FedVar : Federated Learning Algorithm with Weight Variation in Clients

Wooseok Shin
Dept. Of Electrical and Computer Eng.
Sungkyunkwan University
Suwon, Republic of Korea
swsda95@skku.edu

Jitae Shin*
Dept. Of Electrical and Computer Eng.
Sungkyunkwan University
Suwon, Republic of Korea
jtshin@skku.edu

2022 37th International Technical Conference on Circuits/Systems, Computers and Communications (ITC-CSCC) | 978-1-6654-8559-3/22/$31.00 ©2022 IEEE | DOI: 10.1109/ITC-CSCC55581.2022.9894899

Abstract— **Among the studies to solve statistical-related problems in federated learning (FL), there are many studies to improve the non-independent and identically distributed (Non-IID) problem of user data. This problem occurs because each local device collects data from different clients, so the size and characteristics of the data are different. When the data distribution of local devices is IID, it does not negatively affect learning, but when non-IID, it does not reach the general cloud computing performance. In this paper, we propose FedVar, an algorithm that uses the weight standard deviation for each client, to improve the situation in which the distribution of data is non-IID, so FL learning is not done properly. In addition, the performance is verified by comparing the acuity and loss of the proposed algorithm with other existing FL algorithms including FedAvg.**

Keywords: Federated Learning, Non-IID Data, Data Heterogeneity, FedSGD, FedAvg, FedVar

I. Introduction

In the existing machine learning, information obtained from devices such as mobile devices or sensors is collected into one central server and learning is carried out. However, as the number of mobile devices increases exponentially, no matter how many servers there are, a huge amount of computation is burdened. The proposed learning method is federated learning[1]. In federated learning, learning is carried out in each mobile device without sending the data each device has to the central server. The weights derived through learning are transmitted to the central server, and the central server is responsible for aggregating these weights into one weight. The process of lowering the weights collected from the central server back to the mobile device in the above method is called a 'round'. After one round, a new weight is created, and learning of each data is made using this weight. . By sending only weights, not local data, to the central server, learning becomes much safer than the existing learning method of uploading data to the server.

Federated Stochastic Gradient Descent(FedSGD) and Federated Averaging (FedAvg)[2] are proposed as representative algorithms for federated learning. SGD can be applied to the federated optimization problem in which gradient calculation for one batch (randomly selected client) in one communication round is performed. While this method is computationally efficient, it requires a lot of training rounds to get a good model. In general, this is set as a baseline in the CIFAR-10 environment. To apply the above approach to the federated setting, select clients with a C ratio for each round, and calculate the gradient loss for all data that these clients have. In this case, $0 \leq C \leq 1$. Therefore, C controls the global batch size, and if C=1, it means full-batch (non-stochastic) gradient descent. This baseline algorithm is FedSGD. On the other hand, in the case of FedAvg, parameters that have been updated from a mobile device are transmitted to a certain level, assuming a situation in which the mobile device can use the lowest network cost and efficiently. In addition, FedAvg is often used because the results of the two papers are not significantly different.

However, in this federated learning environment, the characteristics of the data are distinctly different from the existing deep learning environment. That is, the data are non-independent and non-identically distributed (Non-IID)[3]. Put it simply, someone's mobile device may have only food photos, while someone else's mobile phone may have only portrait photos. We can`t be sure if a good model will come out if I combine the model learned only from food photography and the model learned only from portrait photography. Also, someone's mobile device may not have 1GB of photos, while someone else's mobile device may have more than 50GB. This non-IID data distribution interferes with data processing in the central server, making the performance of federated learning deteriorate.

A typical federated learning protocol is as follows.[4] Since a large amount of computing and communication resources are required to learn an AI model, federated learning proceeds only when certain conditions such as learning available time and communication state are satisfied. A terminal that satisfies the condition first notifies the server that it is ready to register as a joint learning participant. The number of participants is tens to millions, and the server selects the optimal participant defined in the dictionary and then delivers information related to the task to be performed to each terminal. The terminal learns the local AI model based on the global AI model received from the server. The results of the local model that have been trained are delivered to the server, and the server updates the global AI model. This process corresponds to Round i in Fig.1. The main algorithms of federated learning include FedSGD and FedAvg and Algorithm 1. represents FedAvg algorithm.

In this paper, we propose FedVar (Federated Learning Algorithm with Weight Variation in Clients), which improves data processing in the central server by using the dispersion of weights due to the non-IID characteristics of mobile device data. By calculating the norm of the weight tensors to calculate the mean and variance, and by calculating a new weight average using weights within the calculated standard deviation range, clients with a special data distribution are excluded, and a universal federated learning model is completed. This improves the overall performance of the model. The performance of the proposed algorithm is verified by comparing it with the existing algorithms, FedSGD and FedAvg.

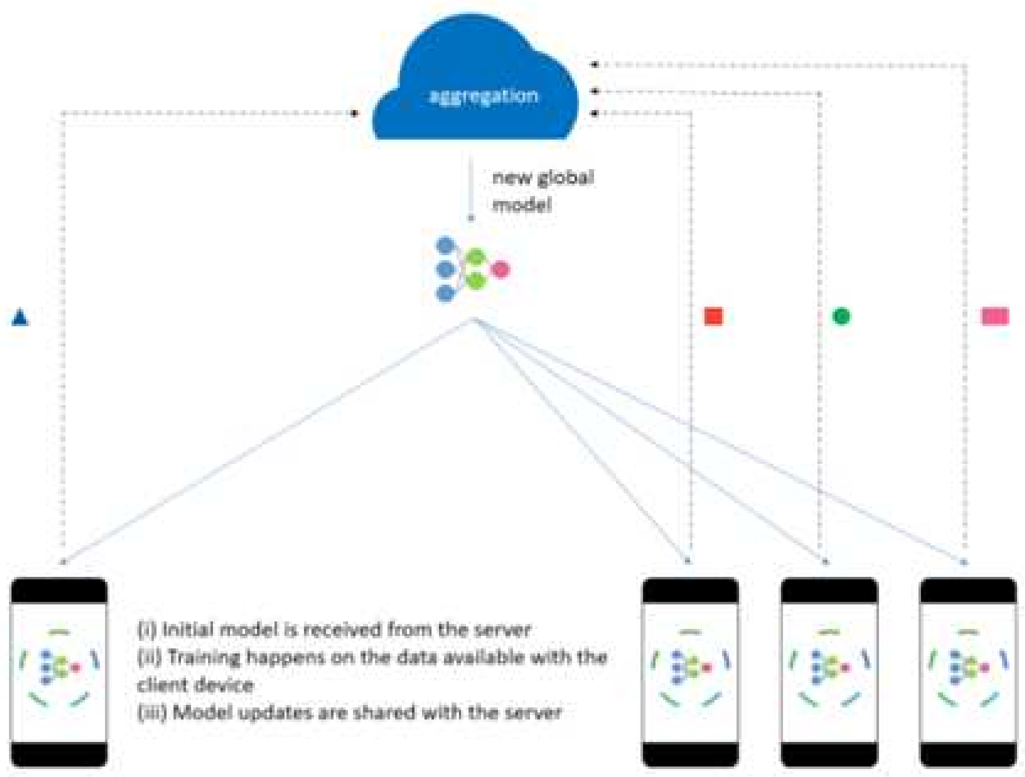

aggregation
new global model
(i) Initial model is received from the server
(ii) Training happens on the data available with the client device
(iii) Model updates are shared with the server

Fig. 1 Federated Learning Protocol

```
Server executes:
  initialize $w_0$
  for each round $t = 1, 2, \dots$ do
    $m \leftarrow \max(C \cdot K, 1)$
    $S_t \leftarrow$ (random set of $m$ clients)
    for each client $k \in S_t$ in parallel do
      $w_{t+1}^k \leftarrow$ ClientUpdate$(k, w_t)$
    $w_{t+1} \leftarrow \sum_{k=1}^{K} \frac{n_k}{n} w_{t+1}^k$

ClientUpdate$(k, w)$:   // Run on client $k$
  $\mathcal{B} \leftarrow$ (split $\mathcal{P}_k$ into batches of size $B$)
  for each local epoch $i$ from 1 to $E$ do
    for batch $b \in \mathcal{B}$ do
      $w \leftarrow w - \eta \nabla \ell(w; b)$
  return $w$ to server
```

Algorithm 1. FedAvg Algorithm

II. Proposed Algorithm

In this paper, as shown in Figure 2, we propose the FedVar algorithm that uses the variance calculated by id entifying the weight distribution to improve the performance in terms of accuracy through the improvement of data processing performed in the FL server. The model parameters w1 to wk mean weights learned from each client and transmitted to the FL server through the uplink. In the existing FedSGD and FedAvg, these data are averaged on the FL server and the model parameter SDA optimized through averaging is delivered to all clients through the downlink. can be used, but model performance may decrease in the Non-IID state. The biggest difference between FedVar and the existing FL method to improve this is that it utilizes a weight distribution for each user. Before transmitting the weights from the server to the mobile device, only the weights within the average to standard deviation range are averaged and transmitted to the mobile device. This allows you to exclude features that cause covariate shifts when the data distribution is not IID. The detailed algorithm is the same as Algorithm 2.

Assuming we compute the average value avg of the clients and the standard deviation std of the clients, only clients greater than (avg - std) or less than (avg + std) will participate in the next round. This process completes a universal federated learning model for multiple clients, excluding clients with special data distributions, which improves overall model performance indicators.

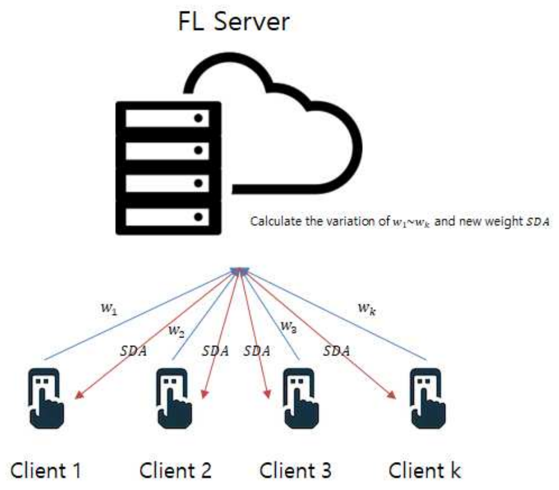
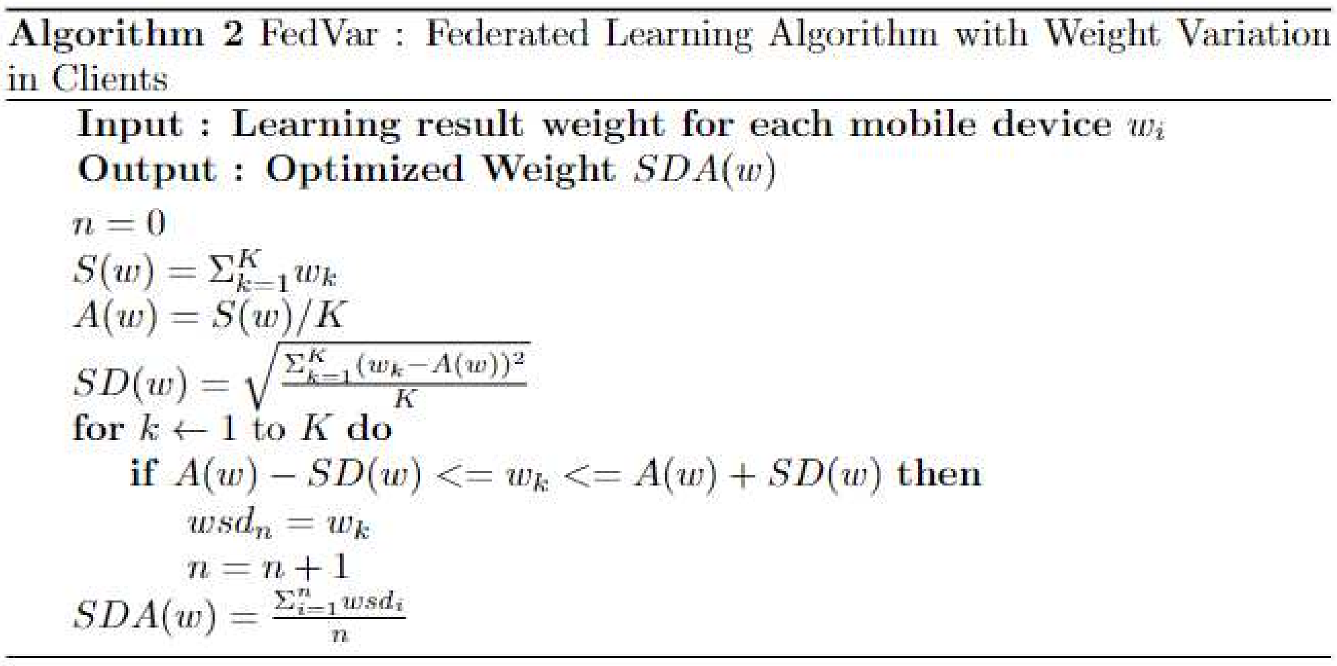


Fig. 2 FedVar



Algorithm 2 FedVar : Federated Learning Algorithm with Weight Variation in Clients

```
Input : Learning result weight for each mobile device $w_i$
Output : Optimized Weight $SDA(w)$
$n = 0$
$S(w) = \Sigma_{k=1}^{K} w_k$
$A(w) = S(w)/K$
$SD(w) = \sqrt{\frac{\Sigma_{k=1}^{K}(w_k - A(w))^2}{K}}$
for $k \leftarrow 1$ to $K$ do
    if $A(w) - SD(w) <= w_k <= A(w) + SD(w)$ then
        $wsd_n = w_k$
        $n = n + 1$
$SDA(w) = \frac{\Sigma_{i=1}^{n} wsd_i}{n}$
```

Algorithm 2. Proposed FedVar Algorithm

III. Experimental Results

In order to verify the performance of the proposed method, an open source that can check the performance of several previously proposed FL algorithms was used. The open source implemented in https://github.com/c-gabri/Federated-Learning-PyTorch.git was used, and the FedVar algorithm was implemented by modifying the FedAvg and FedSGD codes provided by the existing open sources. Open source uses TinyNet[5], GhostNet[6], and MobileNetV3[7] as models, and CIFAR-10, CIFAR-100, and MNIST are provided as datasets.

The number of clients was set to 100, the batch size was set to 50, and the local epoch was set to 5. The size of the random set of clients was set to 10. And the total round was set to 200, and global accuracy performance was measured every 20 rounds.

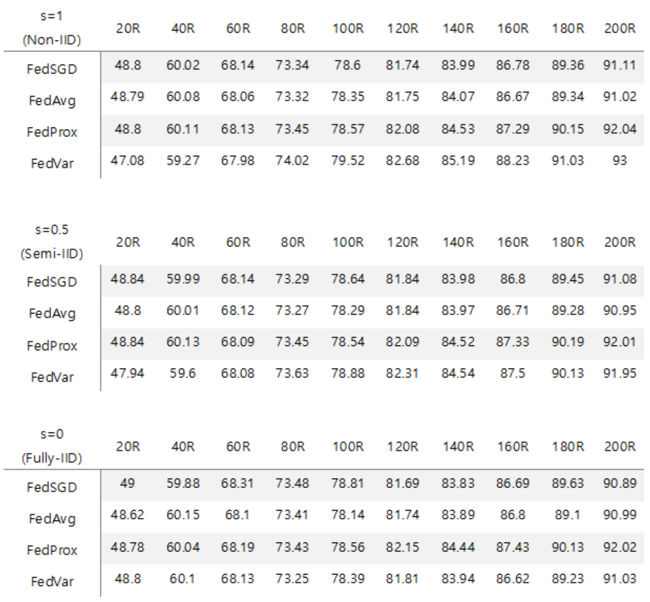
Table 1 shows that FedVar's global accuracy performance improved compared to FedAvg and FedSGD after 40 calls. The reason FedAvg and FedSGD performed better before the 40th communication is presumably because FedVar did not properly find the covariate clients until the 40th round. It also shows that performance is not compromised compared to FedProx, one of the most used algorithms today.

Experiments were also performed according to the heterogeneity of the data. The value of s indicates the degree of heterogeneity. When the value of s is 1, it is completely non-IID, and as the value of s decreases, it becomes closer to fully-IID. All three existing algorithms (FedSGD, FedAvg, FedProx) showed similar performance when the data were fully-IID/semi-IID/non-IID, because none of the three algorithms showed the non-IID nature of the data. This is mainly because it did not solve the problem. However, since FedVar is a specialized algorithm for solving non-IID characteristics of data, it can be seen that the performance is very different in the three cases. In the case of Full-IID, there is no performance difference between FedSGD and FedAvg, and the performance is worse than FedProx. However, it outperforms FedSGD and FedAvg in Semi-IID and outperforms FedProx in Non-IID. It can be seen that the greater the heterogeneity, the better the FedVar's performance.

IV. Conclusion

In this paper, we propose the FedVar algorithm to solve the non-IID problem of data distribution, which is one of the representative problems of federated learning. Using the proposed algorithm, if the data distribution is not IID, those that cause client covariate shifts can be effectively excluded

Table. 1 Experimental Results



s=1 (Non-IID)	20R	40R	60R	80R	100R	120R	140R	160R	180R	200R
FedSGD	48.8	60.02	68.14	73.34	78.6	81.74	83.99	86.78	89.36	91.11
FedAvg	48.79	60.08	68.06	73.32	78.35	81.75	84.07	86.67	89.34	91.02
FedProx	48.8	60.11	68.13	73.45	78.57	82.08	84.53	87.29	90.15	92.04
FedVar	47.08	59.27	67.98	74.02	79.52	82.68	85.19	88.23	91.03	93

s=0.5 (Semi-IID)	20R	40R	60R	80R	100R	120R	140R	160R	180R	200R
FedSGD	48.84	59.99	68.14	73.29	78.64	81.84	83.98	86.8	89.45	91.08
FedAvg	48.8	60.01	68.12	73.27	78.29	81.84	83.97	86.71	89.28	90.95
FedProx	48.84	60.13	68.09	73.45	78.54	82.09	84.52	87.33	90.19	92.01
FedVar	47.94	59.6	68.08	73.63	78.88	82.31	84.54	87.5	90.13	91.95

s=0 (Fully-IID)	20R	40R	60R	80R	100R	120R	140R	160R	180R	200R
FedSGD	49	59.88	68.31	73.48	78.81	81.69	83.83	86.69	89.63	90.89
FedAvg	48.62	60.15	68.1	73.41	78.14	81.74	83.89	86.8	89.1	90.99
FedProx	48.78	60.04	68.19	73.43	78.56	82.15	84.44	87.43	90.13	92.02
FedVar	48.8	60.1	68.13	73.25	78.39	81.81	83.94	86.62	89.23	91.03

from training. Even compared to the popular FedProx algorithm, it has been proven that there is no drop in performance except when Fully-IID. Rather, it was confirmed that it has better performance in a completely non-IID data distribution.

However, the current FedVar algorithm does not work well when the data distribution is multimodal and excludes clients that cause covariate shifts during the entire training period, so while the average accuracy is high, there is a problem that the accuracy of each of the clients is large.

Future research suggests a way to modify the FedVar algorithm when the data distribution is multimodal, possibly with an algorithm that can restrictively allow clients to engage in training, causing significant covariate shifts.

REFERENCES

[1] J Konečný et al, "Federated Learning: Strategies for Improving Communication Efficiency", NIPS Workshop on Private Multi-Party Machine Learning, 2016

[2] B.McMahan et al, "Communication-efficient learning of deep networks from decentralized data," in Artificial Intelligence and Statistics. PMLR, 2017, pp. 1273- 1282

[3] Y Zhao et al. 2018. Federated Learning with Non-IID Data. arxiv:cs.LG/1806.00582

[4] K. Bonawitz, et. al., "Towards Federated Learning at Scale: System Design", MLSys,2019

[5] K. Han et al, "GhostNet: More Features from Cheap Operations", CVPR,2020

[6] K. Han et al, "Model Rubik's Cube: Twisting Resolution, Depth and Width for TinyNets", NeurlPS, 2020

[7] A. Howard et al, "Searching for MobileNetV3", ICCV,2019, pp1314-1324

[8] H.Zhu et al, "Federated learning on non-IID data: A survey", Neurocomputing, Volume 465, 20 November 2021, pp 371-390

[9] K. Hsieh et al, "The Non-IID Data Quagmire of Decentralized Machine Learning", PMLR, 2020

[10] T. Hsu et al. "Federated Visual Classification with Real-World Data Distribution", ECCV, 2020