applied sciences

Article

Weighted Averaging Federated Learning Based on Example Forgetting Events in Label Imbalanced Non-IID

Mannsoo Hong [1], Seok-Kyu Kang [1] and Jee-Hyong Lee [2,*]

[1] Department of Artificial Intelligence, Sungkyunkwan University, Suwon 16419, Korea; msjo91@skku.edu (M.H.); kittyhyen@skku.edu (S.-K.K.)
[2] Department of Computer Science and Engineering, Sungkyunkwan University, Suwon 16419, Korea
* Correspondence: john@skku.edu

Abstract: Federated learning, a data privacy-focused distributed learning method, trains a model by aggregating local knowledge from clients. Each client collects and utilizes its own local dataset to train a local model. Local models in the connected federated learning network are uploaded to the server. In the server, local models are aggregated into a global model. During the process, no local data is transmitted in or out of any client. This procedure may protect data privacy; however, federated learning has a worse case of example forgetting problem than centralized learning. The problem manifests in lower performance in testing. We propose federated weighted averaging (FedWAvg). FedWAvg identifies forgettable examples in each client and utilizes that information to rebalance local models via weighting. By weighting clients with more forgettable examples, such clients are better represented and global models can acquire more knowledge from normally neglected clients. FedWAvg diminishes the example forgetting problem and achieve better performance. Our experiments on SVHN and CIFAR-10 datasets demonstrate that our proposed method gets improved performance compared to existing federated learning algorithm in non-IID settings, and that our proposed method can palliate the example forgetting problem.

Keywords: federated learning; example forgetting event; weighted averaging

check for updates

Citation: Hong, M.; Kang, S.-K.; Lee, J.-H. Weighted Averaging Federated Learning Based on Example Forgetting Events in Label Imbalanced Non-IID. *Appl. Sci.* **2022**, *12*, 5806. https://doi.org/10.3390/app12125806

Academic Editor: Arcangelo Castiglione

Received: 2 May 2022
Accepted: 6 June 2022
Published: 7 June 2022

1. Introduction

In general, deep learning training runs a single centralized model on a large, integrated, and well-balanced dataset. However, in real life, collecting high quality data is an expensive labor because data has become more decentralized and personalized. The rising demand for data privacy precludes certain data from collection. Those uncollectible data cherish rich knowledge. Federated learning is a methodology to access and utilize that knowledge. In a federated learning network, a client is the collector, owner, and user of its data, whereas the server has no access privilege to a client's data. Since there is only so much a single client may collect and store, an allied effort is demanded for server and clients to train a viable global model [1–5].

A standard training process of federated learning begins with the server distributing copies of an initialized global model to participating clients. Clients train the copied local models with their own local data. Trained local models are transferred to the server. The server aggregates the local model parameters to update the global model parameters. This indirect approach to decentralized data destabilizes training, lowering performance compared to the traditional centralized deep learning, that is a single machine directly accessing data to train a centralized model.

To address the inherent issues of federated learning, we focused on the concept of example forgetting event, a phenomenon inspired by catastrophic forgetting [6–8]. Catastrophic forgetting occurs in a continual learning environment when a model fails a task previously learned after learning a new task. If a model is to learn two tasks consecutively, first task A then task B, it is known that the model's performance on task A will drop after learning task B. On the other hand, an example forgetting event happens when a model loses its previously acquired knowledge on a data point during training. While a model is training on data batches, the model will successfully predict data x on a certain point. However, after training on some more batches, the prediction on x will fail.

We presumed that a similar information loss phenomenon will occur for federated learning. For a federated learning environment, we defined forgettable examples as those in which a forgetting event occurs immediately after the global model update. In other words, the local model will succeed in predicting these data before aggregation but global model after aggregation will not. These data are forgotten. Other data that are not forgotten are unforgettable examples. Via various experiments, we discovered that forgetting events occur more frequently in a federated learning environment than a typical centralized learning environment (single central model training).

Based on these discoveries, we suggest a novel method for federated learning, FedWAvg: federated weighted averaging by example forgetting events. Our method reorganizes the weight of each client according to the number of forgettable examples in each round. By implementing weighted averaging as an aggregation method, global models are able to show higher performance in highly unbalanced distribution of decentralized data.

2. Development of Federated Learning

In the early stages of deep learning, the training a model for a task generally occurred as the following scenario. Researchers collect a substantial amount of samples to build a rich, integrated, and well-balanced dataset. They design a single centralized model that will interpret the samples and return results in accordance to the intended task. The researchers make modifications after analyzing the results. They could gather more samples, clean out the dataset, tune hyperparameters, or test different neural network architectures. This has been the case in many environments even outside of the laboratories. Now that deep learning-based artificial intelligence has achieved a certain level, the recent trend is to make practical and industrial use out of it. Researching and developing a deep learning model is one thing; however, commercializing is another. The application of artificial intelligence is facing numerous issues from not only engineering but also law and cost.

In real life, collecting data for a specific target task is highly expensive. Even so, there are only so much data any party could have control over because of privacy issues and corresponding legal disputes. Moreover, with modern communication technology, raw data is becoming more decentralized and personalized. In blatant terms, data collection has become a pricey labor with only restricted access privilege; thus, valuable knowledge is being neglected. Federated learning was proposed henceforth.

Federated learning is a method of training a global model via aggregating multiple local models. In a federated learning network, multiple nodes communicate with each other to share knowledge without transmitting the training data. Usually a central node, a server owned by the developers, will update the global model. Other nodes, machines of application/service users, will have their own local datasets and train local models. These nodes are called clients.

The goal of federated learning is to achieve collective knowledge whilst protecting data privacy in a distributed data environment. That is, the main feature of federated learning is to amalgamate decentralized or shattered knowledge without transmitting data. Since sending data to a server or another client has a chance of violating data privacy, data are strictly utilized within the client; hence, the model aggregation. Transmitting local models instead of data allows the protection of data privacy. Therefore, researchers are able to use a larger scope of sensitive data, indirectly. Since the advantage of utilizing normally inaccessible data is highly beneficial in data driven industry, various studies are being actively conducted [6,7,9].

FedAvg [1] defines the standard training process of modern horizontal federated learning [10]. It communicates model parameters between servers and clients. Local model parameters are aggregated via an averaging algorithm. The averaged parameters become the global model parameters. Since FedAvg requires averaging corresponding local model parameters and updating global model parameters, it is vital that every node shares the same neural network architecture. In doing so, we can find the equivalent parameters by position to average or overwrite. FedAvg mainly focuses on the independently and identically-distributed data environment. As it has already shown stable and suitable performance, later research alters FedAvg to adapt in harsher non-IID environments.

FedProx [11] introduces proximal regularization terms to limit local updates and keep the local model close to the global model. In other words, the proximal regularization term reduces divergence of local models from the global model. FedProx tends to converge faster in the early stage; however, the later stage performance is nearly the same as FedAvg. SCAFFOLD [12] is an algorithm that attempts to solve the client drift problem. SCAFFOLD finds global convergence by using a controlled variate for each client. FedNova [13] is a normalized averaging method that eliminates objective inconsistency while preserving fast error convergence. FedNova is complementary to gradient compression or quantization. FedMA [14] takes an arduous approach. It continues to freeze layers one by one. It builds up a common ground for every node and slowly reduces divergence by freezing parameters. The development of FedAvg-based federated learning algorithms has been towards lowering divergence of clients in non-IID. However, all these methods either result in extremely slower convergence or require increased costs in computing and communication.

Recently, the field of federated learning has branched out to two new frontiers. The first branch is personalizing. It embraces the divergence to a certain level instead of eliminating it. Recent methods such as FedBN [15] and FedBABU [16] use certain layers as guidelines. FedBN excludes batch normalization layers from aggregation. Each client keeps personalized statistics to normalize input data. FedBABU separates neural network architecture to the body and head. Only the body, the feature extractor, is updated whilst the head, classifier, is frozen. The body will update in accordance to the fixed head. It also uses fine-tuning after aggregation. Both methods achieve high performance in personalized settings. However, if a personalized local model is evaluated in another client's personalized setting, it would falter. Moreover, these methods operate in a specific non-IID setting, FedBN using different datasets for each client and one client per dataset, and FedBABU using the old non-IID proposed in FedAvg [1]. The second branch is replacing the averaging algorithm with knowledge distillation [17]. FedDF [18] and FedGen [19] both utilize knowledge distillation as an aggregation technique instead of averaging algorithm. The replacement allows each client to have a neural network that is specialized to its hardware properties. Although knowledge distillation requires a certain number of data collected in the server, ablation studies show that it can be replaced by randomly generated inputs. Still, the ablation studies also note that the performance is enhanced when the distribution of inputs for knowledge distillation resembles that of the local training datasets [18].

In summary, the development of federated learning is confronted on four fronts: how to aggregate knowledge; should generalize or personalize, how to reduce drift or divergence, and which non-IID to handle. As such, solving a federated learning problem is accompanied by the risk of having the solution being less effective in other federated learning situations.

3. Example Forgetting in Federated Learning

A learning event occurs to a data sample when a neural network can successfully perform the intended task on the sample for the first time. Once it is learned, our assumption is that the model will be able to keep the knowledge. However, this is not the case. A forgetting event may rise. A forgetting event is the contrary to a learning event. Once a neural network loses the acquired knowledge of a trained sample, it will fail the task on it [6–8]. This phenomenon has been an issue for some time and is well known. However, this issue has not been dealt with in federated learning.

3.1. Example Forgetting of FedAvg

To verify the issue exists in federated learning, example forgetting events in federated learning and centralized learning are compared. We measure example forgetting events in centralized learning as follows. We randomly batch the training data. During batch training, a batch of data A, is fed forward to check if a model can correctly predict the target. The margin error between the target and prediction is calculated and saved for future reference. After the model is updated by another batch B, we once again feed forward batch A. If the model predicted correctly for batch A previously but incorrectly after the batch B update, or if the new margin error is larger than the previous, a forgetting event has occurred.

Since personalized approaches have more than one model to track, FedAvg is chosen. However, a global model is never trained but overwritten in FedAvg. Therefore, we define example forgetting anew. An example forgetting event in federated learning is that a global model losing knowledge of a data sample which was previously learned to a local model before aggregation. Therefore, finding a forgetting event is different from centralized model training. We evaluate the local training dataset on the local model before communication then evaluate the same dataset on the global model. If an example is successfully classified by a local model but misclassified by a global model, an example forgetting event has occurred. When a global model is aggregated, the knowledge learned in each local model is aggregated as well, resulting in an inevitable information loss. Hence, the newly distributed local models are more likely to forget trained examples. We compared forgetting events in federated learning to centralized learning. We recorded the number of forgetting events for MNIST [20], FMNIST [21], SVHN [22], and CIFAR-10 [23] with DenseNet-121 model [24] as shown in Table 1. In federated learning, compared to centralized learning, forgetting events increased by 1989% in MNIST, 472% in FMNIST, 166% in SVHN, and 271% in CIFAR-10. In particular, there was a huge difference in MNIST. The reason is that in the case of the MNIST dataset, which is relatively easy to classify compared to other datasets, example forgetting rarely occurs during centralized learning. In the case of federated learning, the redistributed global model shows drastic forgetting events. The phenomenon seems proportional to the classification difficulty of the dataset. For fair comparison, all hyperparameters are fixed, but 200 epochs were given in centralized learning, and 40 rounds of 5 epochs are given to each of the 5 clients in federated learning, so that the total epochs for each client are set to be the same as the epochs of centralized learning.

Table 1. Occurrence of example forgetting events.

	MNIST	**FMNIST**	**SVHN**	**CIFAR-10**
Centralized	4044	22,951	84,748	57,717
Federated	80,437	108,415	140,984	156,426
Increase (%)	1989	427	166	271

3.2. Definition of Non-IID

The idea of non-IID in federated learning may seem unorthodox, not the situation itself but the reason it should be considered. One of main considerations in federated learning is data privacy [1,2,5]. No node should have access to another node's data repository nor should it openly share its data to others. This not only distinguishes federated learning from parallel and distributed learning, it complicates many aspects of it. For example, federated learning aggregates local models because it cannot train a single centralized model directly from local datasets. Clients having computationally weak hardware would be less of a problem for we could simply transmit the data to an advanced hardware. A non-IID situation would be easily solved if the centralized control node could receive the local data distribution from each client. The problem of training on biased or skewed data could be rectified once it has the privilege to access and modify other nodes' data.

Considering all the obstacles from data privacy, one could even argue that federated learning is business-oriented in nature.

The academic definition of non-IID is simply if connected clients have varying data distribution [1,25]. Assume there is a dataset $\{x, y\}$. For a client k, the local data distribution would be $P_k(x, y)$. For any other connected client j, the federated learning network is non-IID if $P_k(x, y) \neq P_j(x, y)$. Some connected clients in the network may have the same local data distribution; however, as long as there are clients with different distributions, we can say it is still non-IID. Defining non-IID for a federated learning problem is discussing if it abstractly simulates or simplifies a plausible real life situation. Therefore, a non-IID should be convincing.

The non-IID in our problem is a cluster-level label size imbalance. That is, clients in the same cluster share the same labels while each cluster has non-overlapping labels as shown in Figure 1. To simplify the matter, the number of total local data samples is the same for all clients. Such situations could occur due to location variations. If we are to classify flower images, flower image data collected in one area would differ from those in another area. Therefore, we could cluster nodes based on the data source location. Another possible case would be diagnosing genetic diseases. Some genetic pools would be more susceptible to some diseases.

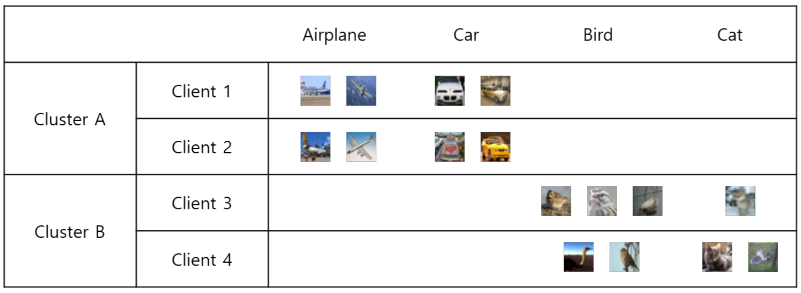

Figure 1. ResNet-based network architecture used in experiment. An illustrative example of label-size imbalanced non-IID. The clients in the first cluster A only has airplane and car data. The clients in the second cluster B only has bird and cat data. The number of each class samples may vary for each client, however, the total number samples are the same.

3.3. *Example Forgetting in Non-IID*

In considering federated learning, one can never dismiss non-IID. As mentioned in Section 3.2, non-IID implemented is a cluster-level label size imbalance. The clients are split into two clusters. Clients of a cluster have the same classes, and each cluster has different classes. Each client has the same amount of data in total. However, the number of clients in each cluster are disproportionate so that certain knowledge would be harder to acquire. Note that the nodes, both server and clients, do not know which cluster each client is in or even if such clusters exist because they do not have privilege to access another node's data. The dataset used is CIFAR-10. CIFAR-10 is an RGB-colored 32 × 32 image dataset of 10 classes. Each cluster is given two randomly chosen classes of data. Each client contains 1200 non-overlapping samples distributed via uniform random sampling.

A shallower variant of ResNet [26] is implemented as the backbone model for FedAvg. It has four residual blocks followed by a global average pooling layer and a single fully connected layer. Each residual block is consisted of two sets of a 3 × 3 convolutional layer and a batch normalization layer. There are three shortcut connections made with a 1 × 1 convolutional layer and a batch normalization layer. Training proceeds normally as FedAvg would as in Section 3.1, except iterating 100 rounds instead. At the end of each round, the number of forgettable examples are recorded.

Table 2 has two clients for the first minor cluster and eight clients for the second majority cluster. The records of forgettable examples, data samples which forgetting event occurred, demonstrate that FedAvg highly favors the majority cluster. The reason for this is that the classes in the majority cluster have a greater chance of being represented. It is easily assumable that such a tendency would be mitigated once the clusters are more balanced. However, the following result begs to differ. Table 3 consists of four minority clients and six majority clients. The records show that the overall tendency of learning is almost the same as the 2-to-8 situation in Table 2. According to the results, the majority clients will always be favored. While they achieve near zero forgettable examples, the minority clients have almost no advance.

Table 2. Forgettable example counts in each client when non-IID (2-to-8).

Round	Minor1	Minor2	Major1	Major2	Major3	Major4	Major5	Major6	Major7	Major8
10	1137	1148	55	73	90	84	103	93	60	79
20	1180	1185	63	38	38	57	66	57	47	60
30	1192	1197	34	22	39	32	45	36	28	39
40	1195	1189	22	12	24	25	29	27	23	22
50	1194	1196	21	8	22	17	27	24	19	22
100	1197	1129	0	1	0	1	2	0	3	5

Table 3. Forgettable example counts in each client when non-IID (4-to-6).

Round	Minor1	Minor2	Minor3	Minor4	Major1	Major2	Major3	Major4	Major5	Major6
10	1188	1151	1180	1178	285	295	308	312	286	292
20	1191	1182	1182	1184	86	87	108	103	77	93
30	1194	1181	1191	1194	21	23	25	27	22	22
40	1186	1193	1181	1194	4	5	9	10	3	9
50	1189	1173	1183	1192	0	2	3	2	0	3
100	1121	1114	1130	1120	0	0	0	0	0	1

4. Proposed Method

As mentioned in the previous Section 2, we also make choices to define the federated learning problem to solve. In our case, we propose a method to build a federated averaged model in a label-skewed non-IID [25]. The primary focus of our proposed method is to modify the aggregation method to consider relativity among clients based on the difficulty of learning local data samples. Meanwhile, data samples are not leaked and only utilized within the possessing client. Our definition of difficulty to learn derives from the fact that some data samples tend to be forgotten during aggregation. Our proposed method applies a hyperparameter to control the change of global model parameters to avoid extreme divergence.

As mentioned in the previous Section 3, an example forgetting problem is more concerning in federated learning than in centralized learning environment. We also observed when there is an unbalanced distribution of data among clients, the averaging aggregation would always favor the majority.

We propose federated weighted averaging (FedWAvg), which rebalances the global model aggregation based on forgettable examples in clients. FedWAvg ensures the global model is more influenced by knowledge that would normally be neglected in averaging aggregation in a non-IID environment. It manipulates the weight of each client strengthening those with more forgettable examples.

The goal of this method is to reduce information loss by model aggregation. The observations in Section 3 prove that the averaging algorithm is skewed towards the majority cluster. The resulting global model will forget most of the information from the minority cluster. This is shown in Tables 2 and 3. Forgetting nearly 90% of the minority (cluster) data allows us to assume that the global model will fail on the intended task if the inferred input is from the minority classes. Thus, FedWAvg aims to reduce the number of forgettable examples—training data samples that are learned then forgotten—after aggregation, which proves that information loss induced by model aggregation is mitigated.

To achieve the above goal, FedWAvg counts forgettable examples every t rounds. The clients send the number of forgettable examples to the server alongside with the local model parameters. Since the number is a single integer, it would barely add to the size of payload. In addition, because the number does not reveal anything about the raw training data or its distribution, it is safe to be transmitted. The number of forgettable examples in each client means how much information is lost in each client. A client with a high number of forgettable examples means the local model did not contribute much to the global model. Therefore, rebalancing clients inverse to the number of forgettable examples would assure that less influential clients would have more influence, and vice versa. In our case, we used simple normalization, dividing numbers of forgettable examples by the sum of them as in Equation (1).

$$norm(X) = \frac{X}{\sum X}, \tag{1}$$

However, applying normalization directly is not recommended because the weights, made by normalizing the numbers of forgettable examples, could drastically change local model parameters. To alleviate radical changes, we added updated ratio hyperparameter α. The ratio α decides how much change the weights will have. If α is small, then there will be less change.

$$(1-\alpha) + \alpha(norm(X) \times N) \quad \text{where } \ 0 \leq \alpha < 1, \tag{2}$$

The N in Equation (2) is the number of clients. Multiplying the weights with the local model parameters provides a rebalanced set of local models. This could be applied to any averaging algorithm-based federated learning methods. In case of FedAvg, the recalculated local model parameters will be averaged; therefore, executing weighted averaging. The complete process is explained in Algorithm 1.

Algorithm 1 Federated weighted averaging by example forgetting

```
Input:
Training data $\{X, Y\}$; neural network $f$; neural network parameter $\theta$;
global server $g$; number of clients $N$; local epoch $E$;
learning rate $\eta$; event period $t$; update ratio $\alpha$
Output:
Global model parameter $\theta_g$
1:  procedure FEDERATED WEIGHTED AVERAGING
2:      initialize $\theta_g$; $F \leftarrow$ list of $N$ ones; $W \leftarrow$ list of $N$ ones
3:      while not training done do
4:          for $n = 1$ to $N$ in parallel do                        ▷ Clients execute in parallel
5:              $\hat{Y} \leftarrow f_\theta(X)$; $\hat{Y}_g \leftarrow f_{\theta_g}(X)$
6:              $\theta \leftarrow \theta_g$
7:              for $e = 1$ to $E$ do
8:                  $\theta \leftarrow \theta - \eta \nabla L(\theta; X)$                    ▷ Update local model
9:              end for
10:             if current round $\bmod t == 0$ then
11:                 $F[n] \leftarrow count(\hat{Y} := Y \&\& \hat{Y}_g \neq Y)$    ▷ Upload forgettable example count
12:             end if
13:         end for
14:         $W \leftarrow (1-\alpha) + \alpha(norm(F) \times N)$   ▷ Calculate averaging weights (Equation (2))
15:         $\theta_g \leftarrow \frac{1}{N} \sum_{n=1}^{N} \theta_n \times W[n]$                 ▷ Update global model
16:     end while
17:     return $\theta_g$
18: end procedure
```

5. Experiments and Results

Two experiments are conducted. The first experiment is conducted on SVHN [22] and CIFAR-10 [23] datasets for an image classification task in a simulated federated learning environment. The environment is a cluster consisted of virtual nodes. A single central node that acts as the central server is connected to all other nodes, which are the local clients. Each client has a connection only to the server. Accessing another client is prohibited. The federated learning environment consists of nine clients. The training dataset is split in two to form a non-IID settings for biased client cluster. Clients within the cluster share the same classes but no class is shared among clusters. The first setting has two minority clients and seven majority clients, and the second setting has two, three, and four clients for each three clusters. For simplicity, we assumed every client is utilized in every communication round and contributes to updating the global model. The same shallower modification of ResNet in Section 3.3 is used as a backbone model f with an Adam optimizer. Hyperparameters are learning rate η 0.001, weight decay 0.0, batch size 64, local epochs E 10, communication rounds 200, and event period t 1. The test results are presented in Tables 4 and 5. According to the results, in both datasets, SVHN (Table 4) and CIFAR-10 (Table 5), FedWAvg achieves higher accuracy than previous methods for both environment. Still, FedWAvg performance is sensitive to the $alpha$ hyperparameter.

Table 4. Test accuracy (%) for SVHN.

	2:7 Top	**2:7 Average**	**2:3:4 Top**	**2:3:4 Average**
FedAvg [1]	53.83	53.49	48.29	34.62
FedProx [11]	70.60	57.89	73.83	59.14
SCAFFOLD [12]	53.09	50.28	19.01	15.02
FedNova [13]	56.33	52.25	42.11	32.21
FedWAvg ($\alpha = 0.1$)	57.27	50.31	72.91	37.16
FedWAvg ($\alpha = 0.2$)	76.07	54.60	87.37	63.84
FedWAvg ($\alpha = 0.3$)	92.37	65.55	88.47	67.00

Table 5. Test accuracy (%) for CIFAR-10.

	2:7 Top	**2:7 Average**	**2:3:4 Top**	**2:3:4 Average**
FedAvg	61.28	45.94	45.20	36.85
FedProx	48.35	46.32	46.55	39.52
SCAFFOLD	44.35	42.13	31.03	27.18
FedNova	54.53	50.79	48.03	39.44
FedWAvg ($\alpha = 0.1$)	68.55	49.38	70.85	41.56
FedWAvg ($\alpha = 0.2$)	51.80	46.26	74.70	50.92
FedWAvg ($\alpha = 0.3$)	80.83	50.07	75.27	51.69

The second experiment compares the change of the number of forgettable examples over rounds. The experiment setup is nearly the same but with the following changes. There are only two clusters containing two clients and eight clients each. A cluster has two classes designated. The dataset used is CIFAR-10. Each client contains 1200 non-overlapping CIFAR-10 samples. Batch size is increased to 128 and communication round is 500. α is 0.3. Tables 6–8 present the number of forgettable examples by three federated learning methods. FedAvg (Table 6) and FedProx (Table 7) show nearly no improvement of reducing forgettable examples. On the other hand, FedWAvg (Table 8) have the number of forgetting examples in half by 200 rounds, although it seems to fluctuate after.

Table 6. FedAvg forgettable example counts in each client when non-IID (2-to-8).

Round	Minor1	Minor2	Major1	Major2	Major3	Major4	Major5	Major6	Major7	Major8
100	1197	1129	0	1	0	1	2	0	3	5
200	1169	1164	0	0	0	0	0	0	0	0
300	1008	1012	0	0	0	0	0	0	0	0
400	1030	1023	0	0	0	0	0	0	0	0
500	1065	1063	0	0	0	0	0	0	0	0

Table 7. FedProx forgettable example counts in each client when non-IID (2-to-8).

Round	Minor1	Minor2	Major1	Major2	Major3	Major4	Major5	Major6	Major7	Major8
100	1200	1193	3	6	6	7	13	13	8	11
200	1170	1172	1	0	1	0	2	0	0	0
300	1185	1184	0	0	0	0	0	1	0	0
400	1190	1177	0	0	0	0	0	2	0	0
500	1157	1147	0	0	0	0	0	0	0	0

Table 8. FedWAvg forgettable example counts in each client when non-IID (2-to-8).

Round	Minor1	Minor2	Major1	Major2	Major3	Major4	Major5	Major6	Major7	Major8
100	1193	1196	5	3	0	2	3	0	1	3
200	527	514	3	7	1	3	4	5	4	4
300	644	672	3	1	1	1	2	1	0	1
400	461	463	2	2	1	4	1	3	5	3
500	504	496	4	6	7	5	2	1	3	3

6. Discussion

The results of the first experiment show that our proposed method has stronger adaptation for extreme non-IID cases where a group of clients have less influence on the global model compared to other clients. FedWAvg expresses exceptional strength when groups are mildly differed in number but hierarchical, nonetheless. By the results of the second experiment, we can tell that our method is running as intended by its design. Combined, we suggest that having less forgetting events has contributed to enhancing the performance of the global model. A vital issue to note is that the performance is highly sensitive to the update ratio hyperparameter α. Currently, α is a hyperparameter, but could be developed to automatized. That is a topic of future research.

The parity between top performance and average performance is too large to ignore. Therefore, we executed a 1000 round training with the same setup as the second experiment. In the beginning, the global model fluctuates greatly every round. Over time, the amplitude decreases. Moreover, the number of forgettable example track records show that, beyond 400 rounds, the numbers stay within 450 to 500 range. It is converging to a certain degree. Presumably, this pattern occurs due to the fact that weights change every round since the event ratio t is 1. The drop of forgettable examples in minor clients after a successful weighted aggregation afflicts the averaging weights. The averaging weights then reduce the influence of weaker—minority cluster—clients causing the global model to drastically lose knowledge previously obtained from weaker clients. In the next round, this will be amended only to be broken again in the upcoming round. In other words, recalculating averaging weights every round creates disincentive to weaker clients. When the global model performs adequately, it causes the global model to lose their knowledge in the next round. Then, the loss of knowledge creates incentive to weaker clients again. Before verifying our presumption, we sought to inspect possible sources of the pattern. It could have been affected by the non-IID environment. The result is visualized in Figure 2.

Figure 2 shows each result of FedWAvg (orange), FedAvg (green), and FedProx (blue) in the same setup. Compared to FedWAvg, although the early trends are more stable, the mid and end performances oscillate just as FedWAvg. Even so, the general performance of FedWAvg is around 20% better than the other two methods. While FedAvg's performance swings around 50% and FedProx's performance is stuck between 50 and 60%, FedWAvg's performance is around 70% and higher. The common fluctuation in all three methods may be caused by the harsh label size imbalance.

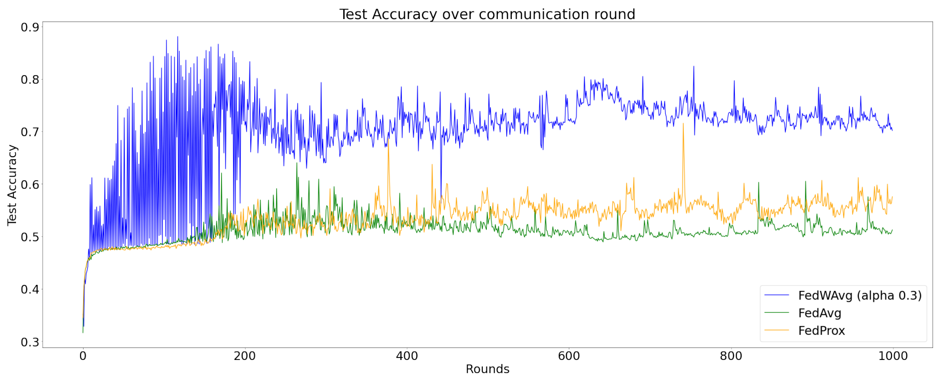

Figure 2. Test accuracy over communication rounds. The orange line is FedWAvg, green FedAvg, and blue FedProx.

The oscillation of FedWAvg results in a larger disparity between top accuracy and average accuracy than previous methods. Despite the comparatively higher general performance and the presumptive possibility of environmental cause, it would be desirable to reduce the amplitude, especially in the early stage of training. After monitoring 1000 rounds, we estimate that the presumption is invalid. Controlling event period t may have some alleviating effect but would not be an effective approach. Because averaging weights are defined by the immediate situation, holding seemingly good weights does not alleviate the issue. Moreover, the patterns in FedAvg and FedProx present that there will be fluctuations regardless. FedAvg, especially, provides insights since FedAvg is practically FedWAvg with fixed weight of ones. Technically, FedWAvg is FedAvg with variable averaging weights.

Another attempt was to try to freeze the classifier parameters. Since the classifier does not update, freezing its parameters would induce feature extractors to learn to fit the classifier. If the same classifier is shared for all clients, the classifier could function like an answer sheet [16]. Thus, we have frozen the classifier and trained FedWAvg in the same setup.

Figure 3 shows that there is an increasing trend with the classifier frozen (blue). However, the difference is less than 10% and also requires costly communications. It requires more than 700 rounds to see a distinguishable difference and 800 rounds to see a significant difference. Moreover, the fluctuation is still apparent. Although parameter freezing may increase the performance, it is blatantly inefficient.

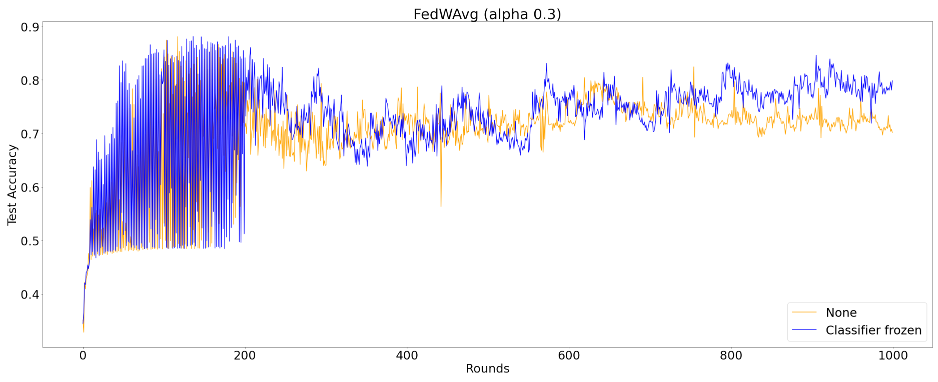

Figure 3. Test accuracy over communication rounds with parameter freezing. The blue line is FedWAvg with classifier frozen, and the orange line is FedWAvg without parameter freezing.

7. Conclusions

We suggested a new definition of the example forgetting problem in accordance with federated learning. We experimentally proved that the issue of example forgetting is worse in federated learning than in centralized learning. We proposed a federated weighted averaging algorithm (FedWAvg) that would weight local model parameters based on forgettable examples in each client. By monitoring the change of the number of forgettable examples, we proved that our method can lessen the information loss caused by model aggregation. Our experiments show that FedWAvg is better suited in extreme non-IID cases compared to previous methods. However, it has more to develop. The fluctuating performance caused by the non-IID environment is yet to be solved. Furthermore, the sensitivity to the update ratio hyperparameter should be copied for the method to be better stabilized.

Author Contributions: Conceptualization, M.H. and S.-K.K.; methodology, M.H. and J.-H.L.; software, M.H.; validation, M.H.; formal analysis, M.H.; investigation, M.H.; resources, M.H.; data curation, M.H.; writing—original draft preparation, M.H.; writing—review and editing, M.H.; visualization, M.H.; supervision, J.-H.L.; project administration, M.H.; funding acquisition, J.-H.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT). (No. 2019-0-00421, Artificial Intelligence Graduate School Program (Sungkyunkwan University)). This research was supported by Healthcare AI Convergence Research & Development Program through the National IT Industry Promotion Agency of Korea (NIPA) funded by the Ministry of Science and ICT. (No. S1601-20-1015).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: MNIST presented in this study are openly http://yann.lecun.com/exdb/mnist (accessed on 10 October 2021). Fashion-MNIST presented in this study are openly https://github.com/zalandoresearch/fashion-mnist.git (accessed on 10 October 2021). SVHN presented in this study are openly http://ufldl.stanford.edu/housenumbers (accessed on 10 October 2021). CIFAR-10 presented in this study are openly https://www.cs.toronto.edu/kriz/cifar.html (accessed on 10 October 2021).

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

References

1. McMahan, H.B.; Moore, E.; Ramage, D.; Hampson, S.; Arcas B.A. Communication-Efficient Learning of Deep Networks from Decentralized Data. In Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, Fort Lauderdale, FL, USA, 20–22 April 2017; pp. 1273–1282.
2. Kairouz, P.; McMahan, H.B.; Avent, B.; Bellet, A.; Bennis, M.; Bhagoji, A.N.; Bonawitz, K.; Charles, Z.; Cormode, G.; Cummings, R.; et al. Advances and Open Problems in Federated Learning. *Found. Trends Mach. Learn.* **2021**, *14*, 1–210. [CrossRef]
3. Konečný, J.; McMahan, H.B.; Yu, F.; Richtárik, P.; Suresh, A.T.; Bacon, D. Federated Learning: Strategies for Improving Communication Efficiency. In Proceedings of the NIPS Workshop on Private Multi-Party Machine Learning, Barcelona, Spain, 9 December 2016.
4. Konečný, J.; McMahan, H.B.; Ramage, D. Federated Optimization: Distributed Optimization Beyond the Datacenter. In Proceedings of the 8th NIPS Workshop on Optimization for Machine Learning, Montreal, QC, Canada, 11 December 2015.
5. Lim, W.Y.B.; Luong, N.C.; Hoang, D.T.; Jiao, Y.; Liang, Y.C.; Yang, Q.; Niyato, D.; Miao, C. Federated Learning in Mobile Edge Networks: A Comprehensive Survey. *IEEE Commun. Surv. Tutor.* **2020**, *22*, 2031–2063. [CrossRef]
6. McCloskey, M.; Cohen, N.J. Catastrophic Interference in Connectionist Networks: The Sequential Learning Problem. In *Psychology of Learning and Motivation*; Academic Press: Cambridge, MA, USA, 1989; Volume 24, pp. 109–165.
7. Ratcliff, R. Connectionist Models of Recognition Memory: Constraints Imposed by Learning and Forgetting Functions. *Psychol. Rev.* **1990**, *97*, 285–308. [CrossRef] [PubMed]
8. Toneva, M.; Sordoni, A.; Combes, R.T.; Trischler, A.; Bengio, Y.; Gordon, G.J. An Empirical Study of Example Forgetting during Deep Neural Network Learning. In Proceedings of the 7th International Conference on Learning Representations, New Orleans, LA, USA, 6–9 May 2019.

9. Yurochkin, M.; Agarwal, M.; Ghosh, S.; Greenewald, K.; Hoang, T.N.; Khazaeni, Y. Bayesian Nonparametric Federated Learning of Neural Networks. In Proceedings of the 36th International Conference on Machine Learning, Long Beach, CA, USA, 9–15 June 2019; pp. 7252–7261.
10. Yang, Q.; Liu, Y.; Chen, T.; Tong, Y. Federated Machine Learning: Concept and Applications. *ACM Trans. Intell. Syst. Technol.* **2019**, *10*, 1–19. [CrossRef]
11. Li, T.; Sahu, A.K.; Zaheer, M.; Sanjabi, M.; Talwalkar, A.; Smith, V. Federated Optimization in Heterogeneous Networks. In Proceedings of the 3rd Conference on Machine Learning and Systems, Austin, TX, USA, 2–4 March 2020.
12. Karimireddy, S.P.; Kale, S.; Mohri, M.; Reddi, S.J.; Stich, S.U.; Suresh, A.T. SCAFFOLD: Stochastic Controlled Averaging for Federated Learning. In Proceedings of the 37th International Conference on Machine Learning, Virtual, 12–18 July 2020; pp. 5132–5143.
13. Wang, J.; Liu, Q.; Liang, H.; Joshi, G.; Poor, H.V. Tackling the Objective Inconsistency Problem in Heterogeneous Federated Optimization. In Proceedings of the 34th Conference on Neural Information Processing Systems, Virtual, 6–12 December 2020.
14. Wang, H.; Yurochkin, M.; Sun, Y.; Papailiopoulos, D.; Khazaeni, Y. Federated Learning with Matched Averaging. In Proceedings of the International Conference on Learning Representations, Virtual, 26 April–1 May 2020.
15. Li, X.; Jian, M.; Zhang, X.; Kamp, M.; Dou, Q. FedBN: Federated Learning on Non-IID Features via Local Batch Normalization. In Proceedings of the 9th International Conference on Learning Representations, Virtual, 3–7 May 2021.
16. Oh, J.; Kim, S.; Yun, S. FedBABU: Toward Enhanced Representation for Federated Image Classification. In Proceedings of the 10th International Conference on Learning Representations, Virtual, 25–29 April 2022.
17. Hinton, G.; Yinyals, O.; Dean, J. Distilling the Knowledge in a Neural Network. In Proceedings of the NIPS Workshop on Deep Learning and Representation Learning, Montreal, QC, Canada, 12 December 2014.
18. Lin, T.; Kong, L.; Stich, S.U.; Jaggi, M. Ensemble Distillation for Robust Model Fusion in Federated Learning. In Proceedings of the 34th Conference on Neural Information Processing Systems, Virtual, 6–12 December 2020.
19. Zhu, Z.; Hong, J.; Zhou, J. Data-Free Knowledge Distillation for Heterogeneous Federated Learning. In Proceedings of the 38th International Conference on Machine Learning, Virtual, 18–24 July 2021.
20. Yann, L.; Corest, C.; Burges, C.J.C. The MNIST Database of Handwritten Digits. Available online: http://yann.lecun.com/exdb/mnist (accessed on 10 October 2021).
21. Xiao, H.; Rasul, K.; Vollgraf, R. Fashion-MNIST. Available online: https://github.com/zalandoresearch/fashion-mnist.git (accessed on 10 October 2021).
22. Netzer, Y.; Wang, T.; Coates, A.; Bissacco, A.; Wu, B.; Ng, A.Y. The Street View House Numbers (SVHN) Dataset. Available online: http://ufldl.stanford.edu/housenumbers (accessed on 10 October 2021).
23. Krizhevsky, A.; Nair, V.; Hinton, M.G. CIFAR. Available online: https://www.cs.toronto.edu/~kriz/cifar.html (accessed on 10 October 2021).
24. Huang, G.; Liu, Z.; Maaten, L. Densely Connected Convolutional Networks. In Proceedings of the 30th IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017.
25. Zhu, H.; Xu, J.; Liu, S.; Jin. Y. Federated Learning on Non-IID Data: A Survey. *Neurocomputing* **2021**, *465*, 371–390. [CrossRef]
26. He, K.; Zhang, X.; Ren, S.; Jian, S. Deep Residual Learning for Image Recognition. In Proceedings of the 29th IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 26 June–1 July 2016.