On improving accuracy in Federated Learning using GANs-based pre-training and Ensemble Learning

Thomas Tsouparopoulos and Iordanis Koutsopoulos
Department of Informatics,
Athens University of Economics and Business, Greece
{tsouparop20,jordan}@aueb.gr

Abstract. This paper presents a novel training pipeline for Federated Learning (FL), enriched in two aspects, with the goal of improving accuracy. First, we exploit the generative ability of Generative Adversarial Networks (GANs) to augment the clients' local datasets with synthetic data and second, we incorporate them into the FL training procedure with the help of Ensemble Learning. Drawing inspiration from their demonstrated potential in Deep Learning (DL), we adeptly modify these techniques to address the privacy concerns and distributed nature inherent in FL. Our proposed FL pipeline lead to a 3% and 2.5% improvement in the accuracy of the *global model* on the MNIST and CIFAR-10 test sets, respectively, compared to the baseline and modified versions of *FedAvg*. This paves the way for exploring the potential of our method in achieving similar or larger improvement in other FL algorithms.

Keywords: Federated Learning · Generative Adversarial Networks · Ensemble Learning · Pre-training

1 Introduction

The proliferation of private and decentralized data has necessitated the adoption of distributed Machine Learning (ML) paradigms, such as Federated Learning (FL). FL enables clients to locally train models on their respective data and then these local models are aggregated at a central server to create a *global model*, synthesizing knowledge from all participating clients [1]. In real-world scenarios, data owners often encounter data imbalance across their datasets, which poses a challenge to the performance of the *global model*. Consequently, clients with smaller datasets may have minimal contributions to the *global model*, potentially resulting in inadequate performance for classes with limited data points.

A natural approach to address data imbalance between clients' datasets or in general data scarcity, is to exploit the recent advancements in GANs and synthesize new data points for those clients with few examples. Synthetic data through generative models can help balance the differences in the sizes of the clients' datasets, prevent overfitting on the majority classes and alleviate, to some extent, the problem of data scarcity. However, while this approach can improve the accuracy of a model, the inherent dissimilarity between synthetic and real data raises concerns about the efficacy of naively incorporating synthetic samples into the training procedure.

An additional drawback of current FL training frameworks is that they do not effectively capitalize on the *idle time* of clients prior to commencing the training

process. While pre-training the FL model on a large public dataset improves in some cases the accuracy of the model, it incurs additional storage requirements for clients, to host the dataset, and requires a labeled dataset, which may not be readily available due to the different distribution of data in the source task. This motivates further research in FL training frameworks that leverage pre-training and data synthesis to harness clients' *idle time* and computational capabilities, ultimately enhancing the performance of the *global model*. This paper explores such approaches, aiming to create favorable conditions for the initialization of the FL training process and thereby enriching the overall FL framework.

Amidst growing interest in addressing the aforementioned challenges in FL, numerous works have proposed novel algorithms and aggregation techniques. However, the vast majority of these works remain reliant on the conventional FL training pipeline, consisting of (*i*) client-level model training, (*ii*) models' parameters transmission to the server, (*iii*) server-side parameters aggregation to construct a new *global model*, and (*iv*) dissemination of the *global model* back to clients for subsequent training rounds. In this paper, we introduce a new, enriched framework for the FL training process, specifically designed to achieve higher accuracy. Our approach diverges from conventional methods and introduces novel mechanisms to optimize the FL pipeline at various stages. The main contributions of this work can be summarized as follows:

- We exploit the *idle time* of clients to pre-train ML models from distributed datasets in FL, through GANs. We then use the Generator to generate synthetic data and the weights of the Discriminator as weight initialization for the network used in the main FL procedure.
- We use Ensemble Learning to incorporate the synthetic data into the FL training procedure. By initiating two distinct learning procedures optimized for real and synthetic data, and by merging their acquired knowledge, we enhance the accuracy performance of the *global model*.
- Our data experiments show an increase in test accuracy of up to 3% for the MNIST and 2.5% for the CIFAR-10 dataset when compared to baseline or modified variations of *FedAvg*.

2 Related Work

FL algorithms: *FedAvg* [1] was chronologically the first algorithm to train a *global model* in the FL setting, by averaging the weights of the clients' models at the server. Its simplicity and effectiveness has made it the standard algorithm of choice, to both implement in applications and compare against in benchmarks. In [2], the authors prove that the learning performance of *FedAvg* suffers when the data are heterogeneous or non independently and identically distributed (non-i.i.d). To alleviate this issue, the authors in [3] add a proximal term to each local objective, to prevent the local parameters from diverging from the *global model* while in [4], the authors use the difference between the update direction of the server and the clients' models to correct the local updates of clients. This class of methods improved the performance over *FedAvg*, however they adhere to a given FL pipeline, while our approach presents a new one.

Pre-training in FL: In the literature of FL, neural networks are mostly initialized with random weights or pre-trained on public datasets [5]. In [6], the authors pre-train a *global model* on massive public image datasets at the server side and then transfer the knowledge to each client where it is combined with the local model. To the best of our knowledge, the only work that directly addresses the issue of pre-training in FL is the one in [7]. The authors use models pre-trained on public datasets to explore their contribution in classification performance and suggest the usage of fractals as synthetic data for pre-training when no pre-trained models are available. We avoid pre-training on public datasets, as they usually drastically differ in terms of statistical heterogeneity and thus, provide marginal improvements in accuracy performance. Instead, we pre-train the clients' models on their training dataset through a GAN, to exploit the benefits of pre-training.

GANs and data synthesis in FL: In [8], the authors provide the first solution to training GANs in the FL setting, by equipping both the server and the clients with a GAN of the same architecture and averaging the parameters of both the Generator and the Discriminator at the server. While aggregating the Generators provides better results, it also raises privacy concerns, as the Generators are able to produce data that resemble the original ones of the clients. In another line of works, they exchange (synthetic) data produced by clients [10], Generators [11] or even both [9] in order to improve the learning performance of FL. These works violate privacy restrictions imposed by real-world scenarios. In contrast, in our framework, we train the GANs on local data, and we do not exchange the Generator's model parameters or synthetic data with the server. As a result, no information about the datasets is leaked to the server. Finally, we use generated data that share statistics with the local datasets to augment the clients' datasets, as opposed to sharing data across clients.

Ensemble Learning in FL: Ensemble Learning methods combine several base models to create a single more accurate model [15]. In the context of FL there are several cases where Ensemble Learning has been used but none of them addresses data imbalances between clients or incorporates synthetic data into the FL training procedure. In [12], the authors construct and train decision trees on samples of training data and then create a global random forest for inference. In [13], the authors use Ensemble Learning for a set of models pre-trained on public data and their goal is to find a set of weights to appropriately combine them for both standard and agnostic empirical risk minimization. Finally, the authors in [14] show that using Ensemble Learning to integrate multiple models trained on multiple datasets in the FL procedure achieves better generalization performance compared to single model approaches. To the best of our knowledge, our work is the first that uses Ensemble Learning in order to incorporate the synthetic data in the FL training procedure.

3 Pre-training with GANs in FL

We propose a new framework for FL training, that consists of three stages. The architecture, comprising a server functioning as a model aggregator and two clients is illustrated in Fig. 1. A GAN with identical architecture is deployed and

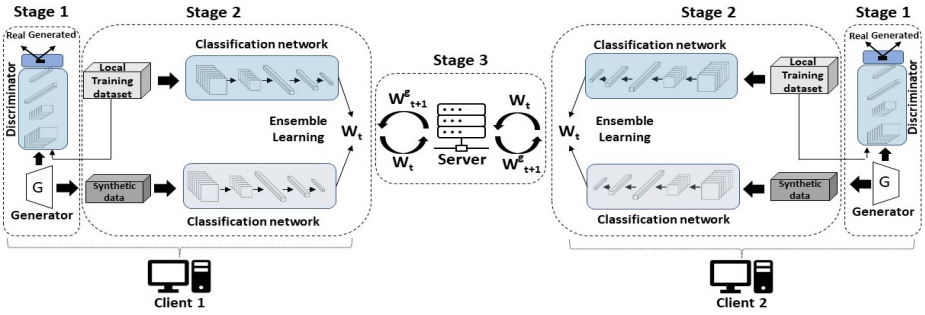

Fig. 1: The architecture of our framework for two clients. At each round t, Ensemble Learning is employed by clients to merge the obtained weights from two identical networks trained on synthetic and real data respectively, into W_t. Subsequently, the server aggregates these weights and returns the resulting *global model* W^g_{t+1} back to clients.

trained on the local datasets. Subsequently, the trained Generator is employed by clients to generate a new training dataset consisting solely of synthetic data. Additionally, two identical classification networks (e.g., CNNs) are trained on the real and synthetic datasets, respectively. The traditional FL pipeline is then utilized for the exchange of model parameters between the clients and the server.

3.1 Our proposed approach

Stage 1 - Local GAN training at each client: The incorporation of GANs in our proposed framework is supported by two fundamental assumptions. Firstly, if the lower layers of the Discriminator are considered as feature extractors, then their pre-training as part of a GAN facilitates the learning of general and transferable information about the dataset. Secondly, because GAN training is unsupervised, it grants us a trained Generator, which we utilize to augment the local clients' dataset with synthetic data. In our proposed framework, each client hosts locally a GAN with the same architecture. The initial training phase occurs offline i.e., before the clients connect to the server. The clients train locally their GAN on their local training dataset for a predetermined number of epochs until the Generator yields satisfactory results. The quality of these results depends on factors such as the client computational capabilities, the difficulty of the dataset and available offline time. Due to these considerations, the naive integration of any amount or quality of synthetic data into the training datasets will not consistently ensure improved results.

After the completion of GAN training, we obtain two outputs: a trained Generator and a trained Discriminator, as depicted in Fig. 2. The Discriminator's objective is to learn common and low-level features present in the local dataset. Thus, by removing the output layer of the trained Discriminator, we transform the remaining network into a feature extractor. We then incorporate a *dense*

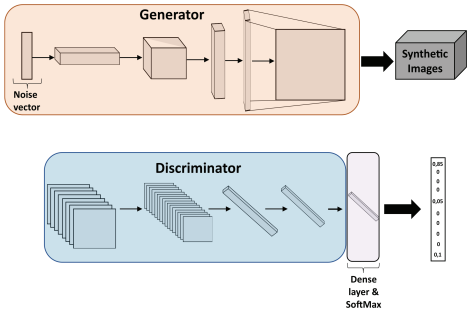

Fig. 2: The trained Generator is used for generating synthetic data and in the trained Discriminator a *dense layer + SoftMax* is added to create a multi-class classification network.

layer + SoftMax to devise a classification network for the third stage of our approach. Finally, we exploit the trained Generator to generate synthetic data and enhance the training datasets of clients, thereby mitigating any data imbalances or scarcities among clients.

Stage 2 - Ensemble Learning: Additionally, we exploit Ensemble Learning to integrate synthetic data into the training procedure. Each client trains two distinct models: one on real data and another on synthetic data. A convex combination of these models results in a final model that effectively captures features from both real and synthetic data. This strategy circumvents sole reliance on potentially scarce or biased real data while also mitigating the risks associated with synthetic data, which may not effectively capture the complexities of real datasets.

Concretely, let us consider a client i with two types of training data: a real dataset, denoted as D_i^{Real}, and a synthetic dataset, denoted as D_i^{Syn} with sizes $|D_i^{Real}|$ and $|D_i^{Syn}|$ respectively. Two models, denoted as M_1 and M_2 are trained on the real and synthetic data, respectively. The local ensemble prediction for a new input x is given by the weighted sum of the two models:

$$Ensemble(x) = w^{Real} * M_1(x) + w^{Syn} * M_2(x)$$

where

$$w^{Real} = \frac{|D_i^{Real}|}{|D_i^{Real}| + |D_i^{Syn}|} \; and \; w^{Syn} = \frac{|D_i^{Syn}|}{|D_i^{Real}| + |D_i^{Syn}|}$$

Stage 3 - Main FL process: In the final stage of the framework, after clients connect to the server, they receive the initial model weights comprising a *dense layer + SoftMax*. Subsequently, each client creates two identical models, wherein the lower layers are initialized using the weights from the pre-trained Discriminator's lower layers, and the last layer is initialized with the weights received from the server. In each iteration, the clients update their two models using *stochastic gradient descent* (SGD). Next, they combine the weights using a convex combination and transmit the resulting model to the server. The server, in

turn, aggregates the weights received from the clients to form a *global model* and returns it to the clients. Finally, the clients utilize it in the next iteration for both their models. This iterative process continues until convergence is attained.

3.2 Discussion

Our proposed pipeline incurs no additional communication cost compared to standard FL pipelines since users exchange only one model per round. While it is computationally intensive, it delivers improved accuracy for the *global model*. It does require larger memory, considering the GAN and two parallel models' loading requirements. Additionally, the framework has higher storage space demands compared to plain FL, as each client must store synthetic data (12.298KB), the Generator model (4.083KB), and the second classification network (2.536KB). Therefore, our approach is best suited for applications where clients' computational and memory limitations are not a concern, and accuracy enhancement is paramount. A typical instance of such a scenario is one where each client corresponds to a different hospital with its own patient records.

4 Experiments and performance evaluation

4.1 Experimental setup

Dataset: We evaluate our framework on the default test partition of MNIST and CIFAR-10 datasets. For pre-processing, images were normalized and then organized into batches containing 32 images. During data allocation, the data is sorted by class (labels), and each client is assigned only one class of data points, ensuring no overlaps (i.e., non-i.i.d distribution). Additionally, clients possess a validation set comprising data points separated from their training dataset.

Data Imbalance and training hyperparameters: To create a scenario with data imbalance, two clients possess only 10% of the data available to the other eight clients locally. In our experimental setup, we train the GAN at the clients with the smaller datasets for 100 epochs, for both the MNIST and CIFAR-10 datasets. Upon completion of training, we use the Generator to produce and store synthetic images that correspond to 10% of the whole training dataset. As a local optimizer for the classification task, we utilize the SGD optimizer with a learning rate of 0.001 and momentum of 0.9.

Baselines: As a first baseline to compare against, we use the conventional FL framework that is based on the *FedAvg* pipeline (*Vanilla FedAvg*). Specifically, the clients train their local models on one dataset consisting of both the synthetic and the real data, and we evaluate the *global model* that results from the aggregation at the server on the test set. We further consider a modified version of the plain FL training procedure to compare against, referred to as *FedAvg with stratified sampling*. In this approach, each client, except for two clients with limited examples, randomly selects an equal-size subset of data points from their dataset at each round for training [15]. As a result, all models' parameters are equally weighted at the server, ensuring equal contributions from all clients. This baseline is introduced to validate that even if the *global model* results from models trained on an equal amount of data (thus equally weighting each model), it

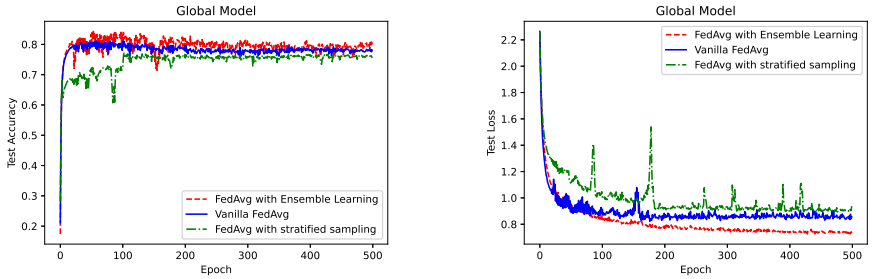

Fig. 3: Test accuracy (left) and loss (right) of the *global model* on the MNIST dataset, when two clients have imbalanced datasets.

would still require more rounds to achieve a certain accuracy compared to our proposed framework that leverages synthetic data through Ensemble Learning.

4.2 Experiment results

We conducted our experiments for 500 communication rounds. As depicted in Fig. 3, our proposed framework (*FedAvg with Ensemble Learning*) demonstrates a notable improvement of 3% in test accuracy for MNIST dataset. Specifically, our framework achieves a test accuracy of 84.28%, surpassing the second-best approach (*Vanilla FedAvg*) with an accuracy of 81.06% and *FedAvg with stratified sampling* with an accuracy of 77.11%. Additionally, our framework exhibits a 11% reduction in test loss (a value of 0.72) when compared to the plain *FedAvg* approach (a value of 0.82). Stratified sampling struggled to attain favorable accuracy and loss due to having less data available for training. Finally, there is also an improvement of 2.5% for the CIFAR dataset in test accuracy when averaged across the communication rounds.

5 Conclusions and Future Work

We introduced a novel training pipeline for FL, which utilized GANs for pre-training in FL to address data imbalance and Ensemble Learning to incorporate the synthetic data into the FL training procedure more effectively. Experimental results demonstrated an improvement of 3% in test accuracy for MNIST and 2.5% for CIFAR-10 datasets. As a future work, there are hyperparameters that require tuning, such as the number of local epochs a GAN should be trained for. In addition, since our approach is computationally intensive, further optimization for a more efficient training procedure is required. More possibilities for combining the information of the two datasets i.e., real and synthetic, could also be explored. Finally, another direction would also be to extend our framework towards personalization so that each client could dynamically adjust at each round the weights of the two models to cater for its local objective.

6 Acknowledgment

This work was supported by the CHIST-ERA grant CHIST-ERA-18-SDCDN-004 (project LeadingEdge, grant number T11EPA4- 00056) through the General Secretariat for Research and Innovation (GSRI).

This work was also conducted in the context of the Horizon Europe project PRE-ACT (Prediction of Radiotherapy side effects using explainable AI for patient communication and treatment modification). It was supported by the European Commission through the Horizon Europe Program (Grant Agreement number 101057746), by the Swiss State Secretariat for Education, Research and Innovation (SERI) under contract number 22 00058, and by the UK government (Innovate UK application number 10061955).

References

1. B. McMahan, E. Moore, D. Ramage, S. Hampson and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," *Artificial intelligence and statistics*, PMLR, 2017.
2. X. Li , K. Huang, W. Yang, S. Wang and Z. Zhang, "On the convergence of *FedAvg* on non-iid data." 2019. [Online]. Available: arXiv:1907.02189.
3. T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar and V. Smith"Federated Optimization in Heterogenous Networks," 2018. [Online]. Available: arXiv:1812.06127.
4. S. P. Karimireddy, S. Kale, M. Mohri, S. J. Reddi, S. U. Stich and A. T. Suresh "Scaffold: Stochastic controlled averaging for federated learning," *International Conference on Machine Learning*, PMLR, 2020.
5. Nguyen J., Malik K., Sanjabi M., Rabbat M., "Where to begin? exploring the impact of pre-training and initialization in federated learning," 2022, [ONLINE]. Available: arXiv:2206.15387.
6. Y. Chen, X. Qin, J. Wang, C. Yu and W Gao, "Fedhealth: A federated transfer learning framework for wearable healthcare," *IEEE Intelligent Systems*, 2020.
7. Chen H. Y., Tu C. H., Li Z., Shen H. W., Chao W. L., "On the Importance and Applicability of Pre-Training for Federated Learning," 2022, [ONLINE]. Available: arXiv:2206.11488.
8. Hardy C., Merrer E. L. and Sericola B., "MD-GAN: Multi-Discriminator Generative Adversarial Networks for Distributed Datasets," *IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, pp. 866-877, 2019.
9. Jeong E., Oh S., Kim H., Park J., Bennis M. and Kim S. L. "Communication-efficient on-device machine learning: Federated distillation and augmentation under non-iid private data," 2018, [ONLINE]. Available: arXiv:1811.11479.
10. Lewy D., Mańdziuk J., Ganzha M. and Paprzycki M., "StatMix: Data augmentation method that relies on image statistics in federated learning," 2022, [ONLINE]. Available: arXiv:2207.04103.
11. Behera M. R., Upadhyay S., Shetty S., Priyadarshini S., Patel P. and Lee K. F., "FedSyn: Synthetic Data Generation using Federated Learning," 2022, [ONLINE]. Available: arXiv:2203.05931.
12. Yujin H., Du P. and Yang K., "Fedgbf: An efficient vertical federated learning framework via gradient boosting and bagging," 2022. [ONLINE]. Available: arXiv:2204.00976.
13. Jenny H., Mohri M. and Suresh A. T. , "Fedboost: A communication-efficient algorithm for federated learning," *International Conference on Machine Learning*, PMLR, 2020.
14. Shi N., Lai F., Kontar R. A. and Chowdhury M., "Fed-ensemble: Improving generalization through model ensembling in federated learning," 2021, [ONLINE]. Available: arXiv:2107.10663.
15. Zhou Z. H., "Machine learning," *Springer Nature*, 2021.