Addressing Class Imbalance in Federated Learning via Collaborative GAN-Based Up-Sampling

Can Zhang[1], Xuefeng Liu[1(✉)], Shaojie Tang[2], Jianwei Niu[1], Tao Ren[1], and Quanquan Hu[1]

[1] Beihang University, Beijing, China
{can_zhang,liu_xuefeng,niujianwei,nebulau}@buaa.edu.cn
[2] The University of Texas at Dallas, Richardson, USA
shaojie.tang@utdallas.edu

Abstract. Federated learning (FL) is an emerging learning framework that enables decentralized devices to collaboratively train a model without leaking their data to each other. One common problem in FL is class imbalance, in which either the distribution or quantity of the training data varies in different devices. In the presence of class imbalance, the performance of the final model can be negatively affected. A straightforward approach to address class imbalance is up-sampling, by which data of minority classes in each device are augmented independently. However, this up-sampling approach does not allow devices to help each other and therefore its effectiveness can be greatly compromised. In this paper, we propose FED-CGU, a collaborative GAN-based up-sampling strategy in FL. In FED-CGU, devices can help each other during up-sampling via collaboratively training a GAN model which augments data for each device. In addition, some advanced designs of FED-CGU are proposed, including dynamically determining the number of augmented data in each device and selecting complementary devices that can better help each other. We test FED-CGU with benchmark datasets including Fashion-MNIST and CIFAR-10. Experimental results demonstrate that FED-CGU outperforms the state-of-the-art algorithms.

Keywords: Federated learning · Class imbalance · Collaborative up-sampling · Generative adversarial networks

1 Introduction

Federated learning (FL) is a popular distributed learning framework that allows a large number of mobile devices to collaboratively train a global model without sharing their local data [16]. However, one phenomenon that can degrade the performance of FL is class imbalance. Class imbalance, defined as the disproportionate ratio of observations in each class, is a concept originally defined for traditional centralized machine learning. In FL, class imbalance can happen

Z. J. Haas et al. (Eds.): WiCON 2022, LNICST 464, pp. 213–227, 2023.
https://doi.org/10.1007/978-3-031-27041-3_15

either within a device (an individual device is class-imbalanced) or among multiple devices (the overall class distribution across devices is imbalanced). The phenomenon of class imbalance happens frequently in practical scenarios of FL. For example, when multiple mobile devices collaboratively train a image classification model using their own photos, the number of photos with different themes can vary greatly both within each device and across all devices according to users' interests. Class imbalance can slow down the convergence and even degrade the performance of the obtained global model [12,25].

How to address the class imbalance has been studied extensively in centralized machine learning. Methods for handling class imbalance can be largely grouped into algorithm-level techniques and data-level methods [6]. A widely used data-level method is up-sampling, which augments samples in minority classes in order to decrease the level of imbalance. Experimental results show that up-sampling generally displays a good performance on various datasets [15]. To implement the up-sampling technique in FL, a straightforward approach is to let each device augment enough number of samples in minority classes until class balance is achieved. However, as up-sampling is implemented independently in each device, samples augmented in one device do not exploit information from others.

Therefore, a collaborative up-sampling strategy in FL that allows devices to help each other without leaking their private data is greatly desired. To address this issue, one strategy is to utilize a Generative Adversarial Network (GAN). In particular, devices first collaboratively train a GAN in the framework of FL. Then the GAN is utilized to augment minority classes for each device. As the GAN is trained collaboratively by all devices, up-sampling for one device leverages information from others.

Although the above idea of using a GAN for collaborative up-sampling is simple, implementing such an idea, however, entails substantial challenges.

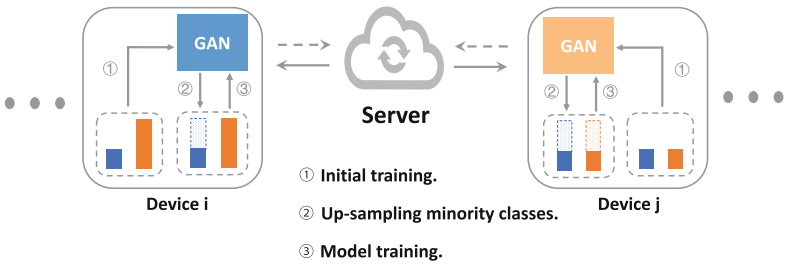


Fig. 1. Collaborative GAN-based up-sampling to address class imbalance in FL.

First, we find that the GAN model generally cannot be well trained in the presence of class imbalance. Therefore, a better approach is to let the two tasks (i.e. training the GAN and up-sampling minority classes) be implemented alternately in an iterative manner. As shown in Fig. 1, a GAN is firstly trained using the original data of all devices. Then the GAN is utilized for augmenting data in

all devices. Afterwards, the GAN is updated again based on the updated data. This process alternates in an iterative manner until the model converges.

Another challenge is for each round of the above process, how to determine the number of augmented data in each device. We find that GAN is generally not well trained initially, and generating a large number of low-quality samples can have a negative effect on the convergence, which may further lead to lower quality augmented samples. The intuition is that the number of augmented data by GAN should be dynamically determined by their quality. To implement this idea, we utilize a genetic algorithm, which utilizes the performance of the target training task as a guidance to dynamically determine the number of augmented data for each device. In addition, we incorporate prior knowledge, which states that the number of augmented data is increased, into the genetic algorithm to reduce the searching rounds of the genetic algorithm.

Another issue is how to select devices in each round of the above process. In the traditional FL framework, devices are generally selected uniformly at random. However, we find that in the presence of class imbalance, choosing devices with complementary classes can often accelerate the convergence of the GAN model. We utilize the Kullback-Leibler Divergence (KLD) distance as a measure to select complementary devices in each communication round of FL. Implementing this strategy, we can reduce the computation and communication cost for the devices.

In view of all the above issues, we propose FED-CGU, the first collaborative GAN-based up-sampling strategy to address class imbalance in FL. In FED-CGU, devices can help each other during up-sampling via collaboratively training a GAN model. In addition, FED-CGU has some advanced designs, including alternately implementing GAN training and up-sampling in an iterative manner, dynamically determining the number of augmented data in each device, and selecting the complementary devices in each round of FL. We test FED-CGU with benchmark datasets including Fashion-MNIST and CIFAR-10, and the experimental results demonstrate that FED-CGU outperforms the state-of-the-art algorithms.

Our contributions are listed as follows:

- We propose the first collaborative up-sampling strategy in FL that allows devices to help each other without leaking their private data.
- We design a strategy that can dynamically determine the number of augmented data for each device.
- We design a device sampling strategy which selects complementary devices to accelerate the convergence of the GAN model.
- We test FED-CGU with benchmark datasets including Fashion-MNIST and CIFAR-10. Experimental results indicate that FED-CGU outperforms the state-of-the-art algorithms.

2 Related Work

Class imbalance can make the overall accuracy of machine learning biased to the majority classes which leads to misclassifying the minority class samples or furthermore treats them as noise [6]. Prior approaches to address this problem in centralized learning can be grouped into algorithm-level methods and data-level methods [6]. Algorithm-level methods punish classification mistakes on the minority classes [21,26]. Such methods make the model focus more on minority classes. On the other hand, data-level methods modify the class distributions to decrease the level of class imbalance. Data-level methods include down-sampling and up-sampling. Down-sampling methods only selects partial data in majority classes [19], and up-sampling methods augment data in minority classes (e.g. rotation, scaling or translation) [1,13]. Experimental results on some large datasets show that up-sampling generally displays better performance than down-sampling [15]. However, all the above methods implement common up-sampling (e.g. flipping, rotation), which are designed for a single device. When these methods are implemented in FL, they can not exploit information from other devices. Although using Generative Adversarial Networks (GAN) [5] and its improved versions [17,18] for up-sampling can leverage other devices' information, training a GAN model in is a challenging task with the presence of class imbalance.

In addition to centralized methods, approaches to address class imbalance in FL are also widely studied. We classify these approaches into several categories. The first category is to optimize the loss function of FL. L. Wang et al. [23] leverages the global imbalance and Q. Li et al. [11] aligns representation learned by the local models to address class imbalance. The second category is personalized federated learning [4,20]. Personalized strategies train a personalized model for each device with the help from other devices. Another category is the aggregation scheme. Aggregation schemes modify the aggregation function of FL to faster the convergence in non-IID data [2,9,14]. Recently, Astraea [3] and FAVOR [22] apply device sampling to deal with class imbalance in FL. Astraea and FAVOR choose the devices to participate in FL to counterbalance the bias introduced by class imbalance. Our proposed FED-CGU is perpendicular to the above approaches and can improve their performance.

3 Algorithm Design

3.1 FED-CGU Overview

The overview of FED-CGU framework is shown in Fig. 2. We choose training a classifier as the target task. In FED-CGU, we also train an extra GAN model and up-sample minority classes alternately in an iterative manner. We propose a complementary device sampling strategy to accelerate the convergence of GAN, and design a genetic algorithm to determine the number of up-sampled data. The workflow of FED-CGU is as follows:

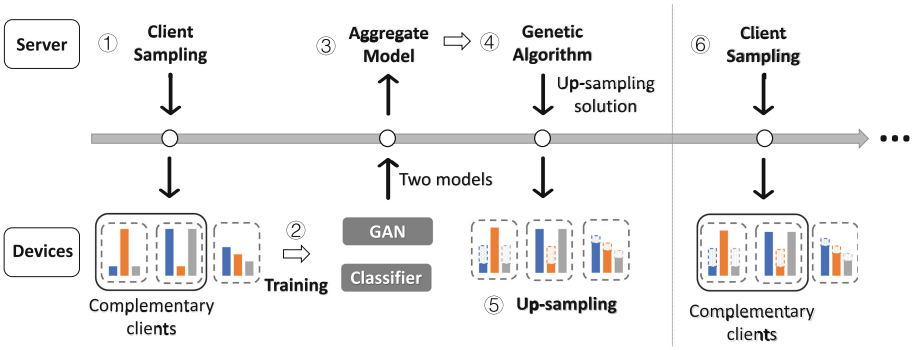


Fig. 2. Overview of FED-CGU framework.

- *Step* 1 : The server selects a few devices whose data distributions are complementary.
- *Step* 2 : The selected devices update the classifier and GAN on their local data and augmented data, and then upload the models.
- *Step* 3 : The server aggregates the uploaded models.
- *Step* 4 : After multiple rounds' training and aggregation, the server utilizes a genetic algorithm to generate up-sampling solutions based on the evaluation of the classifier.
- *Step* 5 : Devices download the solutions and then implement GAN for up-sampling according to the solutions.
- *Step* 6 : Go to *step* 1 and repeat the whole procedure.

3.2 Detailed Design of FED-CGU

We introduce the detailed designs of FED-CGU in this subsection. The content is organized as follows. First, we discuss the complementary device sampling strategy. Then we introduce the design of our up-sampling method. Finally, we introduce how we utilize the genetic algorithm to determine the number of augmented data.

Design of Device Sampling. We observe that in the presence of class imbalance, complementary devices can accelerate each other's training. The key point to leverage this observation is to explore complementary devices. If the overall class distribution of a group of devices is similar to a balanced distribution, the devices in this group are complementary. In this paper, we utilize the Kullback-Leibler Divergence of devices' class distribution to select complementary devices and the optimization function is as follows:

$$\min \sum_{c_i \subset C} D_{KL}(P_{c_i} \| P_{balance}) \tag{1}$$

where C denotes the set of all involved devices, $\{c_1, c_2, \ldots | c_1 \cup c_2 \cup \ldots = C\}$ are independent subsets of C. P_{c_i} denotes the class distribution of clients in c_i and $P_{balance}$ denotes the balanced class distribution. Based on the above function, we can divide devices into complementary subsets and devices in each subset are complementary to each other. In the process of each round's training, we randomly select a subset of devices to participate.

Design of Up-Sampling. Common up-sampling methods (flipping, rotation, etc.) are widely utilized in centralized training. However, since common up-sampling methods are based on devices' local data, they cannot exploit the information from other devices to improve the effects of up-sampling. In addition, simply using local data can also make devices easy to cause over-fitting. If a device does not have data in specific classes, we are even unable to supplement such classes.

To solve the above issues, we utilize a GAN model to perform up-sampling. Since the GAN model is collaboratively trained by all devices, it can leverage the information of all devices' data and prevent over-fitting. A straightforward strategy to implement up-sampling with GAN is to first train a GAN model and then utilize the GAN to augment data. However, since class imbalance can make the model treat the minority classes as noise, directly training a GAN model is a challenging task.

To address this challenge, we propose to train the GAN model and up-sample minority classes alternately in an iterative manner. We initially leverage the original data of all devices to train a GAN model. Then we implement the GAN for augmenting minority classes on all devices. Afterwards, we again train the GAN based on the updated data. We iteratively alternate the above process until the model converges.

Design of Determining the Number of Augmented Data. The number of augmented data is an important variable in FED-CGU. Since the GAN model cannot be well trained initially, mass up-sampled data may lead to the accuracy degradation of the training model. Given insufficient up-sampled data, it cannot adequately exploit all devices' data. Therefore, we need to find the optimal up-sampling quantity. In this paper, we adopt the nondominated sorting genetic algorithm II (NSGA-II), a widely used multi-objective evolutionary algorithm to search for the up-sampling solution.

In our settings, the objectives of the searching are the optimal up-sampling quantities of different classes in each device. We denote S^* as the optimal up-sampling quantities and the chromosomes of S^* are shown as follows:

$$S^* = (Q, p_1, p_2, \ldots, p_n) \tag{2}$$

where Q denotes the number of augmented data and p_i denotes the proportion of the i-th device. The number Q and p_i are both a one-dimensional array with c parameters, where c is the number of classes.

To optimize the searching, we set the validation accuracy as the evaluating indicator. The objective function of the genetic algorithm is:

$$\arg\max_{S^*} \sum_{D^i \in D^*} E(w - \eta \nabla_w l(w; D^i \cup Aug^i(S^*))) \tag{3}$$

where D^i and Aug^i denote the local data and augmented data based on S^* in the i-th device, respectively. The parameter of w is the target classifier model, η is the learning rate and l is the loss function. We update w on the up-sampled data and evaluate the model on the validation data with the evaluation function E.

The Computational Cost of the Genetic Algorithm. The computational cost of the genetic algorithm consists of server-side and device-side cost. The largest cost on the server-side is fast non-dominated sorting, which has a time complexity of $O(N^2)$. In our experimental settings (in the evaluation section), it takes less than 3s in each round using NVIDIA GeForce GTX 1660. The computation on the device-side is the calculation of the solutions' fitness. The computation time on the device-side for each device is less than 10s in each round with NVIDIA GeForce GTX 1660.

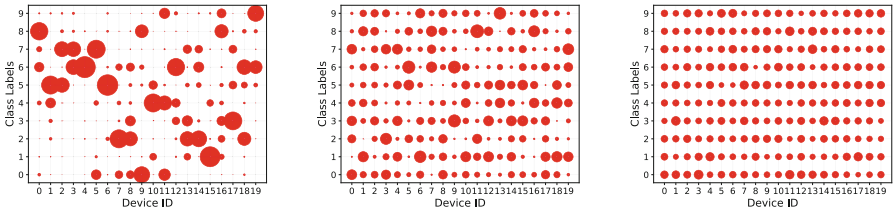
In addition, to accelerate the searching, we also incorporate prior knowledge, which is gradually increasing the number of augmented data, into the genetic algorithm. At the beginning of training, we just augment a small amount of data as the GAN model is not well trained. When the performance of the GAN model gets better, We gradually increase the amount of up-sampling. Applying this prior knowledge, we reduce the searching rounds of the genetic algorithm and thus save the computational power of the devices.

4 Evaluation

In this section, we empirically evaluate the proposed FED-CGU framework on benchmark data classification with general multi-class datasets. We first compare FED-CGU with FedAvg, Astraea and FAVOR. Then we validate the performance of different up-sampling methods. In addition, the effects of device sampling and the iterative GAN training strategy are also evaluated.

4.1 Experimental Setup

Datasets. The experiments are performed on the widely used Fashion-MNIST [24] and CIFAR-10 [10] datasets. We hold out 10% of the original training data as the validation dataset. The remaining training data are partitioned to the devices according to the following class imbalance settings.



(a) $\alpha = 0.1$ (b) $\alpha = 1$ (c) $\alpha = 10$

Fig. 3. The above figures visualize how data are distributed among 20 devices on different α values. The horizontal axis represents the device ID and the vertical axis represents the class labels. The sizes of circles indicate the number of samples per class allocated to each device. The smaller the α is, the more likely the devices hold samples from only one class.

Class Imbalance Settings. The total number of devices N is set to 20. We use the Dirichlet distribution to create the distribution of devices' training data, which is proposed by [14]. A hyper-parameter α is utilized to denote the level of class imbalance. As shown in Fig. 3, the smaller α is, the higher the level of class imbalance is.

Models. We modify the popular ResNet-18 [7] for CIFAR-10. The modified model consists of a 7×7 convolutional layer followed by a 3×3 max-pooling layer, four convolutional layers with 3×3 kernel size followed by a 2×2 average-pooling layer, a 512 fully connected layer, and finally a 10 class softmax output layer. The model for Fashion-MNIST consists of two convolutional layers with 5×5 kernel size followed by a fully connected layer.

4.2 Results of FED-CGU

Table 1. Results of FED-CGU.

Test Accuracy on Fashion-MNIST and CIFAR-10							
Fashion-MNIST				CIFAR-10			
	$\alpha = 0.1$	$\alpha = 1$	$\alpha = 10$		$\alpha = 0.1$	$\alpha = 1$	$\alpha = 10$
FedAvg	64.63%	72.14%	78.26%	FedAvg	60.24%	67.51%	73.10%
FAVOR	77.60%	82.91%	85.64%	FAVOR	64.74%	76.49%	78.58%
Astraea	78.69%	82.17%	85.15%	Astraea	70.34%	76.78%	80.31%
two-step	70.40%	76.07%	81.35%	two-step	65.12%	72.21%	78.34%
FED-CGU	81.04%	83.31%	85.79%	FED-CGU	75.74%	80.45%	82.81%

We compare our proposed FED-CGU with FedAvg, FAVOR, and Astraea on Fashion-MNIST and CIFAR-10 datasets. We also compare FED-CGU with a proposed two-step baseline, in which we first train a GAN model and then utilize the GAN to augment data once. Table 1 shows the results on Fashion-MNIST after 40 training rounds and CIFAR-10 after 60 training rounds.

According to the results, our proposed FED-CGU reaches the highest accuracy on the Fashion-MNIST and CIFAR-10 datasets on various α settings. In addition, FED-CGU gets even better results than FAVOR and Astraea when $\alpha = 0.1$ (achieves up to 11% higher accuracy). The reason is that, compared with the two strategies, FED-CGU utilizes the up-sampling to release class imbalance in each device during the training. Since the level of the class imbalance is reduced through up-sampling, the convergence of GAN can be accelerated. Therefore, FED-CGU outperforms FAVOR and Astraea. FED-CGU also outperforms the two-step strategy. Since the class imbalance can affect the performance of training, the GAN of the two-step strategy cannot be well trained simply on original data. Thus using such a GAN model in the two-step strategy for up-sampling introduces much noise to the devices. In the ablation experiments of Sect. 4.5, we deeply discuss the two-step strategy (train the GAN once) and iterative strategy.

4.3 Performance of Different Up-sampling Methods

In this subsection, we compare the performance of using a global GAN model for up-sampling with using local up-sampling methods in FL. We evaluate the effects of different up-sampling methods under various levels of class imbalance.

In this experiment, we first utilize up-sampling to supplement the devices' data and then train the model in federated learning. We compare 6 up-sampling methods (i.e. using global GAN for up-sampling, using local GAN for up-sampling, flipping, scale, rotation and shifting). Since common local up-sampling methods (flipping, scale, rotation and shifting) rely on original data, we set the minimum number of each class to 50. With regard to the GAN-based up-sampling, we pretrain a GAN model and use the trained GAN model to perform up-sampling. The local GAN is trained on every single device's local data and the global GAN is trained on all devices' data. The results are shown in Table 2.

Table 2. Test accuracy of different up-sampling methods.

Test Accuracy on CIFAR-10							
	$\alpha = 0.1$	$\alpha = 1$	$\alpha = 10$		$\alpha = 0.1$	$\alpha = 1$	$\alpha = 10$
Global GAN	67.2%	73.3%	75.5%	Scale	59.1%	70.9%	75.7%
Local GAN	60.1%	70.4%	74.6%	Rotation	58.1%	71.2%	74.2%
Flipping	58.7%	70.6%	74.7%	Shifting	57.1%	71.0%	73.8%

We first compare the effects of using a global GAN for up-sampling with other up-sampling approaches. Under various levels of class imbalance, using the global GAN model for up-sampling achieves the highest effect. When the level of class imbalance is high ($\alpha = 0.1$), using global GAN for up-sampling performs significantly better than other methods (accuracy is 7.1%–10.1% higher than other methods). The significant difference between global GAN and other approaches is that the data generated by the global GAN model leverages information from other devices. Using the global GAN model for up-sampling can introduce information of other devices' data to the local device. When the data are nearly balanced, we only need to augment fewer data to balance the class distribution, and less information is introduced by the global GAN. This can explain the results of $\alpha = 10$. When the degree of imbalance increases, more up-sampled data are needed and more information is introduced. Therefore, using the global GAN for up-sampling obtains the maximum advantage over local methods at $\alpha = 0.1$.

We then discuss the results of common up-sampling methods (such as rotation, flip, etc.) and using local GAN for up-sampling. When $\alpha = 0.1$, the performance of the local GAN is slightly higher than that performed by other local up-sampling methods. This is because common up-sampling methods only utilize the information of data with a certain label when augmenting the data with a such label. In this extreme imbalanced case ($\alpha = 0.1$), common up-sampling methods need to supplement a large amount of data, which can easily make the model overfit. On the other hand, the training of the GAN model utilizes all the data with various labels. As information of data with all the labels is utilized, data generated by the GAN model is better than that by common up-sampling methods.

4.4 Effects of Device Sampling

In this subsection, we verify the opinion that applying device sampling can improve the performance of up-sampling. We first compare directly implementing up-sampling, with implementing both up-sampling and device sampling. We regard the gradients of centralized learning as the ground truth and calculate the similarity between gradients of federated learning with centralized ones. The detailed experimental results are shown in Fig. 4. We find that, with device sampling, the similarity between centralized gradients and federated gradients is smaller than that without device sampling under the same level of class imbalance.

We also compare the communication cost for our complementary sampling and random sampling to reach a target accuracy rate. Experimental results in Fig. 5 show the communication cost of different up-sampling curves on Fashion-MNIST and CIFAR-10. Results show that complementary sampling is actually more communication-efficient than random sampling. When $\alpha = 0.1$, the communication consumption of training a CNN on CIFAR-10 using random sampling to reach 70% accuracy is 1795.5 MB whereas complementary sampling uses merely 215 MB. Our complementary sampling also achieves a 57.83% reduction to reach 75% accuracy on Fashion-MNIST when $\alpha = 0.1$.

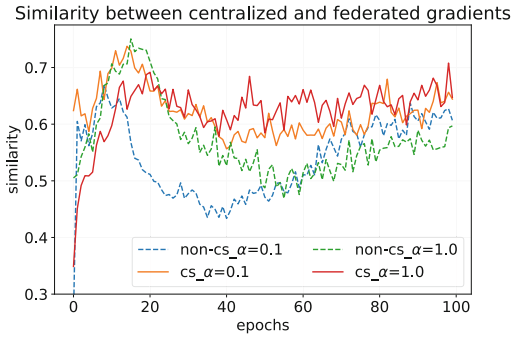
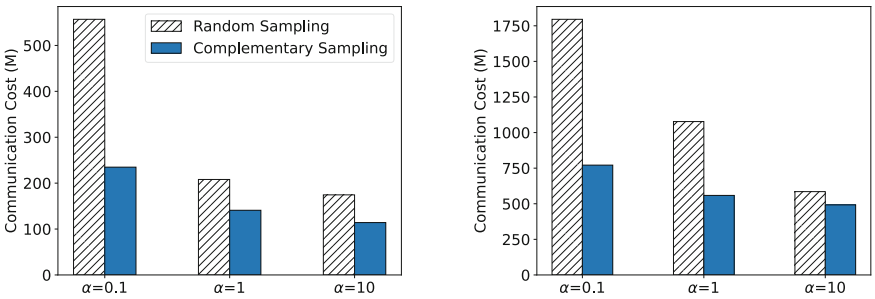


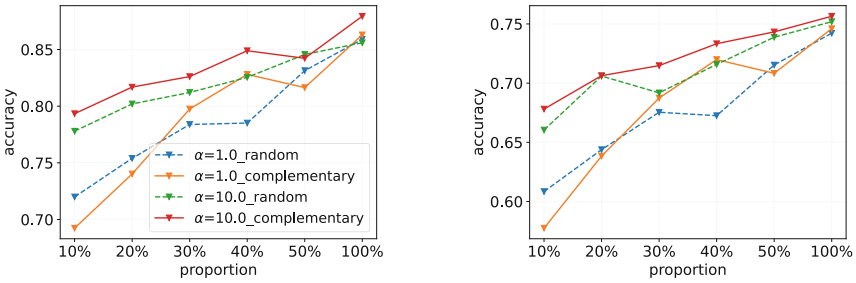
Fig. 4. Comparison of directly sampling devices and sampling devices after up-sampling on CIFAR-10. The curves of 'non-cs' and 'cs' show the results of directly using up-sampling and using both up-sampling and device sampling, respectively.



(a) Fashion-MNIST. (b) CIFAR-10.

Fig. 5. Communication cost to reach a target accuracy for complementary sampling and random sampling.

We then adjust the proportion of selected devices in each round. As shown in Fig. 6, our complementary sampling achieves better performance, especially when the proportion equals 30%–40%. Since complementar0y devices have information that other devices do not have, they can better help each other. This is the reason why our method is superior to random sampling. When all devices participate in each round, our method is the same as random sampling in practice. It should be noted that if there are very few devices selected in each round, the accuracy of complementary sampling is sometimes not as good as random sampling. This is because if only few devices are selected in each round, it can be difficult for some devices to find complementary devices.



(a) Fashion-MNIST (b) CIFAR-10

Fig. 6. Effects of the sampling proportions on Fashion-MNIST and CIFAR-10

4.5 Comparison of the Two-Step and Iterative Strategies

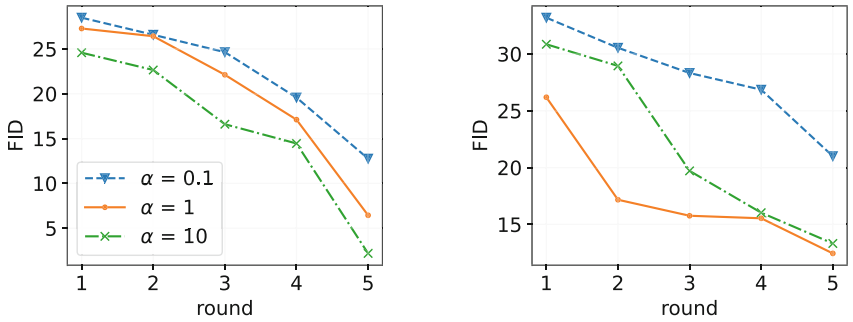
In this part, we compare the two strategies of up-sampling: (1) the two-step strategy: train the GAN first and then perform up-sampling once; (2) the iterative strategy: simultaneously perform up-sampling and train the GAN in an iterative manner.

Performance of Iteratively Training GAN. We first validate the assumption that training GAN and up-sampling alternately in an iterative manner can improve the performance of GAN. We utilize the Frechet Inception Distance score [8] of GAN to show the GAN's performance.

$$FID = \|\mu_x - \mu_g\|_2^2 + TR(\Sigma_x + \Sigma_g - 2(\Sigma_x\Sigma_g)^{\frac{1}{2}}) \tag{4}$$

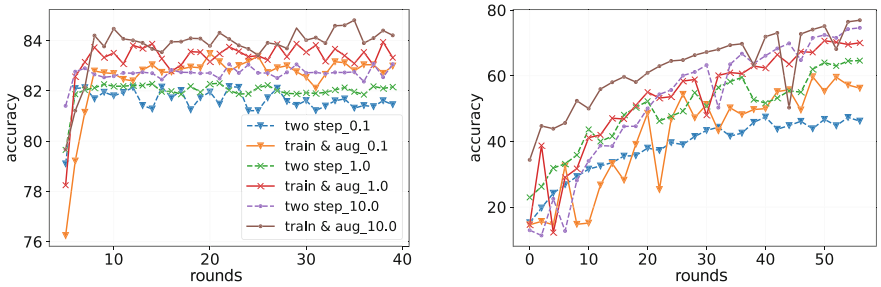
The Frechet Inception Distance score calculates the distance between the real data distribution x and the generated distribution g based on the mean and covariance matrix. Lower scores indicate that the generated data are closer to the original data. We validate the GAN during the training by the FID score [8]. As shown in Fig. 7, the quality of data generated by the GAN models is getting better after each iteration in all scenarios.

Test Accuracy of Using GAN Trained in Different Strategies. We then compare the test accuracy of the classifier with the two-step baseline and our iterative strategy. According to the results shown in Fig. 8, it is better to iteratively perform up-sampling and train the GAN. When the level of class imbalance is high ($\alpha = 0.1$), iteratively performing up-sampling and training the GAN achieves significantly higher accuracy than the two-step strategy. The reason is that, in the two-step strategy, the GAN model cannot be well trained in the presence of class imbalance. Data generated by the GAN of the two-step strategy may have much noise that can degrade the accuracy. In the iterative strategy, the class imbalance is gradually balanced during training, and finally, the GAN model is trained in the IID scene.



(a) Fashion-MNIST. (b) CIFAR-10.

Fig. 7. Performance of GAN during the training on Fashion-MNIST and CIFAR-10.



(a) Fashion-MNIST. (b) CIFAR-10.

Fig. 8. Test accuracy with different GAN training strategies on Fashion-MNIST and CIFAR-10. The numbers at the end of the legend $(0.1, 1.0, 10)$ are the values of α, which indicates the level of class imbalance.

Note that the initial training process of the iterative strategy is more oscillatory than that of the two-step strategy when the level of class imbalance is high. Since the GAN is not well trained at the beginning of training, the iterative strategy does not augment much data. As the training process proceeds, the iterative strategy augments more and gets better performance.

5 Conclusion

Class imbalance, one of the challenges in FL, can degrade the performance of the final model. To address this problem, we propose the first collaborative GAN-based up-sampling strategy in FL that allows devices to help each other without leaking their private data. To accelerate the convergence of the GAN model, we design a device sampling strategy that selects complementary devices. In addition, we adopt a genetic algorithm to dynamically determine the number of augmented data for each device. Experimental results on Fashion-MNIST and CIFAR-10 indicate that FED-CGU outperforms the state-of-the-art algorithms.

Acknowledgements. This work was supported in part by National Natural Science Foundation of China under Grant No. 61976012.

References

1. Chawla, N.V., Lazarevic, A., Hall, L.O., Bowyer, K.W.: SMOTEBoost: improving prediction of the minority class in boosting. In: Lavrač, N., Gamberger, D., Todorovski, L., Blockeel, H. (eds.) PKDD 2003. LNCS (LNAI), vol. 2838, pp. 107–119. Springer, Heidelberg (2003). https://doi.org/10.1007/978-3-540-39804-2_12
2. Dinh, C.T., Tran, N.H., Nguyen, T.D.: Personalized federated learning with moreau envelopes. In: Advances in Neural Information Processing Systems, vol. 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, 6–12 December 2020, virtual (2020)
3. Duan, M., Liu, D., Chen, X., Liu, R., Tan, Y., Liang, L.: Self-balancing federated learning with global imbalanced data in mobile systems. IEEE Trans. Parallel Distrib. Syst. **32**(1), 59–71 (2021). https://doi.org/10.1109/TPDS.2020.3009406
4. Fallah, A., Mokhtari, A., Ozdaglar, A.E.: Personalized federated learning with theoretical guarantees: a model-agnostic meta-learning approach. In: Advances in Neural Information Processing Systems, vol. 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, 6–12 December 2020, virtual (2020)
5. Goodfellow, I.J., et al.: Generative adversarial nets. In: Advances in Neural Information Processing Systems, vol. 27: Annual Conference on Neural Information Processing Systems, Montreal, Quebec, Canada 2014, 8–13 December 2014, pp. 2672–2680 (2014)
6. He, H., Garcia, E.A.: Learning from imbalanced data. IEEE Trans. Knowl. Data Eng. **21**(9), 1263–1284 (2009). https://doi.org/10.1109/TKDE.2008.239
7. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, 27–30 June 2016, pp. 770–778. IEEE Computer Society (2016)
8. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: GANs trained by a two time-scale update rule converge to a local nash equilibrium. In: Advances in Neural Information Processing Systems, vol. 30: Annual Conference on Neural Information Processing Systems 2017, Long Beach, CA, USA, 4–9 December 2017, pp. 6626–6637 (2017)
9. 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, ICML 2020. Proceedings of Machine Learning Research, 13–18 July 2020, Virtual Event, vol. 119, pp. 5132–5143. PMLR (2020)
10. Krizhevsky, A.: Learning multiple layers of features from tiny images. Technical report (2009)
11. Li, Q., He, B., Song, D.: Model-contrastive federated learning. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, 19–25 June 2021, pp. 10713–10722. Computer Vision Foundation / IEEE (2021)
12. Li, X., Huang, K., Yang, W., Wang, S., Zhang, Z.: On the convergence of FedAvg on non-IID data. In: 8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, 26–30 April 2020 (2020)

13. Lim, S., Kim, I., Kim, T., Kim, C., Kim, S.: Fast autoaugment. In: Advances in Neural Information Processing Systems, vol. 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, Vancouver, BC, Canada, 8–14 December 2019, pp. 6662–6672 (2019)
14. Lin, T., Kong, L., Stich, S.U., Jaggi, M.: Ensemble distillation for robust model fusion in federated learning. In: Advances in Neural Information Processing Systems, vol. 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, 6–12 December 2020, virtual (2020)
15. López, V., Fernández, A., Moreno-Torres, J.G., Herrera, F.: Analysis of preprocessing vs. cost-sensitive learning for imbalanced classification. open problems on intrinsic data characteristics. Expert Syst. Appl. **39**(7), 6585–6608 (2012)
16. McMahan, B., Moore, E., Ramage, D., Hampson, S., y Arcas, B.A.: Communication-efficient learning of deep networks from decentralized data. In: Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS 2017, 20–22 April 2017, Fort Lauderdale, FL, USA. Proceedings of Machine Learning Research, vol. 54, pp. 1273–1282. PMLR (2017)
17. Mirza, M., Osindero, S.: Conditional generative adversarial nets. CoRR abs/1411.1784 (2014)
18. Odena, A., Olah, C., Shlens, J.: Conditional image synthesis with auxiliary classifier GANs. In: Proceedings of the 34th International Conference on Machine Learning, ICML 2017. Proceedings of Machine Learning Research, Sydney, NSW, Australia, 6–11 August 2017, vol. 70, pp. 2642–2651. PMLR (2017)
19. Seiffert, C., Khoshgoftaar, T.M., Hulse, J.V., Napolitano, A.: Rusboost: a hybrid approach to alleviating class imbalance. IEEE Trans. Syst. Man Cybern. Part A **40**(1), 185–197 (2010)
20. Smith, V., Chiang, C., Sanjabi, M., Talwalkar, A.S.: Federated multi-task learning. In: Advances in Neural Information Processing Systems, vol. 30: Annual Conference on Neural Information Processing Systems 2017, Long Beach, CA, USA, 4–9 December 2017, pp. 4424–4434 (2017)
21. Sun, Y., Kamel, M.S., Wong, A.K.C., Wang, Y.: Cost-sensitive boosting for classification of imbalanced data. Pattern Recognit. **40**(12), 3358–3378 (2007)
22. Wang, H., Kaplan, Z., Niu, D., Li, B.: Optimizing federated learning on non-IID data with reinforcement learning. In: 39th IEEE Conference on Computer Communications, INFOCOM 2020, Toronto, ON, Canada, 6–9 July 2020, pp. 1698–1707. IEEE (2020)
23. Wang, L., Xu, S., Wang, X., Zhu, Q.: Addressing class imbalance in federated learning. In: Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, 2–9 February 2021, pp. 10165–10173. AAAI Press (2021)
24. Xiao, H., Rasul, K., Vollgraf, R.: Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms. CoRR abs/1708.07747 (2017)
25. Zhao, Y., Li, M., Lai, L., Suda, N., Civin, D., Chandra, V.: Federated learning with non-IID data. CoRR abs/1806.00582 (2018)
26. Zhou, Z., Liu, X.: Training cost-sensitive neural networks with methods addressing the class imbalance problem. IEEE Trans. Knowl. Data Eng. **18**(1), 63–77 (2006)