Swag Technique and Dirichlet Distribution to Address Non-Iid Data in Federated Learning

Ahmed Al-Ghanimi[1*], HAYDER AL-GHANIMI[2]

[1*] *Department of Computer Science, Faculty of Pharmacy, Babylon University, Babylon, Iraq*.

E-mail: ahmed.h.o.alghanimi@gmail.com

[2]*Technology Engineering Department of Medical devices,Hilla University College, Babylon, Iraq.*

Abstracts: Federated learning deals with the challenge of accessing data from different information sources while preserving their privacy in centralized learning. We can use this paradigm to learn a common global model for multiple clients using model aggregation cycles, without sharing data. Here aggregating the local models is a crucial part of the training. However, the model may experience accuracy and performance loss while aggregating heterogeneous data. We propose a new aggregation method with sampling, FEDSBME, using the Bayesian inference. We sample the local models of the participating clients and build a Bayesian ensemble model to create a powerful aggregation. The sampling of local models is performed using two approaches, SWAG and Dirichlet distribution sampling. Our experimental results prove that our suggested approach can preserve the accuracy and performance of the model when clients' data are heterogeneous (non-iid) and with deeper neural networks.

Keywords: Federated learning, SWAG, Dirichlet distribution, Bayesian ensemble model.

1. INTRODUCTION

The Federated learning (FL) is a form of distributed learning [1]. It trains the artificial intelligence models using several data resources while the data is kept undisclosed [2, 3]. Training data is distributed between clients such as smartphones, hospitals, and local information sources[4, 5]. Some of the complexities in centralized learning that gave rise to the emergence of federated learning are: 1- Sensitivity of data privacy 2- transferring large amounts of data to the server through the network is time-consuming 3- Real-world data are often non-iid, leading to client heterogeneity [6]. FL's nature and key features in server-clients communications solve the problems challenging centralized learning. Federated averaging (FedAvg) is the most popular and used FL algorithm to manage the training process and the relationships between clients and the server [7-9]The crucial part of the FL process is transferring parameters from clients to the server and vice versa. This is an active research field in FL.

The effect of heterogeneous clients with non-iid data becomes evident in the most critical step in FL, the aggregation of the weights. Non-iid data are also known as statistical heterogeneity or unbalanced data[10]. In FedAvg, this kind of data causes model divergence and distancing from local optimums[11]. This makes the performance of FedAvg in the face of statistical heterogeneity a current open research question. To solve the problem of aggregation with unbalanced data, many researches have presented approaches for strong aggregation. Liu and Zhong [12] suggested a Bayesian approach to estimate the global posterior probability $p(\theta|D)$ of the multivariate Gaussian function $N(\mu, \sum)$. They updated the mean and covariance of parameter in weight aggregation step which maximize memory use and communicative overhead. Li et al. [13]suggested FedProx based on FedAvg. They used an isotropic penalty function but they make θ local models move away from the global optimum. Yurochkin et al[14]. proposed a non-parametric Bayesian approach to adjust the clients weights using adjusting the repeated layers before averaging. This algorithm has linear computational and communicational dependency that makes it not suitable for deeper neural networks. The above issues make proposed methods unsuitable to solve the problem of aggregation with non-iid data.

If Bayesian inference approach is used in the aggregation of weights with unbalanced data [15], we can reach a more optimal estimate of the global posterior instead of point posterior estimate. We aggregate the models achieved in each client's training cycle and sample the global model from them using different methods. This minimizes the client drift due to unbalanced data considerably and minimizes their distance from the global optimum[16]. Two special advantages of this approach are: first, sampling of each client reduces the cost of transferring large amounts of parameters and increasing communication cycles. Second, sampling gets us closer to preservation of privacy. The key point of this theory is client sampling. Even when there is discrimination and maximum differences in the statistical population's frequency, the client-sampling approach is guaranteed to gets

(*Ahmed Al-Ghanimi) Corresponding author

close to the normal state, while reducing the global model divergence due to the clients' drift [16].

In this paper, we suggest a sampling-based aggregation method to estimate the global posterior. We first average the weights of the clients in each training epoch using SWA and put the results in a Bayesian ensemble model. We can infer many other distributions of the global model by sampling this model. We use two methods, Gaussian (normal) distribution sampling and Dirichlet distribution sampling. Server uses the weights sampled from the Bayesian ensemble model in each new epoch. We set two goals to reach a stronger prediction than the previous approaches. 1- presenting a new aggregation method to solve the problem of model divergence when using non-iid data. 2- Increasing the optimality of the model and converging to the best point by Gaussian optimization of SWA (SWAG). We use a Stochastic Weight Averaging (SWA) with a cyclic or learning rate to scan the model while pulling ourselves us out of the noisy local minima to build a better Bayesian ensemble model. We call our algorithm FedSBME. A Covid-19 data with a 3-class classification (suffering from Covid-19, suffering from pneumonia, healthy) is used for the experiments. Our suggested algorithm keeps its precision and performance in the face of non-iid data and neural networks with deeper architectures.

Most trained models in the literature can address the challenges of non-IID data but perform the execution process (model training and testing) with maximum parameter transfer in numerous communication epochs, which increases communication costs. Our proposed approach, on the other hand, is capable of superior training with high performance with minimum parameter transfer in the least number of epochs. The present research employs 4 distinctive COVID-19 datasets for the execution process. This training strategy with real data is scarcely seen in other FL-based studies. Nevertheless, we managed to reach the highest convergence with deeper architectures (ResNet-50). Therefore, our research direction is as follows:

- The use of novel FL with the prospect of solving the problem of imbalanced data from heterogeneous clients in an attempt to diagnose COVID-19 as the current major health issue

- The development of a new aggregated approach based on Bayesian inference with a sampling technique in each epoch to provide a secure framework for exchanging the least number of parameters between the collaborative systems in the training process

- The ability to protect privacy as well as beneficial information exchange and reach convergence promptly in minimum communication epochs

- The use of SWA instead of SGD in each client to obtain the best global optimum

This paper consists of the following sections. Section 2 briefly reviews achievements in related work. Then the proposed method of this article will be described in Section 3. Section 4 also addresses the evaluation of the performance of the proposed method compared to other methods. Finally, in section 5, conclusions and future orientations are also outlined.

Table 1. Advantages and disadvantages of some of the proposed methods.

Author(s)	Proposed approach	Advantages	Disadvantages
Al-Shedivat et al. [27]	Presented a global posteriori inference by averaging local posterior	Reducing complexity	Not using a new approach
Guo et al. [25]	Introduced constant federal learning to merge different aspects of data heterogeneity	Used a regularization technique to access previous epoch data	Constant learning rate despite applying Gaussian noise to the gradients (we leave the nose)

Yun et al [19]	Can update low-dimensional gradients using a low-dimensional global model.	Reducing communicational costs by considering the certainty coefficient for each client	We can access the original data using reverse engineering. Using data averages are not just a sufficient condition to keep data secure.
Liu et al. [12]	Used Bayesian inference to estimate the local posteriors and multivariate Gaussian distribution to solve heterogeneity problem.	Using Laplace approximation with priori distribution for online local training of the clients	Not experimenting with unlabeled data
Zhu et al. [39]	Introducing data-free knowledge distillation approach to deal with the heterogeneous data problem	Using knowledge distillation to solve the problem of heterogeneous data to make the model converge in less communication cycles	Not using real-world data in the experiments and focusing on local minima
Huchinson et al. [28]	Calculate the posterior distribution of the parameter using the observed data	Uses KL divergence between local distributions leads to correct prioris of local functions	Not using proper data set for experiments
Li et al. [36]	Uses semi-supervised knowledge distillation for heterogeneous data in federated learning	Makes preserving the privacy of the participants and improving the communication possible by introducing adaptive aggregation method.	Data generated by the generator is not accurately tested. No security analysis for data.
Li et al. [21]	Introducing the approach of federated learning to preserve privacy	Controlling the drift of local architectures	Not focusing on the data heterogeneity

2. RELATED WORKS

FL Standard FL with balanced data has a proper default setting [17]. Due to the observed difference between the federated learning and the distributed one, we'll still face different challenges in each of them. Communication between the server and the clients has many bottlenecks. This makes dealing with them effective in reaching the server's maximum efficiency and best end result. Preserving privacy is essential in this regard. This mean data should be kept and should remain in its local location. Current studies are mostly about the stability, convergence, and communicational costs of federated learning while using non-iid data[3, 13, 16, 18-23]. Some of the suggested models use shared data between the server and the clients to reduce model drift [20]. Others use adaptive optimization to better update the global model [11]. Lin et al. [23] research resembles ours. They made merging the models possible. Li et al. [13] looked for a better local training of the clients in heterogeneous conditions. Liu et al [24] used a Bayesian approach to estimate posterior locally to solve the precision drop issue when using heterogeneous data. Guo et al. [25] present a constant federated learning (CFL) to merge various aspects of the data heterogeneity. Thorgeirsson et al. [26] calculate the weight uncertainty in aggregation stage of federated learning algorithm (FedAVg) and find the posterior distribution of the model's weights using Bayes rule. They used stochastic features of the Fourier transform to get a Gaussian estimation and $\mathrm{N}(\mathrm{V_n}, \partial^2 \sum_{\mathrm{n}}^{-1})$ as the posterior belief.

Al-Shedivat et al. [27] presents a global posterior inference by averaging local posteriors. Hutchinson et al. [28] found the posterior distribution of the parameters using the observed data and used it to find the distribution of predictions of y. There has been studies on non-convex federated learning settings [29, 30]. Woodworth et al. [31] did a Stochastic Gradient Decent (SGD) analysis using heterogeneous data. In homogeneous settings, LocalSGD is better than Minibatch SGD, but it is not true for heterogeneous data. Wang et al. [32] suggested a normalized averaging method that removes the objective paradox and errors of convergence. Wang et al[33]. found a suitable trade-off between local updating and global aggregation using non-iid data. It tries to minimize the loss function and maneuvers on non-convex methods. Liu et al. [30] generalize the Adagrad optimization steps of stochastic gradient decent optimization to reach an adaptive convergence rate for non-convex functions. Shoham et al. [34] reduce weight divergence by optimizing a decomposed global posterior into a computational formula $p(\theta|D) = p(D_n|\theta) + p(\theta|D_n)$ during training. In this equation D_n represents a local data complement and n represents the n^{th} client. Similar works [35-38] use knowledge distillation along with Bayesian inference for inference and getting better precision. All researches always look for better precisions and faster model convergences. Our method outperforms them due to using Dirichlet distribution and SWAG which leads to better convergence and management of the heterogeneous data. The following table shows some of the advantages and disadvantages of the proposed methods.

3. OUR PROPOSED METHOD: BAYESIAN ENSEMBLE MODEL FOR STRONG AGGREGATION

The newly developed federated learning (FL) aims to obtain a globally optimal model for the server in the process of ensemble learning with multiple clients. The server's globally optimal model is accompanied by the server's maximum learning rate. This goal is achieved by maximizing the global posterior, P(θ|D) , on the server. As seen in Fig. 1, we employ a Bayesian model ensemble to enable the maximum learning rate in non-IID data-involved training. In the first step of training clients, we use a multivariate Gaussian function to aggregate the clients' parameters. By using this distribution, we aim to approximate the local posteriors of each client, so that we can eventually approximate the global posterior which maximizes learning on the server. In the second step, each client's diagonal covariance and average are what we transfer to the server (the reason for selecting the diagonal covariance is explained in the following). By aggregating the covariance and average in a common model, we seek a model ensemble the framework of which is based on the Bayesian inference. In the third step, the model ensemble is sampled through both the SWAG technique (left figure) and Dirichlet distribution (right figure). The samples are transferred to the server. Finally, in the fourth step, the server initiates its new training epoch with the samples taken from the clients, and in the fifth step, upon the termination of given training epochs, it transfers the average of implementation parameters. This process (steps 1 to 5) continues until the desired accuracy is achieved on the server. It should be noted that we assume imbalanced data in heterogeneous clients, according to which the maximum desirable accuracy is obtained in five steps.

3.1. Sub-headings

Fig. (1) gives a view of the new aggregation methods (red box). We propose a Bayesian approach to deal with the model drift when using non-iid data. Our goal is to maximize the model learning when the data distribution is unbalanced. By maximizing global posterior, we can maximize learning in the server. We need a global posterior estimation $p(\theta|D)$ that learning from the global models and reaching an optimum model ($\bar{\theta}$, $\bar{\theta}$), are its two special cases. Using posterior estimation, we can infer these two cases. Estimating the output probability $p(y|x;\theta)$, inferred from a global model with parameter θ, is one approach to reduce the model's drift using Bayesian inference. We wish to see the effect of parameter θ in predicting y. To predict the output, we combine all of the output probabilities of local models.

$$p(y|x;D) = \int p(y|x;\theta)p(\theta|D)dw \quad (1)$$

In equation 1 the output probability using all the samples observed by the model is the product of output probability y affected by parameter θ times the posterior probability. In FL maximizing the global posterior to reach a global optimum is a common goal. However, evaluating it in the federated learning is impossible [12]. Generally, the mentioned argument means equation (3-1) cannot be solved.

This makes us estimate it using Monte Carlo methods $£ = \frac{1}{N}\sum_{i=1}^{N}\theta_i \equiv \bar{\theta}$ as an aggregation of the local model, global model, and its posterior. Using m models, we estimate the posterior as follows:

$$p(y|x;D) \approx \frac{1}{m}\sum_{m=1}^{M} p(y|x;\theta^m) \quad (2)$$

Equation (2) shows an ensemble model called a Bayesian ensemble model in Bayesian approach. In this model $\theta^m \sim p(\theta|D)$. This means aggregating the clients' output gives us a sequence of θ parameters. We'll use this sequence to estimate the global posterior probility. The challenge now is to estimate the client model using the posteriors. We'll discuss it in the next section.

3.2. Bayesian model set with estimated posteriors

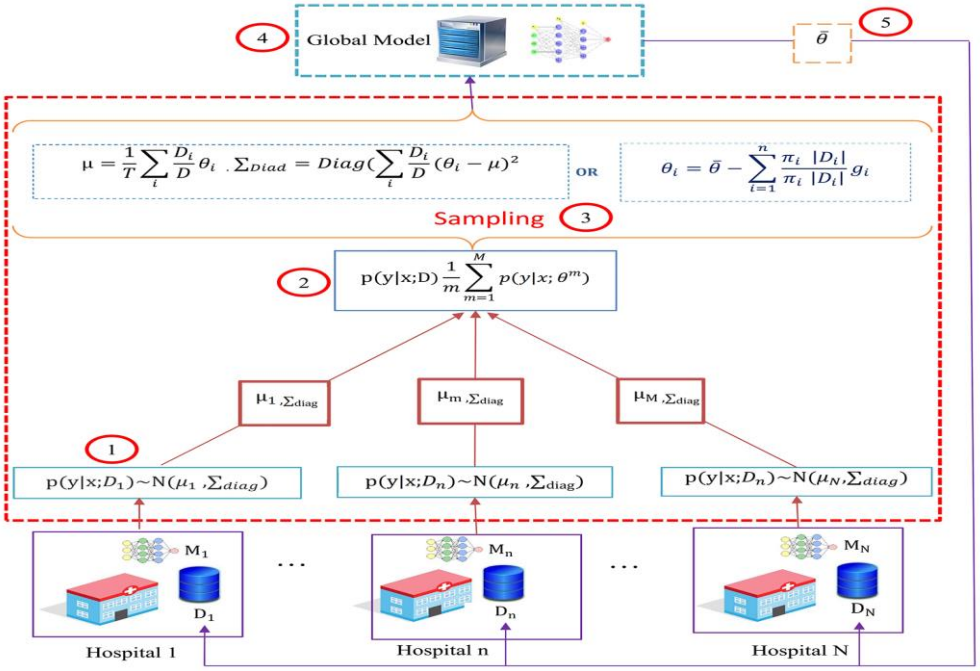

Figure 1. The new aggregation method based on sampling of the Bayesian ensemble model.

3.2.1. Sampling a Bayesian ensemble group using a Gaussian multivariate distribution SWAG

Using Gaussian Stochastic Average Weighting (SWAG) instead of SGD is one of the methods to estimate the posterior probability based on the clients' models. This provides a solution to last section's challenge. We can estimate $p(\theta|D)$ easily in this case. It's best to discuss SWA, the base method, before studying SWAG. SWA has cyclical or constant learning rate and begins the job using a pre-trained solution. It starts from the start point of SGD and reaches the optimum points by tracking it in the local geometrical space. At the end of training epochs, we average our solution with the SGD solution to get to the SWA special solution, which leads to a more optimum solution. However, if the training rate in SWA is cyclical based on FGE, it will jump some of the local minima [40] . At the end of the training, we average the calculated weight points and SGD weight points to present the solution to find the optimal θ. Advantages of SWA over SGD are using less memory, has little computational overhead, and finds a more extensive and optimal solution in the end. Using SWA leads us to a Bayesian framework to estimate the global posterior with it. Actually, we can easily find the posterior region in the weight space by searching all the weight space while updating parameters. We begin by SWAG, a Gaussian distribution SWA. SWAG builds the Gaussian distribution $p(\theta|D)$ by adjusting the parameters based on the weights resulted from SGD. Maddox et al. [15] convert SWA into a Gaussian (normal) distribution SWAG using a multivariate Gaussian model $N(\mu, \sum_{diag})$. Here $\sum$ is the covariance matrix tracking the local changes along the local geometry. We update the covariance matrix using every new θ_i point reached along the SWA path. Considering the millions of parameters in of neural networks, computing covariance $\sum$ in each round R will need complex computations and lots of memory to update all of them in the rounds. It makes us use its diagonal matrix $\sum_{diag}$ instead. Now in each round we just update the main diagonal elements and update the rest of the matrix's element in the last round of SGD. This helps us to

estimate the global posterior by averaging Bayesian model. We can sample this Gaussian method and perform the Bayesian model averaging.

In FL, operation $g_i = -(\theta_i - \bar{\theta})$ represents k steps of SGD for a small set of client's data $D_i \subset D$ (each client's data as a subset of all data sampled from all clients). θ_i Parameters calculated from each client model are considered in the step k of updating weights to go through global model parameters. We use the diagonal Gaussian distribution $N(\mu,\sum_{Diag})$ to adjust client models with each other.

$$\mu = \frac{1}{T}\sum_i \frac{D_i}{D}\theta_i \qquad ,\sum_{Diag} = Diag(\sum_i \frac{D_i}{D}(\theta_i - \mu)^2 \tag{3}$$

In equation (3), we calculate D_i using samples from client μ, a fraction of the whole sample data, and update the diagonal covariance resulted from the weight space tracking. Now we can easily use this $\theta^{(m)} \sim N(\mu,\sum_{Diag})$ distribution to sample our Bayesian model set in equation 4. we square each element using $(\theta_i - \mu)^2$ expression. Note that in both possible cases in the client's model $\bar{\theta}$ and $\{\theta_i\}$ are considered a sample of $N(\mu,\sum_{Diag})$ distribution.

3.2.2. Sampling a Bayesian ensemble model using a Dirichlet distribution

As another method to estimate the global posterior, we use the Dirichlet method. To combine expert models in supervised learning, the Bayesian inference uses combined models as the prior distribution. In this method, we use stochastic gradients to analyze $\mathrm{p}(\theta|\mathrm{D})$ estimation. Considering the client's model as $\theta_i = \bar{\theta} - g$, a convex combination or weighted average of the clients' models can become a better model for each client.

$$\theta = \sum_{i=1}^{n} \frac{\pi_i |D_i|}{\sum_i \pi_i |D_i|} \qquad , \qquad \theta_i = \bar{\theta} - \sum_{i=1}^{n} \frac{\pi_i |D_i|}{\pi_i |D_i|}\, g_i \tag{4}$$

In equation (4) $\pi = \left|\pi_1, \ldots, \pi_{|S_n|}\right|^T \in \Delta^{|S_n|-1}$ reperesnets a vector showing each client's parameters and D_i is the clients' sample data. We use the Dirichlet distribution $Dir(\Upsilon)$ as the distributied model of $\theta^{(m)}$ and sample $\theta^{(m)}$ using the following equation:

$$\theta^{(m)} = \sum_i \frac{\pi_i^{(m)}}{\sum_i \pi_i^{(m)} D_i}\theta_i \qquad , \pi^{(m)} \sim p(\pi) = p(\pi_1, \ldots, \pi_{|S_n|} = \frac{1}{B_\Upsilon}\prod_i \pi_i^{\Upsilon_i} \tag{5}$$

In the above equation, if $\Upsilon = \left[\Upsilon_1, \ldots . \Upsilon_{|S_n|}\right]^T > 0$, we have a Dirichlet distributed parameter.B_Υ Is a multivariate Beta function for normalization Generally, we perform two operations in the Bayesian ensemble model method.First, we calculate the posterior distribution $p(\theta|D)$ from client models. Second we sample the $\theta^{(m)} \sim p(\theta|D)\}_{m=1}^{m}$ estimation using Bayesian ensemble model. We mostly solved the problem of heterogeneous data in sampling process and converting each client into a global model in communicating with server. We'll prove our claims in the experiments section. FedSBME algorithm is as follows.

In algorithm 1, the server begins by a fully supervised, adjusted learning rate training of the models using labeled data and SWA method (line 1). The resulting (θ) parameters will be sent to all clients after training the model. The S_n client initializes its model with the θ parameters and begins its Rth round of training. Each client performs training in K steps using its local data and an adjusted local training rate using SWA method (line 2-8). Once adjusted epochs finished, the client models of the current rounds are sampled and a Bayesian ensemble model is built. Then we estimate the global posterior and sample using equation (3) or (5) (SWAG or Dirichlet distribution) (Line 9-13) Finally we send the θ_i resulted from sampling to the server. Server returns the resulting θ_i after training θ with its own data (Line 15).

Algorithm1: FEDSBME

1: Server input: θ and η_{SWA} . $labeled\ data\ \{x_i, y_i\}_{n=1}^{N}$;

2: Clients input: k Local step size and model local labeled data

3: for $r \leftarrow 1$ **to** R **do**

4: **Sample** clients $S_n \subseteq \{1, \ldots, N\}$;

5: **Communicate** to all clients $i \in S_n$;

6: **For** each client $i \in S_n$ in parallel **do**

7: **Initialize** local model $\theta_i \leftarrow \theta$;

8: $\theta_i \leftarrow$ **Client Training in each Round** (θ_i , D_i,η_i, k);

9: **Create** £ $= \frac{1}{N} \sum_{i=1}^{N} \theta_i \equiv \bar{\theta}$, $\bar{\theta} = \sum_{i \epsilon S_n} \frac{D_i}{\sum_{i \acute{\epsilon} S} D_i} \theta_i$;

10: **Create** global model distribution $p(\theta|D)$ from $\{\theta_i ; i \epsilon S_n\}$;

11: **Sample** m global models $\{\theta^{(m)} \sim p(\theta|D)\}_{m=1}^{M}$;

12: **Create** $\{\theta^{(m)}\}_{m=1}^{M} = \{\bar{\theta}\} \cup \{\theta_i ; i \epsilon S_n\}$;

13: **Transfer** θ_i to the Server;

14: **End**

15: Server output: $\bar{\theta}$

4. EXPERIMENTAL RESULTS

4. 1. Result's preparation

4. 1.1. Setting and parameters in the proposed method

To execute and evaluate the proposed method, we use CNN architecture shown in Fig. 2. We use a batch normalization (batch size 128) in 100 rounds for 4 clients. We do an $M = 4$ (4 models belonging to 4 clients) sampling from the Bayesian ensemble model using learning rate $Lr = 0.01$ in the first round and multiply the learning rate by (0.9, 0.6, 0.3) respectively in a scheduled way. Finally, we calculate the new learning rate for the current round. Transferring data to the clients is done using Dirichlet distribution. We set a constant server learning rate $Lr = 0.01$. Setting a very large learning rate in non-iid local data distribution will cause a slow convergence for a large number of communication cycles. Models cannot reach the desired performances in a small number of communication cycles if the learning rate is set too small. Using a scheduled adaptive learning rate will give the best performance. Our results show reaching the desired accuracy and performance needs reducing the convergence rate 20% to 70% in each round. We set sampling rate for the Bayesian ensemble model at $C = 0.03$ in each training epoch. Using Resnet50 model gives us good accuracy compared to FedProx. We use Resnet50 as the backbone of a pre-trained model, and a new classification head instead of the existing one. We use SoftMax activation function and entropy loss function. Images are 224×224 pixels and are distributed unevenly into an

unbalanced 3-class output between 4 clients. Our server has an i7 CPU, a GTX 1660 GPU, and used Pytroch 10000, Windows 10, Python 3.8.1 with Anaconda, and CUDA 11.3.

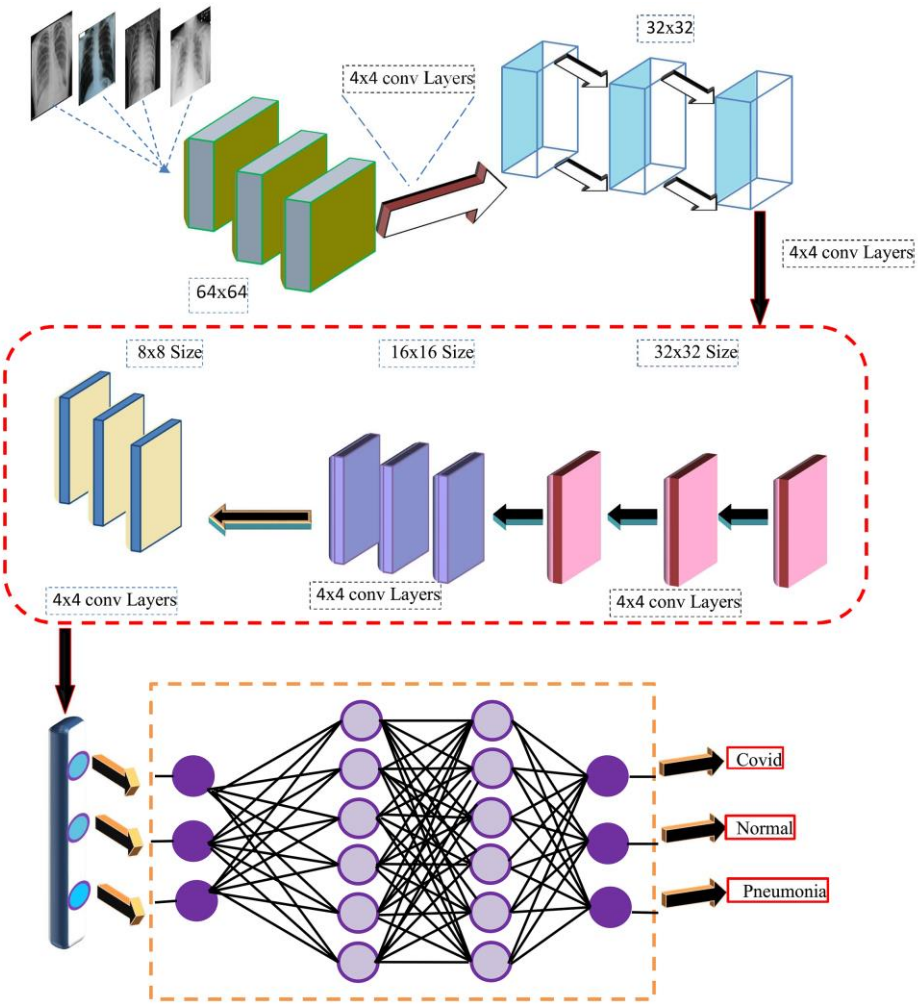

Figure 2. The CNN architecture for feature extraction and Covid-19 classification.

4. 1. 2. Datasets

We used several datasets available in research resources and Kaggle repository. This dataset consists of X-Ray images belonging to three classes: suffering Covid-19, healthy, and suffering pneumonia. In each training step, we used 30% of the dataset to train the server and distributed the remaining 70% between clients. 20% and 10% of the server 30% share was used for training and test, respectively. 40% of the client's 70% share is used for training and 30% is used for testing local model. Table 2 shows the details of the datasets for each class.

- **Dataset 1:** This dataset is from a Covid-19 radiography database classified into three classes. It is equally divided into 438 cases in each class (suffering Covid-19, suffering viral pneumonia, healthy).

- **Dataset 2:** This dataset is gathered from Kaggle repository. It was put into two separate folders for training and testing purposes. The training set included 111 'suffering from Covid-19' images, 70 healthy images, and 70 'suffering from viral pneumonia' images. The test set included 26, 20, and 20 images from suffering from Covid-19, healthy, and suffering from viral pneumonia people, respectively.
- **Dataset 3:** The third data set is also taken from Kaggle repository. It contained 3788 two-dimensional x-ray images. This included 1102, 1345, and 1341 x-ray images from people suffering Covid_19, suffering viral pneumonia, and healthy ones respectively.
- **Dataset 4:** This dataset is taken from a radiography database researchers gathered from various countries for Covid-19 diagnosis research purposes. In three update, it is a total of 15153 images in a 3-class classification. It included 3616, 10192, and 1345 images from people suffering Covid_19, viral, healthy

people, and people suffering from pneumonia, respectively. The follwing figure shows pictures from the dataset with a 3-class classification. We'll give more details of getting results using these datasets.

Table 2. Datasests table showing details of each class.

Dataset Name	Reference	Total Sample no.	Suffering Covid-19	Healthy	Suffering pneumonia
DataSet1	[41]	1314	438	438	438
DataSet2	[42]	317	137	90	90
DataSet3	[43]	3788	1102	1341	1345
DataSet4	[44]	15153	3616	10192	1345

4. 1. 3. Evaluation criteria

We evaluated the proposed method using four criteria: F1-Score, Accuracy(Acc), Specificity(SPE), and Sensitivity(SEN). The detailed results of our evaluation using those criteria will be described in the later sections.

$$ACC = \frac{TP+TN}{TP+TN+FP+FN} \quad (6)$$

$$PREC = \frac{TP}{TP+FP} \quad (7)$$

$$SEN = \frac{TP}{TP+FN} \quad (8)$$

$$SPE = \frac{TN}{TN+FP} \quad (9)$$

4. 2. Evaluation of the proposed method

We first describe the reasons to choose 4 clients for the test process. Fig (3), resulting form the initial test, gives a compelling reason for this. Then, we'll show the results of running the experiments on the clients and server as diagrams.

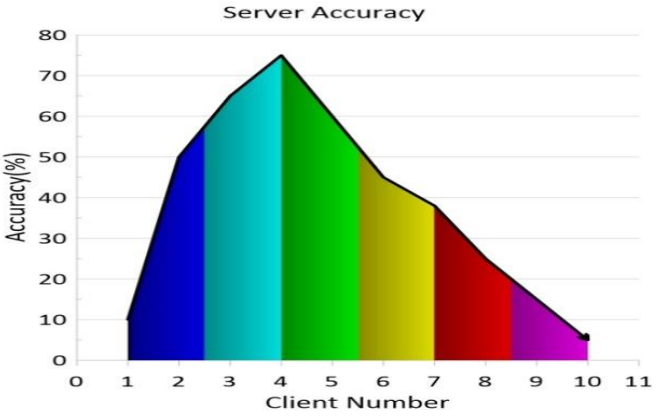

Figure 3. A view of training the server using different clients.

As shown in Fig (3), the server is most accurately trained when there are 4 clients. The model's accuracy drops with further increase in clients' number. We reduced the number of clients in training the server to 4.

Table 3. Server's classification results during proposed method's evaluation.

Type	Class	$PREC_{cl}$	SPE_{cl}	SEN_{cl}	ACC_{cl}	F1-score
Server	Covid-19	99.01	97.11	94.64	98.88	98.49
	Normal	98.26	96.64	99.37	99.01	99.32
	Pneumonia	98.33	98.25	97.26	98.65	98.06

Table 4. Clients' classification result during proposed method's evaluation.

Type	Class	$PREC_{cl}$	SPE_{cl}	SEN_{cl}	ACC_{cl}	F1-score
Client 1	Covid-19	97.06	93.19	87.95	93.57	93.12
	Normal	98.64	96.39	93.19	96.59	94.75
	Pneumonia	98.99	97.14	94.75	96.0	98.19
Client 2	Covid-19	98.32	94.11	92.21	95.67	97.13
	Normal	99.01	97.36	97.33	97.31	98.07
	Pneumonia	96.78	98.67	93.49	97.67	99.67
Client 3	Covid-19	96.15	93.73	97.27	92.43	98.43
	Normal	98.22	94.91	93.39	98.33	97.55
	Pneumonia	96.41	98.43	98.18	94.71	94.92
Client 4	Covid-19	98.29	96.08	92.44	96.66	89.34
	Normal	94.36	95.36	98.53	98.46	97.48
	Pneumonia	97.79	97.32	99.94	97.05	99.25

Figs. (4) and (5) diagrams show the results of implemening the proposed approach on a server and 4 clients. In Fig. (5) the accuracy of the client 1 increases till round 35, but decreases slightly after that. Finally, after round 70, we see a constant accuracy trend in the execution. In client 2, the accuracy increases up to round 10, and after some decreasing, finally becomes increasing again in round 58, and reaches a constant accuracy of 98%. In client 3, accuracy keeps increasing till round 30, gets to its maximum up to round 58, and we'll see a constant accuracy trend after that.

In client 4, the accuracy increases till round 4, gets to its maximum at round 75, after some ups and downs, and then always remains constant druing the experiment. The accuracy of the model training in the first 10 rounds is good. it has a short constant trend from round 50, and finally becomes increasing from round 85 in the rest of the training epochs. The server training results are crucial for us. Considering the theories and ideas described in section 3, and special setting used in training and testing the model, our proposed model practically increases the accuracy of the model in less communication cycles when using unbalanced data. We stopped the experiments here. We used the criteria of section 3.1.4 for one serve, 4 clients evaluation results in tables (4) and (3) Covid-19 detection accuracy in clients 1, 2, ..., and server were 93.57, 95.67, 92.43, 96.66, 98.88 percent respectively. Server had the best detection accuracy. Accuracies in detecting healthy people were 96.59, 97.31, 98.33, 98.46, 99.32 percent respectively.

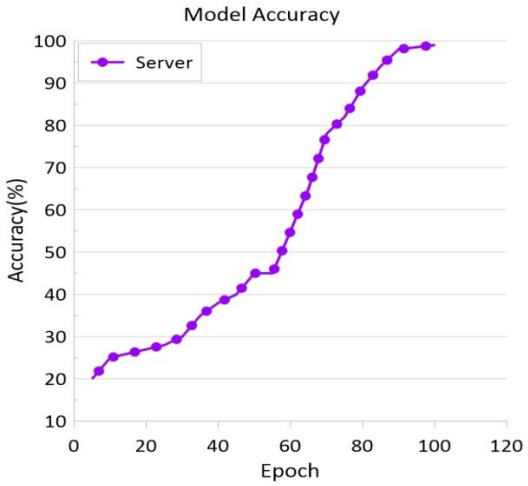

Figure 4. Server's result during model training.

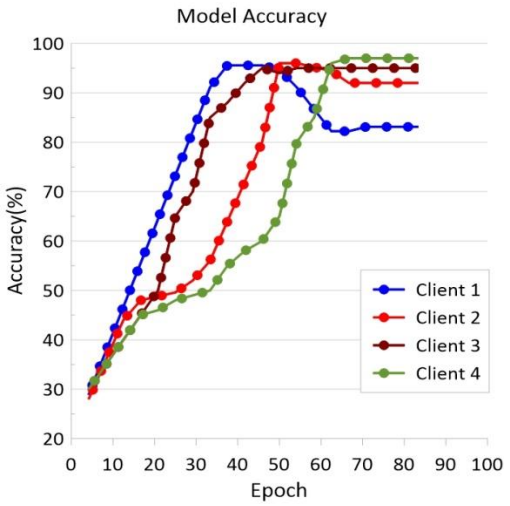

Figure 5. Clients' results during model training.

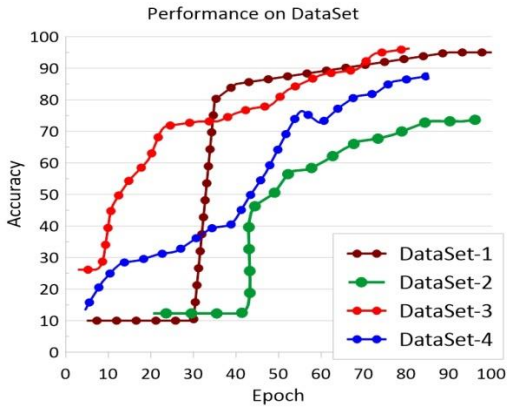

Figure 6. The performance of datasets on ResNet50 model.

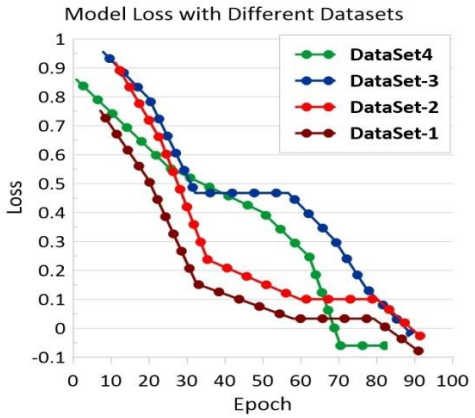

Figure 7. The loss of Datasets ResNet50 model.

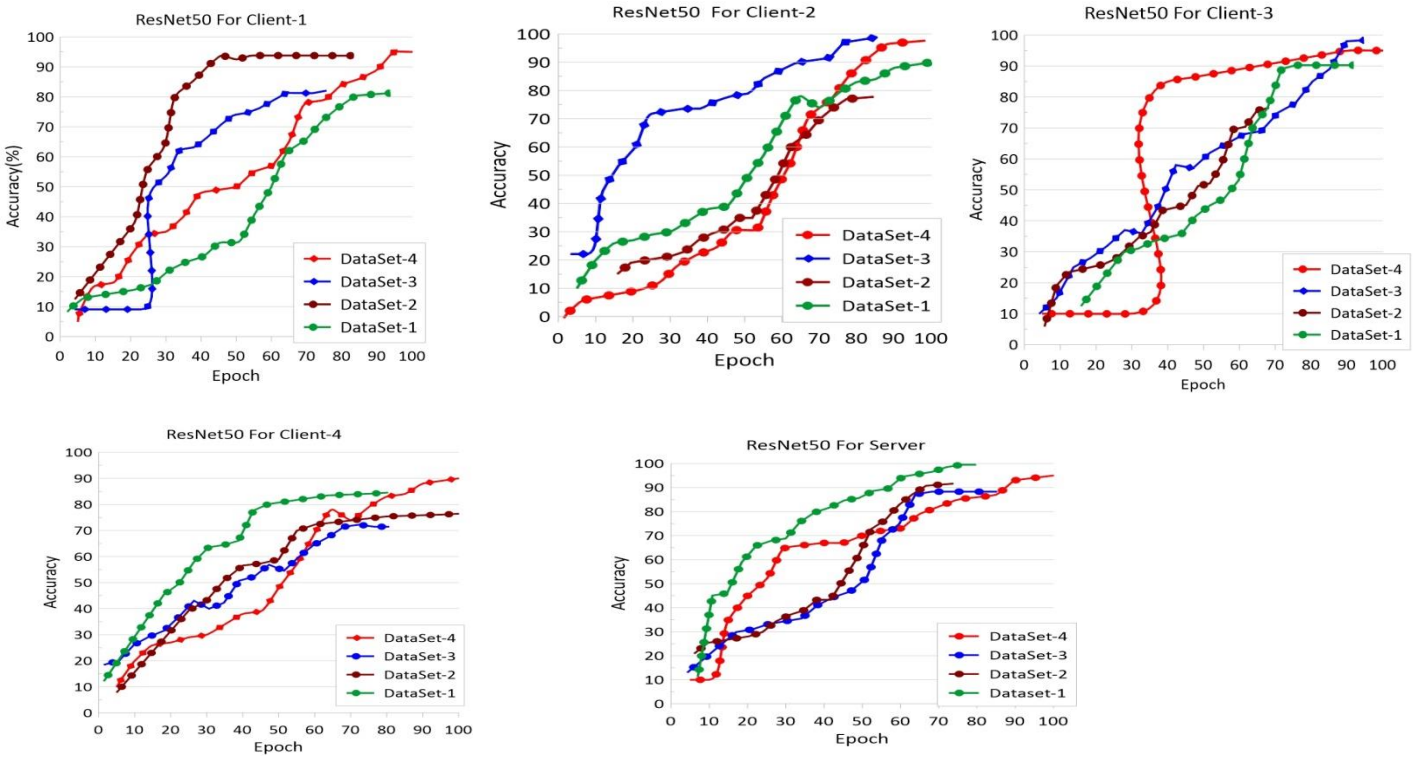

Figure 8. ResNet50 model's performance for clients and server using different datasets.

Client 4 was the most accurate client and server was the most accurate of all with a 99.32% accuracy. Accuracies in detecting viral pneumonia were 96.0, 97.67, 94.71, 97.05, 98.06 percent respectively. Fortunately, the server has the best results as in the previous classes. As described in Section 4.1.2, we used 4 datasets to train and evaluate the model. We described the details of dividing the data between the server and the client in the same section. We first train the model with the first dataset, then with the second one, and so on. Fig (6) shows the performance of different datasets in training the model. Datasets 1 and 4 has the best accuracy among the datasets used. Fig. (7) shows the loss due to training the model using selected datasets. The biggest loss is during using datasets 2 and 3. Fig (8) shows Resnet50 model's performance for different datasets in clients and server. Client 1 has the best accuracy when using Dataset 4. The accuracies of client 2 for datasets 1 and 4 are almost the same. Training client 2 with both of those datasets gives a desirable performance. Client 3 gives the best accuracy for Datasets 3 and 4. Client 4 reaches a desirable accuracy using datasets 2 and 4. Server reaches the desired performance using dataset 4. It is known than reaching the desired performance in deep learning methods requires lots of datasets. Our results clearly show that. Dataset 4 has the most samples, and shows the best performance in our experiments. Evaluation of the proposed method is done using two SGD and SWA optimizers in 1000 communication cylces. Accuracies were 99.58%, and 93.02% for SWA and SGD, respectively. Using SWA increased the Bayesian ensemble model's accuracy The follwoing diagram shows the result of this analysis.

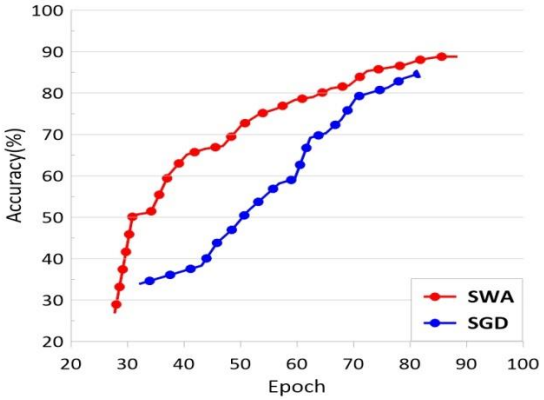

Figure 9. The results of using different optimizers in the proposed method.

Table 5. Evaluation results for the proposed method compared to the results of the previous approaches.

	Introduction			**DataSet**			
Author Name	**Method**	**Model**	**DataSet1**	**DataSet2**	**DataSet3**	**DataSet4**	
			Train Acc%	**Test Acc %**			
Yoon et al. [45]	Generalization and re-parameterization to solve the issue of non-convex federal	FedProx	73.05	83.90	57.96	82.90	78.06
Yoon et al. [19]	Federated reinforced Learning (MAFL)	FedMix	73.95	67.77	73.26	81.22	86.05
Shoham et al. [34]	Adding a penalty expression to loss function to control non-convexity	FedCurv	88.83	80.57	75.95	61.94	49.08
Collins et al. [18]	Personalized federated learning	FedRep	85.57	61.88	73.48	63.55	83.70
Karimireddy et al. [16]	Used customer similarity to reduce customer drift	SCAFFOLD	81.68	90.64	85.03	91.28	70.51
Li & Wong. [21]	Transfer learning and knowledge distillation	FEDMD	70.89	59.54	82.27	63.28	73.48
Proposed Method	Bayesian ensemble model and sampling	Our Model	89.56	91.04	95.58	92.54	96.78

4.3. Comparing our proposed method with the other methods

Our proposed method has a best accuracy of 96.78% which is the highest compared to FedMix, SCAFFOLD, and FedProx accuracies. The following diagram and table, showing the results of the execution, proves it. Table (5) shows the results of evaluation and comparing the proposed method with the previous methods. FedMix has the best test accuracy among the previous approaches. Our method tops all of the previous methods. This evaluation is done using 4 datasets of chest x-ray images, taken from several Covid-19 databases. The software programs and hardware equipment discussed in section 1.1.4 were the requirements of this evaluation.

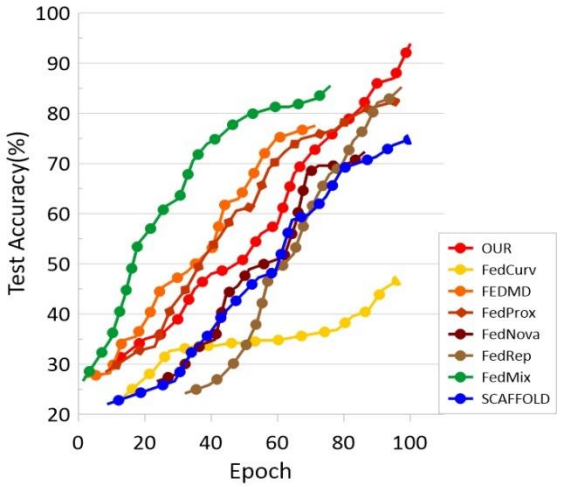

Figure 10. Results of using different optimizers in the proposed method.

5. CONCLUSION AND RECOMMENDATIONS

In federated learning, weight aggregation is the crucial step known as the most important cause of restrictions in implementing this novel approach. Needless to say, implementation will go forward without any problems if balanced data and basic CNNs are employed. Therefore, engaging with non-IDD data is the main issue. We recommend a new aggregated approach based on Bayesian inference and sampling, using which we managed to reach the desired accuracy in training the model with imbalanced data and deeper architectures such as ResNet-50. Using new strategies in our proposed approach, we hope to solve other complications in the newly developed federated learning (FL). In this regard, our new vision is to verify and enforce privacy protection as well as the principle of maintaining the model's balance through training non-IDD data and heterogeneous clients.

REFERENCES

[1] F. Haddadpour and M. Mahdavi, "On the convergence of local descent methods in federated learning," arXiv preprint arXiv:1910.14425, 2019.
[2] I. Dayan et al., "Federated learning for predicting clinical outcomes in patients with COVID-19," Nature medicine, vol. 27, no. 10, pp. 1735-1743, 2021.
[3] I. Feki, S. Ammar, Y. Kessentini, and K. Muhammad, "Federated learning for COVID-19 screening from Chest X-ray images," Applied Soft Computing, vol. 106, p. 107330, 2021.
[4] M. Mohri, G. Sivek, and A. T. Suresh, "Agnostic federated learning," in International Conference on Machine Learning, 2019: PMLR, pp. 4615-4625.
[5] P. Kairouz et al., "Advances and open problems in federated learning," Foundations and Trends® in Machine Learning, vol. 14, no. 1–2, pp. 1-210, 2021.
[6] F. E. Casado, D. Lema, M. F. Criado, R. Iglesias, C. V. Regueiro, and S. Barro, "Concept drift detection and adaptation for federated and continual learning," Multimedia Tools and Applications, vol. 81, no. 3, pp. 3397-3419, 2022.
[7] Y. Zhou, Q. Ye, and J. Lv, "Communication-efficient federated learning with compensated overlap-fedavg," IEEE Transactions on Parallel and Distributed Systems, vol. 33, no. 1, pp. 192-205, 2021.
[8] A. Mitra, R. Jaafar, G. J. Pappas, and H. Hassani, "Linear convergence in federated learning: Tackling client heterogeneity and sparse gradients," Advances in Neural Information Processing Systems, vol. 34, pp. 14606-14619, 2021.
[9] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of fedavg on non-iid data," arXiv preprint arXiv:1907.02189, 2019.
[10] A. Mitra, R. Jaafar, G. J. Pappas, and H. Hassani, "Achieving linear convergence in federated learning under objective and systems heterogeneity," arXiv preprint arXiv:2102.07053, 2021.
[11] S. Reddi et al., "Adaptive federated optimization," arXiv preprint arXiv:2003.00295, 2020.
[12] L. Liu and F. Zheng, "A bayesian federated learning framework with multivariate gaussian product," arXiv e-prints, p. arXiv: 2102.01936, 2021.
[13] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," Proceedings of Machine Learning and Systems, vol. 2, pp. 429-450, 2020.
[14] M. Yurochkin, M. Agarwal, S. Ghosh, K. Greenewald, N. Hoang, and Y. Khazaeni, "Bayesian nonparametric federated learning of neural networks," in International Conference on Machine Learning, 2019: PMLR, pp. 7252-7261.
[15] W. J. Maddox, P. Izmailov, T. Garipov, D. P. Vetrov, and A. G. Wilson, "A simple baseline for bayesian uncertainty in deep learning," Advances in Neural Information Processing Systems, vol. 32, 2019.
[16] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh, "Scaffold: Stochastic controlled averaging for federated learning," in International Conference on Machine Learning, 2020: PMLR, pp. 5132-5143.
[17] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in Artificial intelligence and statistics, 2017: PMLR, pp. 1273-1282.
[18] L. Collins, H. Hassani, A. Mokhtari, and S. Shakkottai, "Exploiting shared representations for personalized federated learning," in International Conference on Machine Learning, 2021: PMLR, pp. 2089-2099.

[19] T. Yoon, S. Shin, S. J. Hwang, and E. Yang, "Fedmix: Approximation of mixup under mean augmented federated learning," arXiv preprint arXiv:2107.00233, 2021.
[20] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated learning with non-iid data," arXiv preprint arXiv:1806.00582, 2018.
[21] D. Li and J. Wang, "Fedmd: Heterogenous federated learning via model distillation," arXiv preprint arXiv:1910.03581, 2019.
[22] A. Qayyum, K. Ahmad, M. A. Ahsan, A. Al-Fuqaha, and J. Qadir, "Collaborative federated learning for healthcare: Multi-modal covid-19 diagnosis at the edge," arXiv preprint arXiv:2101.07511, 2021.
[23] T. Lin, L. Kong, S. U. Stich, and M. Jaggi, "Ensemble distillation for robust model fusion in federated learning," Advances in Neural Information Processing Systems, vol. 33, pp. 2351-2363, 2020.
[24] L. Liu, F. Zheng, H. Chen, G.-J. Qi, H. Huang, and L. Shao, "A Bayesian Federated Learning Framework with Online Laplace Approximation," arXiv e-prints, p. arXiv: 2102.01936, 2021.
[25] Y. Guo, T. Lin, and X. Tang, "A New Analysis Framework for Federated Learning on Time-Evolving Heterogeneous Data."
[26] A. T. Thorgeirsson and F. Gauterin, "Probabilistic predictions with federated learning," Entropy, vol. 23, no. 1, p. 41, 2020.
[27] M. Al-Shedivat, J. Gillenwater, E. Xing, and A. Rostamizadeh, "Federated learning via posterior averaging: A new perspective and practical algorithms," arXiv preprint arXiv:2010.05273, 2020.
[28] M. Hutchinson, M. Reisser, and C. Louizos, "Federated Functional Variational Inference."
[29] X. Wu, S. S. Du, and R. Ward, "Global convergence of adaptive gradient methods for an over-parameterized neural network," arXiv preprint arXiv:1902.07111, 2019.
[30] X. Li and F. Orabona, "On the convergence of stochastic gradient descent with adaptive stepsizes," in The 22nd International Conference on Artificial Intelligence and Statistics, 2019: PMLR, pp. 983-992.
[31] B. E. Woodworth, K. K. Patel, and N. Srebro, "Minibatch vs local sgd for heterogeneous distributed learning," Advances in Neural Information Processing Systems, vol. 33, pp. 6281-6292, 2020.
[32] J. Wang, Q. Liu, H. Liang, G. Joshi, and H. V. Poor, "Tackling the objective inconsistency problem in heterogeneous federated optimization," Advances in neural information processing systems, vol. 33, pp. 7611-7623, 2020.
[33] S. Wang et al., "Adaptive federated learning in resource constrained edge computing systems," IEEE Journal on Selected Areas in Communications, vol. 37, no. 6, pp. 1205-1221, 2019.
[34] N. Shoham et al., "Overcoming forgetting in federated learning on non-iid data," arXiv preprint arXiv:1910.07796, 2019.
[35] C. Wu, F. Wu, L. Lyu, Y. Huang, and X. Xie, "Communication-efficient federated learning via knowledge distillation," Nature communications, vol. 13, no. 1, pp. 1-8, 2022.
[36] X. Li, Y. Gong, Y. Liang, and L.-e. Wang, "Personalized Federated Learning with Semisupervised Distillation," Security and Communication Networks, vol. 2021, 2021.
[37] H. Seo, J. Park, S. Oh, M. Bennis, and S.-L. Kim, "Federated knowledge distillation," arXiv preprint arXiv:2011.02367, 2020.
[38] E. Jeong, S. Oh, H. Kim, J. Park, M. Bennis, and S.-L. Kim, "Communication-efficient on-device machine learning: Federated distillation and augmentation under non-iid private data," arXiv preprint arXiv:1811.11479, 2018.
[39] Z. Zhu, J. Hong, and J. Zhou, "Data-free knowledge distillation for heterogeneous federated learning," in International Conference on Machine Learning, 2021: PMLR, pp. 12878-12889.
[40] P. Izmailov, D. Podoprikhin, T. Garipov, D. Vetrov, and A. G. Wilson, "Averaging weights leads to wider optima and better generalization," arXiv preprint arXiv:1803.05407, 2018.
[41] T. D. Pham, "Classification of COVID-19 chest X-rays with deep learning: new models or fine tuning?," Health Information Science and Systems, vol. 9, no. 1, pp. 1-11, 2021.
[42] M. K. Bohmrah and H. Kaur, "Classification of Covid-19 patients using efficient Fine-tuned Deep learning DenseNet Model," Global Transitions Proceedings, vol. 2, no. 2, pp. 476-483, 2021.
[43] https://www.kaggle.com/datasets/mushaxyz/covid19-customized-xray-dataset- (accessed 2022).
[44] "covid19-radiography-database." https://www.kaggle.com/tawsifurrahman/covid19-radiography-database (accessed (2022).
[45] X.-T. Yuan and P. Li, "On Convergence of FedProx: Local Dissimilarity Invariant Bounds, Non-smoothness and Beyond," arXiv preprint arXiv:2206.05187, 2022.

DOI: https://doi.org/10.15379/ijmst.v10i2.1421