, and learning rate of the clustered FL models.**	**Client training has a significant impact on FL efficiency.**

3. Proposed Methodology

This section describes genetic CFL optimization technique using a comprehensive statistical method. There are two sections in the workflow, the first round of broadcasting is represented by Algorithm 1, which tells about the number of clusters and the federated training which uses genetic algorithm is represented by Algorithm 2. This section mainly describes the differential behavior of the algorithm with various hyper parameters that includes number of iterations, client ratio(n), minimum samples, batch size, and learning rate (η). The following Table 2 highlights most of the symbols used during the algorithm.

Algorithm 1 Clustering and initial broadcasting
N = no. of clients
η: Learning Rate
η_m: Learning Rate List

procedure SERVER
  w^0: Initialization of the server model
  Define η_m based on rates of learning $[1e-1, 1e-5]$
  Broadcasting(w^0, sampling(η_m, 3))
procedure CLIENT
  $i \leftarrow 0$
  while $i \neq N$ **do**
    $j \leftarrow 0$
    while $j \neq 3$ **do**
      Learn w_j^0 on η_j
      $losses \leftarrow \text{loss}(w_j^0)$
    return $w_{min}^0, \eta_{min}, losses_{min}$
procedure SERVER
  $w^0 \leftarrow \frac{\sum(w_n^0)}{n}$

Algorithm 2 FL-based genetic optimization for clustered data.
rounds: The number of loops required to train the decentralized approach

function MUTATE(η)
  variable $\leftarrow$ sampling($[-1, 0, 1]$)
  $\eta \leftarrow \eta + \frac{\eta \times variable}{10}$
  return η
function CROSSOVER(η_N)
  Generate a new array η_{temp} to insert η
  $\eta_{temp}[0, 1] \leftarrow \eta_N[0, 1]$
  for $k \leftarrow 2$ to $size(\eta_N)$ **do**
    $parent_A, parent_B \leftarrow$ sampling$(0, size(\eta_N))$
    $\eta_{temp}[k] \leftarrow$ MUTATE$(\frac{\eta_N[parent_A]+\eta_N[parent_B]}{2})$
  return η_{temp}
function EVOLVE($losses_N, \eta_N$)
  $losses_N$, order $\leftarrow$ sort($losses_N$)
  $\eta_N \leftarrow$ sort(η_N, order)
  return CROSSOVER(η_N)
procedure TRAIN
  len $\leftarrow$ size(clusters)
  ind $\leftarrow$ 0 to len
  Assign η_{global} with structure (len, size(clusters[ind])
  clusters$_{unique}$ = Identical(cluster)
  for $i \leftarrow 0$ to iterations **do**
    for $k \leftarrow 0$ to $dimensions$(clusters) **do**
      ind = [clusters.index(cluster[i])]
      $\eta_{global}[k] \leftarrow$ EVOLVE(losses[ind],$\eta_N[ind]$)
    Losses caused by empty arrays, η_N
    for $k \leftarrow 0$ to N **do**
      $w_k^0[k]$, losses[k], $\eta_N[k]$ = train(w^0, η_{global}[clusters[k][clusters[k].nextIndex])
    $w^0 \leftarrow \frac{\sum(w_m^0)}{N}$

Table 2. Symbol representations.

Symbol	Meaning
N	no. of customers
η	Learning Rate
b	Batch Size
$\overline{C_i}$	ith customer
w^0	Weights of models
w_n^0	Weights of models of nth customer

Dataset Description

The dataset used in this work is taken from the kaggle repository https://www.kaggle.com/datasets/mykeysid10/covid19-dataset-for-year-2020?select=covid_data_2020-2021.csv (accessed on 26 July 2022). There are 10 attributes, among which the attribute Corona result indicates whether a person has a positive or negative Corona result. Table 3 displays an overview of the information of each column that is used in our implementation.

Table 3. Dataset description.

Column Name	Description
test date	Date of arrival of the test in the laboratory. It should be noted that this is the citizen's first test, in the DD / MM / YYYY format.
cough	Did cough symptoms appear before the test? 1—Yes, 0—No, NULL—Unknown.
fever	Did fever appear before the test? 1—Yes, 0—No, NULL—Unknown.
sore throat	Did a sore throat appear before the test? 1—Yes, 0—No, NULL—Unknown.
shortness of breath	Did breathing difficulties occur before the test? 1—Yes, 0—No, NULL—Unknown.
headache	Did a headache appear before the test? 1—Yes, 0—No, NULL—Unknown.
corona result	Results of the first corona test performed on the subject. Categorical variable, 3 categories: • Positive—Indicates carrier of the virus. • Negative—No hands to carry the virus. • Other—not performed, at work or uncertain.
age 60 and above	Indicator 60 years or older (1) or under 60 years (0).
gender	Sex of the subjects. Male / Female / Null (unknown).
test indication	What is the indication for testing? • Abroad—arrival from abroad. • contact with confirmed—contact with Verified patient. • other—other indication or not specified.

The objective of Algorithm 1 is to identify the attribute values of an edge device distinctly without violating its security. The server model(w^0) is broadcasted to N all the clients, $C \subseteq \{C_0, C_1, ..., C_{tot}\}$. Along with the distributed server models, three different learning rates η are also broadcasted. The learning rates are selected from the array (η_m), which ranges from $[1e-1, 1e-5]$. The sample size is also chosen randomly and a more number of samples can also improve the training accuracy. Every edge device is offered w^0, which is duplicated for all η values and supervised independently for a full iteration. Some data features such as complexity, size, ambiguity, and variance are unique to edge devices. These data features will effect the training and thus the hyper-parameters η are selected carefully. Out of the three models at the edge device only one model with least loss $w_{\min}^0$ is selected. Every edge device will return $w_{\min}^0$, $\eta_{\min}$, and $losses_{\min}$. These statistics are important because of their ability to represent data on respective edge devices.

The models $\{w_0^0, w_1^0, \ldots, w_n^0\}$, learning rates $\{\eta_0, \eta_1, \ldots, \eta_n\}$ and their respective losses are obtained at the server. The server model is developed by integrating edge device models using the model aggregation technique. The model weights (w_n^0) are added iteratively as follows:

$$\sum_{i=0}^{n}(w_i^0) = w_0^0 + w_1^0 + w_2^0 + \cdots + w_n^0. \tag{1}$$

The output of the aggregation is divided by the number of clients, and the equation is as follows:

$$w^0 \leftarrow \frac{\sum(w_n^0)}{n}. \tag{2}$$

In phase 2, we assign every edge device, a cluster-ID, as demonstrated by Algorithm 2. This algorithm component has been restricted by the algorithm's main control loop, which continues for i iterations. Each i-th cycle,

1. genetic algorithms optimize hyper-parameters based on mutation, crossover, and evolution;
2. clients receive optimized hyper-parameters for cluster-based servers;
3. every client is prepared using a set of parameters;
4. combining client models produces the most effective server model.

Each cluster contains a unique range of hyper-parameters personalized towards the edge devices which are a part of it. Training initiates the development of such aggregated parameters every ith iteration. During genetic optimization, hyper-parameters interact with the ideal range for each iteration. Every iteration changes the contents of $\eta_{global}[k]$, which stores the learning rates for every cluster. The shape of the data is $< C, size(C_i) >$, where C shows the set of clusters, C_i shows the ith cluster and $size(C_i)$ shows the wide range of edge devices in every ith cluster. The losses of a cluster with the shape m_0 are used to sort the hyper-parameters.

$$losses_N, order \leftarrow sort(losses_N) \tag{3}$$

$$\eta_N \leftarrow sort(\eta_N, order) \tag{4}$$

After sorting, we achieve different users across crossover and mutation. The ideal performers continue to pass on their genetic mutations to the next generation, while others have developed by mating with previous transmission users as

$$\eta_{new} = \{\eta_0, \eta_1, \ldots, \eta_{size(C[m])}\} \tag{5}$$

The updated learning rates η_{new} determine whether effectively or partially through natural selection. The sum of η obtained from previous generations can differ slightly. The modified parameters are derived from (5):

$$\eta_{new} = \{\eta_{old}[0], \eta_{old}[1], \ldots, \left(\frac{\eta_{old}[P_A] + \eta_{old}[P_B]}{2}\right)\left(1 + \frac{f}{10}\right)\}, \tag{6}$$

There are two locations in which $P_A, P_B \in [0, 9]$ and $f \in [-1, 1]$.

All devices are configured according to their specific cluster hyper-parameters after genetic evolution. A training process involves repeating model aggregation, genetic optimization, and training for $i - 1$ iterations before a new epoch is achieved.

Artificial Neural Networks (ANN) were used in this work for classification of the data in each cluster. In this work, we used an ANN with three layers, input layer, a hidden layer, and the output layer. The hyper-parameters used in the ANN are as follows: activation function used in the hidden layer is relu, whereas sigmoid activation function is used in the output layer. The optimization function used is adam.

4. Results and Discussion

The objective of this section is to provide an overview of the experiments which were conducted for the evaluation and assessment of the genetic CFL architecture. Section 4.1 examines how genetic CFL architecture performs on the COVID dataset and how they compare with generic FL architecture. The genetic CFL architecture's performance analysis is discussed in Section 4.2.

4.1. COVID Dataset Performance for Genetic CFL Architecture

This subsection discusses the models' training and performance evaluation. COVID-19 dataset samples are first used to train the server model. In turn, clients are allocated the model based on their client ratio. For this experiment, 100 clients are randomly selected, and three client ratios are tested: 0.1, 0.15, and 0.3. Models' performance is generally evaluated using 10, 15, and 30 clients, respectively. Observations are chosen at random for each client device in the dataset. The purpose is to ensure that its observations are non-IID and replicate the key features of an actual situation. Section III discusses how the hyper-parameters are genetically modified after two training epochs. Tables 4 and 5 shows all such iterations, and Figure 2 plots the most successful performance against each round.

Table 4. Comparison of the performance of FL on various hyper-parameters on the COVID test data.

Client Ratio	Rounds	FL				
		Accuracy	**Loss**	**Precision**	**Recall**	**F1-Score**
0.1	3	0.9148	0.2609	0.9132	0.9146	0.9145
	6	0.9212	0.2524	0.9203	0.9211	0.9210
	10	0.9367	0.2115	0.9362	0.9366	0.9365
0.15	3	0.9156	0.2608	0.9151	0.9155	0.9154
	6	0.9197	0.2560	0.9192	0.9196	0.9195
	10	0.9224	0.2554	0.9216	0.9221	0.9220
0.3	3	0.9068	0.2758	0.9059	0.9064	0.9063
	6	0.9166	0.2648	0.9158	0.9164	0.9163
	10	0.9187	0.2629	0.9179	0.9185	0.9184

Table 5. Comparison of the performance of genetic CFL on various hyper-parameters on the COVID test data.

Client Ratio	Rounds	Genetic CFL				
		Accuracy	**Loss**	**Precision**	**Recall**	**F1-Score**
0.1	3	0.9271	0.2469	0.9267	0.9269	0.9268
	6	0.9271	0.2467	0.9268	0.9270	0.9269
	10	0.9271	0.2460	0.9270	0.9269	0.9268
0.15	3	0.9223	0.2552	0.9218	0.9221	0.9220
	6	0.9223	0.2551	0.9220	0.9222	0.9221
	10	0.9223	0.2549	0.9222	0.9221	0.9220
0.3	3	0.9208	0.2627	0.9199	0.9206	0.9205
	6	0.9208	0.2625	0.9205	0.9207	0.9206
	10	0.9208	0.2621	0.9207	0.9206	0.9205

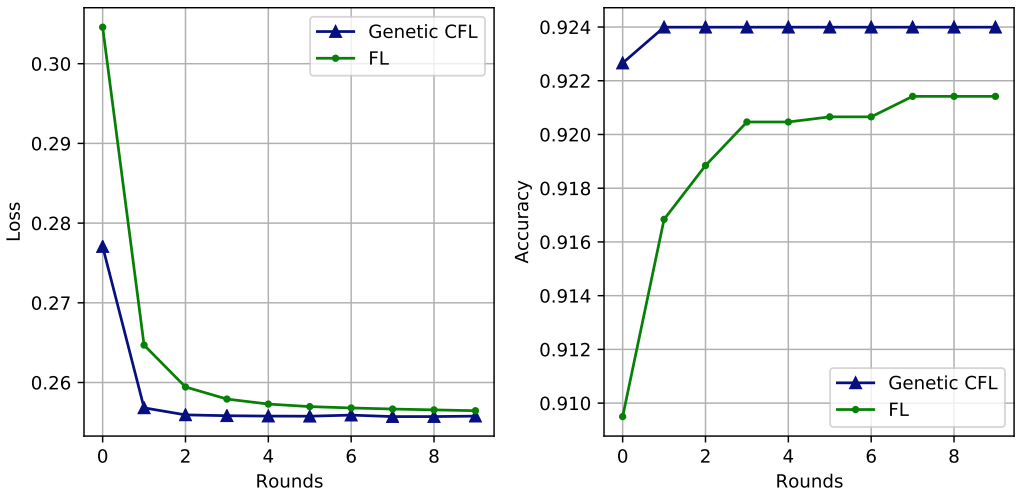

Figure 2. Evaluation of Loss and Accuracy on COVID-19 dataset- FL vs Genetic CFL.

As the training hyper-parameters cannot be determined earlier, the training and performance of the model are locally optimized, and the training is said to be more personalized [24,40]. Server models learn smoothly and converge faster than typical FL models after training. Tables 4 and 5 and Figure 2 illustrate the performance of the models for both the architectures in terms of accuracy, loss, precision, recall, and F1-Score. Each iteration shows how genetic CFL outperforms generic FL. The accuracy and loss are higher and lower in genetic CFL architecture than in the generic FL architecture. The accuracy and loss indicates that useful information is aggregated at the server. The loss value is used to train the ANN. However, accuracy or other metrics such as precision, recall, and F1-Score are also used to assess the training outcome. Table 6 depicts the training accuracy, training loss, validation accuracy and validation loss of the proposed genetic CFL algorithm on the COVID-19 dataset. From the table, it can be observed that in the first round, the maximum training and validation accuracy and minimum training and validation loss is attained after 1st epoch. After 1st epoch, the performance is reduced, indicating that genetic CFL algorithm has encountered overfitting problem. Similar performance can be noted in rounds 2, 3, 4 and 5. Hence, we can conclude that, in order to reduce the training time and the resource consumption (CPU and memory consumption) in the proposed genetic CFL approach, 1 epoch is sufficient for all the rounds.

Table 6. Accuracy and loss of genetic CFL in several rounds.

Round No	Epochs	Genetic CFL			
		Training Loss	Training Accuracy	Validation Loss	Validation Accuracy
1	1	0.6276	0.9126	0.6172	0.9126
	2	0.6090	0.9109	0.5986	0.9125
	3	0.5906	0.9109	0.5803	0.9125
2	1	0.4883	0.9109	0.4787	0.9125
	2	0.4726	0.9109	0.4636	0.9125
	3	0.4580	0.9109	0.4494	0.9125
3	1	0.3857	0.9109	0.3791	0.9125
	2	0.3754	0.9109	0.3694	0.9125
	3	0.3663	0.9109	0.3609	0.9125

Table 6. *Cont.*

Round No	Epochs	Genetic CFL			
		Training Loss	Training Accuracy	Validation Loss	Validation Accuracy
4	1	0.3262	0.9109	0.3220	0.9125
	2	0.3204	0.9109	0.3165	0.9125
	3	0.3153	0.9109	0.3116	0.9125
5	1	0.2926	0.9109	0.2892	0.9125
	2	0.2895	0.9109	0.2863	0.9125
	3	0.2868	0.9109	0.2836	0.9125

4.2. Genetic CFL Performance Analysis

The Genetic CFL method appears to perform better with a larger data set. As a result, improving the quality of test results for every sample might help overall hyper-parameter optimization. Considering several datasets that differ in data characteristics and data points, an ideal grouping of similar scenarios leads to higher model accuracy. There needs to be a balance between cluster size and cluster number. In the given instance, a perfect combination could verify that the performance of such methods as in decentralized design produces better results. In a practical application, the predicted number of edge devices is more than in an artificial environment. Increasing the number of clients results in improved performance. Genetic CFL optimizes hyper-parameters to increase throughput for a relatively small set of optimization iterations.

The proposed genetic CFL architecture performance is better than regular CFL architecture while using fewer iterations. According to COVID data, our architecture is more efficient and iterative in clustering. It provides that the proposed genetic CFL is flexible and adjustable method for optimizing hyper-parameters. The proposed architecture has the advantage of adaptability over other methods by allowing it to be tailored to meet the dataset and the necessary situation. The majority of other architectures require a lot of manual effort to adjust hyper-parameters. It resets the mechanism and loads a new set of parameters for data analysis and applications. There are both time and resource costs associated with the conventional mechanism. Furthermore, each client is tested separately, which affects server and client performance. The proposed genetic CFL model ensures service delivery for all client devices while increasing the server model's performance.

5. Conclusions and Future Directions

In this paper, we used the genetic algorithm to optimize the rate at which hyper-parameters are learned and the batch size for clustering through FL. To evaluate the performance of the proposed genetic CFL algorithm, we used the COVID-19 dataset. In addition, we discussed the best deployment conditions and limitations of the algorithm. In future, we would like to test the genetic CFL model on scalable and real-time datasets. The refinement of model parameters becomes accurate when the sample size increases, resulting in higher performance in the real-time scenario. We would also like to test the proposed model on several applications such as recommendation systems, image classification, and natural language processing. Furthermore, time-sensitive techniques could be combined with genetic CFL.

Author Contributions: Conceptualization, D.R.K.; Data curation, D.R.K. and T.R.G.; Formal analysis, D.R.K.; Methodology, D.R.K.; Resources, T.R.G.; Software, D.R.K.; Supervision, T.R.G.; Validation, T.R.G.; Visualization, T.R.G.; Writing—original draft, D.R.K. and T.R.G. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The dataset used in this work is taken from the kaggle repository https://www.kaggle.com/datasets/mykeysid10/covid19-dataset-for-year-2020?select=covid_data_2020-2021.csv(accessed on 4 April 2022).

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

COVID-19	Coronavirus
AI	Artificial Intelligence
ML	Machine Learning
FL	Federated Learning
GCFL	Genetic Clustered Federated Learning
PCFL	Privacy-preserving and Communication efficient scheme for Federated Learning
FedCD	Federated Cloning-and-Deletion
FedMA	Federated Matched Averaging
IFCA	Iterative Federated Clustering Algorithm
FeSEM	Federated Stochastic Expectation Maximization method
WOA	Whale Optimization Algorithm
MLP	Multilayer Perceptrons
ANN	Artificial Neural Networks

References

1. Liu, B.; Yan, B.; Zhou, Y.; Yang, Y.; Zhang, Y. Experiments of federated learning for covid-19 chest X-ray images. *arXiv* **2020**, arXiv:2007.05592.
2. Hageman, J.R. The coronavirus disease 2019 (COVID-19). *Pediatr. Ann.* **2020**, *49*, 99–100. [CrossRef] [PubMed]
3. Saleem, K.; Saleem, M.; Zeeshan, R.; Javed, A.R.; Alazab, M.; Gadekallu, T.R.; Suleman, A. Situation-aware BDI reasoning to detect early symptoms of covid 19 using smartwatch. *IEEE Sens. J.* **2022**. [CrossRef]
4. Mothukuri, V.; Parizi, R.M.; Pouriyeh, S.; Huang, Y.; Dehghantanha, A.; Srivastava, G. A survey on security and privacy of federated learning. *Future Gener. Comput. Syst.* **2021**, *115*, 619–640. [CrossRef]
5. Zhang, W.; Zhou, T.; Lu, Q.; Wang, X.; Zhu, C.; Sun, H.; Wang, Z.; Lo, S.K.; Wang, F.Y. Dynamic-Fusion-Based Federated Learning for COVID-19 Detection. *IEEE Internet Things J.* **2021**, *8*, 15884–15891. [CrossRef]
6. Lian, X.; Zhang, C.; Zhang, H.; Hsieh, C.J.; Zhang, W.; Liu, J. Can decentralized algorithms outperform centralized algorithms? A case study for decentralized parallel stochastic gradient descent. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 1–11
7. Manoj, M.; Srivastava, G.; Somayaji, S.R.K.; Gadekallu, T.R.; Maddikunta, P.K.R.; Bhattacharya, S. An incentive based approach for COVID-19 planning using blockchain technology. In Proceedings of the 2020 IEEE Globecom Workshops GC Wkshps, Taipei, Taiwan, 7–11 December 2020; pp. 1–6.
8. Alazab, M.; Tang, M. *Deep Learning Applications for Cyber Security*; Springer: Berlin/Heidelberg, Germany, 2019.
9. Kumar, R.; Khan, A.A.; Kumar, J.; Golilarz, N.A.; Zhang, S.; Ting, Y.; Zheng, C.; Wang, W. Blockchain-federated-learning and deep learning models for covid-19 detection using ct imaging. *IEEE Sens. J.* **2021**, *21*, 16301–16314. [CrossRef]
10. Yarradoddi, S.; Gadekallu, T.R. Federated learning role in big data, jot services and applications security, privacy and trust in jot a survey. In *Trust, Security and Privacy for Big Data*; CRC Press: Boca Raton, FL, USA, 2022; pp. 28–49.
11. Nilsson, A.; Smith, S.; Ulm, G.; Gustavsson, E.; Jirstrand, M. A performance evaluation of federated learning algorithms. In Proceedings of the Second Workshop on Distributed Infrastructures for dEep Learning, Rennes, France, 10 December 2018; pp. 1–8.
12. Victor, N.; Alazab, M.; Bhattacharya, S.; Magnusson, S.; Maddikunta, P.K.R.; Ramana, K.; Gadekallu, T.R. Federated Learning for IoUT: Concepts, Applications, Challenges and Opportunities. *arXiv* **2022**, arXiv:2207.13976.
13. Sattler, F.; Müller, K.R.; Wiegand, T.; Samek, W. On the byzantine robustness of clustered federated learning. In Proceedings of the ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Virtual, 4–8 May 2020; pp. 8861–8865.
14. Malekijoo, A.; Fadaeieslam, M.J.; Malekijou, H.; Homayounfar, M.; Alizadeh-Shabdiz, F.; Rawassizadeh, R. Fedzip: A compression framework for communication-efficient federated learning. *arXiv* **2021**, arXiv:2102.01593.
15. Fang, C.; Guo, Y.; Hu, Y.; Ma, B.; Feng, L.; Yin, A. Privacy-preserving and communication-efficient federated learning in internet of things. *Comput. Secur.* **2021**, *103*, 102199. [CrossRef]
16. Alazab, M.; Huda, S.; Abawajy, J.; Islam, R.; Yearwood, J.; Venkatraman, S.; Broadhurst, R. A hybrid wrapper-filter approach for malware detection. *J. Netw.* **2014**, *9*, 1–14. [CrossRef]

17. Ozfatura, E.; Ozfatura, K.; Gündüz, D. Time-correlated sparsification for communication-efficient federated learning. In Proceedings of the 2021 IEEE International Symposium on Information Theory (ISIT), Melbourne, VI, Australia, 12–20 July 2021; pp. 461–466.
18. Briggs, C.; Fan, Z.; Andras, P. Federated learning with hierarchical clustering of local updates to improve training on non-IID data. In Proceedings of the 2020 International Joint Conference on Neural Networks (IJCNN), Glasgow, UK, 19–24 July 2020; pp. 1–9.
19. Zhao, Y.; Li, M.; Lai, L.; Suda, N.; Civin, D.; Chandra, V. Federated learning with non-iid data. *arXiv* **2018**, arXiv:1806.00582.
20. Wang, H.; Yurochkin, M.; Sun, Y.; Papailiopoulos, D.; Khazaeni, Y. Federated learning with matched averaging. *arXiv* **2020**, arXiv:2002.06440.
21. Yao, X.; Huang, T.; Wu, C.; Zhang, R.; Sun, L. Towards faster and better federated learning: A feature fusion approach. In Proceedings of the 2019 IEEE International Conference on Image Processing (ICIP), Taipei, Taiwan, 22–29 September 2019; pp. 175–179.
22. Ghosh, A.; Chung, J.; Yin, D.; Ramchandran, K. An efficient framework for clustered federated learning. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 19586–19597. [CrossRef]
23. Kopparapu, K.; Lin, E.; Zhao, J. Fedcd: Improving performance in non-iid federated learning. *arXiv* **2020**, arXiv:2006.09637.
24. Gadekallu, T.R.; Pham, Q.V.; Huynh-The, T.; Bhattacharya, S.; Maddikunta, P.K.R.; Liyanage, M. Federated learning for big data: A survey on opportunities, applications, and future directions. *arXiv* **2021**, arXiv:2110.04160.
25. Zhang, C.; Xie, Y.; Bai, H.; Yu, B.; Li, W.; Gao, Y. A survey on federated learning. *Knowl.-Based Syst.* **2021**, *216*, 106775. [CrossRef]
26. Bonawitz, K.; Eichner, H.; Grieskamp, W.; Huba, D.; Ingerman, A.; Ivanov, V.; Kiddon, C.; Konečnỳ, J.; Mazzocchi, S.; McMahan, B.; et al. Towards federated learning at scale: System design. *Proc. Mach. Learn. Syst.* **2019**, *1*, 374–388.
27. Xie, M.; Long, G.; Shen, T.; Zhou, T.; Wang, X.; Jiang, J.; Zhang, C. Multi-center federated learning. *arXiv* **2020**, arXiv:2005.01026.
28. Chai, Z.; Ali, A.; Zawad, S.; Truex, S.; Anwar, A.; Baracaldo, N.; Zhou, Y.; Ludwig, H.; Yan, F.; Cheng, Y. Tifl: A tier-based federated learning system. In Proceedings of the 29th International Symposium on High-Performance Parallel and Distributed Computing, Stockholm, Sweden, 23–26 June 2020; pp. 125–136.
29. Likas, A.; Vlassis, N.; Verbeek, J.J. The global k-means clustering algorithm. *Pattern Recognit.* **2003**, *36*, 451–461. [CrossRef]
30. Kim, J.Y.; Cho, S.B. Evolutionary optimization of hyperparameters in deep learning models. In Proceedings of the 2019 IEEE Congress on Evolutionary Computation (CEC), Wellington, New Zealand, 10–13 June 2019; pp. 831–837.
31. Xiao, X.; Yan, M.; Basodi, S.; Ji, C.; Pan, Y. Efficient hyperparameter optimization in deep learning using a variable length genetic algorithm. *arXiv* **2020**, arXiv:2006.12703.
32. Aljarah, I.; Faris, H.; Mirjalili, S. Optimizing connection weights in neural networks using the whale optimization algorithm. *Soft Comput.* **2018**, *22*, 1–15. [CrossRef]
33. Beruvides, G.; Quiza, R.; Rivas, M.; Casta no, F.; Haber, R.E. Online detection of run out in microdrilling of tungsten and titanium alloys. *Int. J. Adv. Manuf. Technol.* **2014**, *74*, 1567–1575. [CrossRef]
34. Agrawal, S.; Sarkar, S.; Alazab, M.; Maddikunta, P.K.R.; Gadekallu, T.R.; Pham, Q.V. Genetic CFL: Hyperparameter optimization in clustered federated learning. *Comput. Intell. Neurosci.* **2021**, *2021*, 7156420. [CrossRef] [PubMed]
35. Zhang, H.; Zhang, M.; Liu, X.; Mohapatra, P.; DeLucia, M. FedTune: Automatic Tuning of Federated Learning Hyper-Parameters from System Perspective. *arXiv* **2021**, arXiv:2110.03061.
36. Khodak, M.; Tu, R.; Li, T.; Li, L.; Balcan, M.F.F.; Smith, V.; Talwalkar, A. Federated hyperparameter tuning: Challenges, baselines, and connections to weight-sharing. *Adv. Neural Inf. Process. Syst.* **2021**, *34*, 19184–19197.
37. Ibraimi, L.; Selimi, M.; Freitag, F. BePOCH: Improving federated learning performance in resource-constrained computing devices. In Proceedings of the 2021 IEEE Global Communications Conference (GLOBECOM), Madrid, Spain, 7–11 December 2021; pp. 1–6.
38. Taheri, R.; Shojafar, M.; Alazab, M.; Tafazolli, R. FED-IIoT: A robust federated malware detection architecture in industrial IoT. *IEEE Trans. Ind. Inform.* **2020**, *17*, 8442–8452. [CrossRef]
39. Qayyum, A.; Ahmad, K.; Ahsan, M.A.; Al-Fuqaha, A.; Qadir, J. Collaborative federated learning for healthcare: Multi-modal covid-19 diagnosis at the edge. *arXiv* **2021**, arXiv:2101.07511.
40. Arikumar, K.; Prathiba, S.B.; Alazab, M.; Gadekallu, T.R.; Pandya, S.; Khan, J.M.; Moorthy, R.S. FL-PMI: Federated learning-based person movement identification through wearable devices in smart healthcare systems. *Sensors* **2022**, *22*, 1377. [CrossRef]