Federated Learning with Generative Adversarial Network to Improve Training on Non-i.i.d Data

Candidate Name:	Zhao Jiayi
School or Department:	Department of Computing
Faculty Mentor:	Prof. Guo Song
Degree Applied:	MSc in Information Technology
Degree by:	The Hong Kong Polytechnic University
The Date of Defence:	December, 2021

Abstract

Federated Learning has been a novel privacy-preserved distributed Machine Learning framework since it was proposed in 2017. However, statistical heterogeneity between client's local data may decrease global model performance of even make federated learning algorithm non convergence at model parameter aggregation section. This work focus on migrating the impact of non_iid data have on FedAvg algorithm, using simulation data generated from Multi_GAN for data augmentation. Considering privacy guarantee of federated learning, the discriminators will be trained based on local dataset so that raw data exchange is prohibited between clients and server. Non_iid assumption is the major reason of the data distribution drift between clients, and a multi_path generator is implemented on server side to cover multimodel distribution of discriminators in GAN training.

To perform a quantitative analysis on the influence of non_iid data have on model performance in federated learning algorithm, Dirichlet distribution function is implemented in data preprocess step with a tunable parameter non_iid_alpha in chapter 2. Experiments on synthesize dataset shows ideal performance of multi_path generator on multimodel distributed training dataset, and a visualization of the generated images from multi_path generator GAN in non_iid dataset is shown in chapter 3. Chapter 4 provide the global model performance evaluation, showing the improvement of classification accuracy on full batch gradient descent algorithm from 0.54 to 0.90, as well as in FedAvg algorithm which test accuracy increase from 0.48 to 0.86. A convergence analysis on local training epoch of a specific client is also given, which can be another insight evidence on the performance improvement of GAN based data augmentation to federated learning under non_iid assumption.

Contents

CHAPTER 1 INTRODUCTION **1**

1.1 Federated Learning Algorithm 1

1.2 Core Challenge 2

1.3 Survey of Related Work 2

1.4 Innovative Method 4

1.5 Methodology 4

CHAPTER 2 ANALYSIS ON HETEROGENEOUS DATA **7**

2.1 Heterogeneity in Federated Learning 7

2.1.1 Heterogeneity Database 7

2.1.2 System Heterogeneity and Statistical Heterogeneity 7

2.1.3 Classification of Statistical Heterogeneity 9

2.1.4 Performance Evaluation under Non_iid Data 12

2.2 Dirichlet Data Partition and Visualization 13

2.2.1 Dirichlet Distribution 13

2.2.1 Visualization on Mock Dataset 14

2.3 Performance Evaluation under Dirichlet Distribution 22

2.3.1 Performance Evaluation on MNIST Dataset 22

2.3.2 Performance Evaluation on Cifar10 Dataset 24

CHAPTER 3 GAN BASED DATA AUGMENTATION **28**

3.1 Data Augmentation in Federate Learning 28

3.1.1 Model Based Methodology on Migrating Non_iid Clients 28

3.1.2 Data Based Methodology on Migrating Non_iid Clients 29

3.2 GAN Training on Distributed Framework 30

3.2.1 Multi_path Generator GAN on Synthesize Dataset 30

3.2.2 Multi_path Generator GAN on iid MNIST Dataset 32

3.2.3 Multi_path Generator GAN on Non_iid MNIST Dataset 36

3.2.4 Multi_path Generator GAN on Non_iid Cifar 10 Dataset 41

3.2.5 Training Loss Analysis in Multi_path Generater GAN 45

CHAPTER 4 FEDERATED LEARNING USING DATA AUGMENTATION..... 48

4.1 Data Augmentation using Generated Image..... 48

4.2 Performance Evaluation on Augmented Dataset..... 50

4.3 Performance Evaluation on Local Model..... 54

CONCLUSION..... 57

REFERENCE..... 58

Chapter 1 Introduction

1.1 Federated Learning Algorithm

In the past decades, researchers who have been deeply cultivated in Computer Science have witnessed the rapid development of Machine Learning in the field of Artificial Intelligence. In its subdivisions, such as Computer Vision(CV), Natural Language Processing(NLP) and Speech Recognition(SR), the success of machine learning is based completely on the foundation of the development on Deep Neutral Networks(DNN).

However, if we go into the bottom and try to figure out the reason why deep learning systems can complete so many tasks that are supposed to be "impossible" for human ourselves in such subdivision areas, for example a face recognition application of which test accuracy reaches the level required by the functional standard, is based on huge amount of data like millions of labeled face images to be used on model training. The accuracy on machine learning depends directly on the quantity of training data, while the difficulty of obtaining high-quality and well-labeled training data is gradually increasing.

Legislator in many sovereign independent states or regions are trying to regulate the acquirement and management of data, through the introduction of policies and regulations to protect data security: the California Consumer Privacy Act (CCPA) was proposed in the United States in 2020, and the General Data Protection Regulation (GDPR)[1] was published by EU since 2017. Therefore in the foreseeable future, the difficulty for data analysis practitioners to collect and share data between different organizations will increase, especially in the areas like financial institutions and medical facilities.

As a result, machine learning researchers are urgently looking for an algorithm, which can balance the accuracy of DNN while providing assurance fitting the policy of the confidential data. To be more specific, the training set in each client's local model will be strictly protected on the local database, while the critical parameters can be integrated through server and clients, which is exactly the core methodology of Federated Learning. McMahan[2] completed the foundation work of federated learning in 2017. His theoretical

scheme and model architecture have been the baseline for relevant works up to now.

1.2 Core Challenge

However, with the industrialization of federated learning technology, some practical problems still need to be solved by researchers.

Considering the reality situation of linking millions of smart phones into the federated learning network, even if the data size of each model parameter is bytes, such a large-scale data transmission can still be considered as a bottleneck of training process[3].

Also consider the methodology of data privacy protection, researchers using Secure Multiparty Computation (SMC) and Differential Privacy (DP) to strengthen the privacy protection in federated learning[4]. However such algorithm is based on sacrificing model performance or reducing system efficiency, and it is still an issue to be solved on balancing the privacy and efficiency in federated learning algorithm.

Finally, similar to the distributed learning method, federated learning needs to cover the heterogeneity circumstance: both structural heterogeneity and statistical heterogeneity[5]. The structural heterogeneity include the hardware equipment of the clients, the network transmission capacity and whether the client has been authorized to join the federated learning network manually. While in statistical heterogeneity, in most environments data generation violates the assumption of Independent and Identical Distribution(i.i.d). Non-i.i.d data will increase the complexity of model and reduce the performance in federated learning[6].

1.3 Survey of Related Work

Federated Averaging algorithm (FedAvg) was proposed[2] based on the foundation work of distributed machine learning[7], which aims to only process the critical parameters exchange between clients and servers, such as the weight of Stochastic Gradient Descent (SGD). FedAvg algorithm can ensure the local data to be stored in client's own database independently, while the global model reach the guaranteed convergence. However, since

the FedAvg algorithm did not analysis deeply into the algorithm optimization on non-i.i.d data, the researchers gave the following updates in subsequential work.

Some researchers try to relieve the statistical heterogeneity status using distributed optimization method, especially using local SGD as optimizer[8]: Li T[9] proposed FedProx, a convergence guaranteed algorithm by adding a proximal term with a tunable parameter $\mu >= 0$. Wang J[10] proposed FedNova algorithm, which can adopt the aggregated weight automatically as well as offering a theoretical analysis on the convergence rate and bias of local SGD on non-i.i.d data. Reddi S[11] proposed FedOpt based on the study of adaptive optimization like Adagrad[12], Adam[13] and Yogi[14] subjecting to federated learning algorithm, and offer a proposed learning rate on both client and server side to reach a better accuracy under a given rounds. Similar study on key parameters, Deng Y[15] proposed algorithm DRFA considering a model parameter and a mixing parameter updating on client and server side independently, and the theoretical analysis showing this algorithm to be more robust as well as convergence guaranteed. As a summary work, He C[16] provided FedML offering an open library and benchmark for federated learning research.

Beside optimization algorithm study, Karimireddy S P[17] offered SCAFFOLD algorithm and proposed the conception “Client Drift” in the heterogeneous data distribution of federated learning at the first time. Similar to SCAFFOLD, Zhao Y[18] brings out the concept “earth mover’s distance” between clients in federated learning process, and declare a small subset of globally sharing data on all edge devices can improve training on non-i.i.d data. Both of these essay illuminated my research in someway. Reisizadeh A[19] proposed an Affifine Transformation method for the data stored on device named FLRA, and trying to improve training in another light.

Another way to manipulate data heterogeneity under the heuristic of “client drift” is called “client clustering”. Ghosh A[20] proposed IFCA to cluster clients before processing federated learning algorithm, and He C[21] proposed similar algorithm named FedFKT, focusing on the image processing with a small CNN training on edge side while a large CNN on server side, and built communication using knowledge distillation. Briggs C[22] proposed Federated Learning + Hierarchical Clustering (FL + HC) algorithm trying to cluster the clients based on unsupervised clustering algorithm before federated learning

algorithm, and convert non-i.i.d data distribution to IID data distribution. Besides unsupervised clustering method, the combination between reinforcement learning and federated learning has also been proposed, to increase the global model accuracy on non_iid data[35][36][37][38]. However, since the employment of client clustering algorithm on image recognition has been wildly analyzed, there is still a lack of investment on NLP, and in the mean time, communication cost is another important aspect that may be the bottle neck of federated learning processing, since complex algorithm always accompanied by large size of data transmission, instead of the weight of SGD only in the original algorithm FedAvg.

1.4 Innovative Method

Generative Adversarial Network(GAN) was proposed on 2014[23], which has been a well developed as well as mainstream algorithm to generate simulated data after years iteration[24]. There are many papers related to representative variants of GAN, such as Conditional GANs[25], CycleGAN[26], Wasserstein GAN[27] and DCGANs[28], and all of these algorithms as been well studied. The combination between GAN and distributed learning[29][30] or federated learning have also been proposed before[31][32][33][34].

However most of research focus on training GAN under the constrain of federated learning condition, and it is still a remaining unsolved technical problem using GAN as a simulated data generation to solve statistical heterogeneity in federated learning.

1.5 Methodology

The research was implemented based on the FedAvg algorithm as baseline[2].

Consider K clients with given private datasets and in each training epoch, a group of client controlled by fraction C was chosen randomly to participate in the federated learning process. C = 1 means all clients are chosen to participate in one training epoch during federated learning global round, so called full-batch gradient descent. The training process can be described as:

$$\min_{\omega \in \mathbb{R}} f(\omega) \quad where \quad f(\omega) := \frac{1}{n} \sum_{i=1}^{n} f_i(\omega), \quad and \quad f_i(\omega) \leftarrow \mathbb{L}(x_i, y_i; \omega)$$

In machine learning algorithm, $\mathbb{l}(x_i, y_i; \omega)$ can be considered as the loss function of optimization Stochastic Gradient Descent(SGD) on client (x_i, y_i) with model parameter ω, and the model is trying to minimize the global average training loss with a fixed learning rate η. We assert P_k as the set of index of data on client k with $n_k = |P_k|$, the formula above can be re-write as:

$$f(\omega) = \sum_{k=1}^{K} \frac{n_k}{k} F_k(\omega) \quad where \quad F_k(\omega) = \frac{1}{n_k} \sum_{i \in P_k} f_i(\omega)$$

The average gradient on local data can be calculated as g_k, which can be used to update global gradients ω_{t+1} at round t.

$$g_k = \bigtriangledown F_k(\omega_t) \quad and \quad \omega_{t+1} \leftarrow \omega_t - \eta \sum_{k=1}^{K} \frac{n_k}{n} g_k$$

In brief summary, optimization function SGD is guaranteed to be convergence by updating ω_{t+1} after epochs training between client and server, and the federated learning model will reach the balance of global accuracy and local accuracy after iterations update theoretically.

In typically Generative Adversarial Network(GAN)[23], the system was simultaneously trained on two models: a generator model G using Gaussian noise as original input and providing simulated data to discriminator D. Model D calculate the possibility whether sample from model G is true or false, while model G trying to maximize the possibility for model D make mistakes. So generally, generative adversarial network is mini-max game. The value function $V(G, D)$ with predefined noise variable $P_z(z)$ as well as stochastic gradient on each model are given below:

$$V(G,D):\min_{G}\max_{D}V(G,D)=\mathbb{E}_{x\sim P_{data}(x)}[logD(x)]+\mathbb{E}_{z\sim P_{z}(Z)}[log(1-D(G(z)))]$$

$$SGD \quad in \quad discriminator:\bigtriangledown_{\theta_d}\frac{1}{m}\sum_{i=1}^{m}[logD(x^{(i)})]+log(1-D(G(z^{(i)})))$$

$$SGD \quad in \quad generator:\bigtriangledown_{\theta_g}\frac{1}{m}\sum_{i=1}^{m}log(1-D(G(z^{(i)})))$$

In another foundation work combining GAN with distributed machine learning framework, MD-GAN[39] was proposed to train distribution GAN under conditional constrain, both origin distributed system and federated learning system. MD-GAN place generator on server side and discriminator on client side, while discriminator exchange is allowed between clients. MD-GAN implement the structure of GAN as a whole, and multiple GAN system was placed on both server side and client side. In MD-GAN, data transmission between clients is prohibited matching the private policy of federated learning.

Chapter 2 Analysis on Heterogeneous Data

2.1 Heterogeneity in Federated Learning

2.1.1 Heterogeneity Database

The proposal and development of the Federated Learning algorithm makes the traditional machine learning working platform expanded from the centralized data center to the remote and isolated data cluster: on the basis of the privacy assurance of the local data in edge devices, using distribution technologies such as big data technology and federated optimization algorithms, a globalized neural network model for the overall system which accuracy is not lower than that of a simple machine learning algorithms shall be obtained. Some edge devices of modern distributed networks such as mobile phones, sensors in the self-driving cars, HDFS from a business company or encrypted databases from a financial system, each of them are generating a large amount of data every single day, and to those data cluster with barely data exchange, is what we call a Information Island.

There are obvious differences between various isolated information islands, for example, the user portrait of a student whose major is computer science in the database of HSBC should be completely different from that of a professional lawyer in the same bank. Suppose there is a super-organization which is independent from all other major commercial banks, under the condition of hiding sensitive information such as user name or user ID, it is quite difficult for that organization to obtain the global model through traditional federated learning algorithm. And this is only one of the simple cases for the statistical heterogeneity in federated learning in practical use.

2.1.2 System Heterogeneity and Statistical Heterogeneity

The difference between clients in federal study can be divided into System Heterogeneity and Statistical Heterogeneity theoretically. In original federated learning algorithm, an independent client upload the local model parameters, which is trained based on its local data, to the server system. And the model parameters will be assigned back from the server to the client after a simple arithmetic average is processed with parameters

from all other clients, and this data exchange process shall be repeated until the global model reach convergence.

However a common phenomenon in practical use is that the client's local model is not uniformed, and obviously, the neural network parameters from different model structures cannot be averaged calculated. In addition, for the non-enterprise-type federated learning projects, considering the difference between client's hardware equipment (CPU and GPU), network connection status (LTE and wifi), and whether the client has authorized permission of their personal device to participate in the federated learning project, the number of local devices that actually participated in the federated learning project may only be 10%-50% of the total number. Some devices that have been participated in the algorithm may also have the chance of withdrawing from the project at any time. All those unexpected and emergency concerns that has been discussed above, can be considered as the system heterogeneity in federated learning.

The existence of this embarrassment will challenge the fault tolerance and robustness of the distributed system. While some new algorithms which have been proposed in recent two years provide an enlightening solution for solving system heterogeneity: non-uniformed system can be converged through Knowledge Distillation algorithm [40-43], and zero-shot learning algorithm[44-46] targeting at the situation where model have nearly very little training data. So that the federated system can train a neural network without dependence on large-scale training dataset, and can still achieve satisfactory convergence.

Considering another restrictive application environment focusing on the data distribution, such as the banking system example that has been discussed above, or an intelligence word prediction system in mobile phone - an edge device for every individual user. The system will recommend user the next word based on the current input word through a personalized recommendation algorithm, and the training of the text recommendation model is highly differentiated and private.

It can be clearly inferred that the data distribution across edge devices shall be very different, but there is also a potential statistical distribution law that can better fit the privacy data of multiple clients to form a global model[47]. This paradigm of generating independent data sets from different clients, so called statistical heterogeneity, violates the

i.i.d assumption of data distribution which is often used in distributed optimization problems. A distributed system based on non_iid data will increase a certain degree of complexity in modeling and convergence evaluation for traditional algorithm theory.

2.1.3 Classification of Statistical Heterogeneity

Back to federated learning algorithm, in training iteration t some clients that have be chosen according to the hyper-parameter fraction $C \in (0, 1)$ will upload their model parameters to global model ω_t. Under the original federated learning framework, the global model converges to a joint statistical model $f(\omega)$ based on the distributed learning algorithm. The non_iid data is the uneven distribution of data between different clients i and j, which can be present as $P_i \neq P_j$. The mathematical expectation of the model trained based on the client isolated local data does not conform to the data distribution of the local data:

$$\mathbb{E}_{D_k}\left[F_k(\omega)\right] \neq f(\omega)$$

Where $E_{D_k}[*]$ represent the mathematical expectation on local dataset D_k for client k, and $F_k(*)$ represent the local model for client k.

To be more specific, the non_iid hypothesis includes the following four states, all of which have practical meanings corresponding to real life[22]:

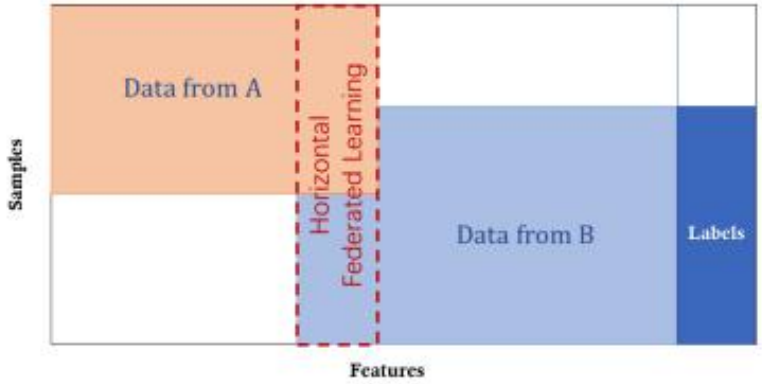
Feature Distribution: The exception of input features $P_i(x)$ is unevenly distributed among clients, which means that the input features of training data are distributed unevenly between clients.

Label Distribution: The exception of labels of training data $P_i(y)$ are not distributed among clients evenly, which means that the distribution of the labels of input data between clients is not evenly distributed, while the input data remains independent. For example, the data distribution between some client's training data $P_i(y_1) << P_i(y_2)$, while the distribution of local training data for another client is $P_i(y_1) >> P_i(y_2)$.

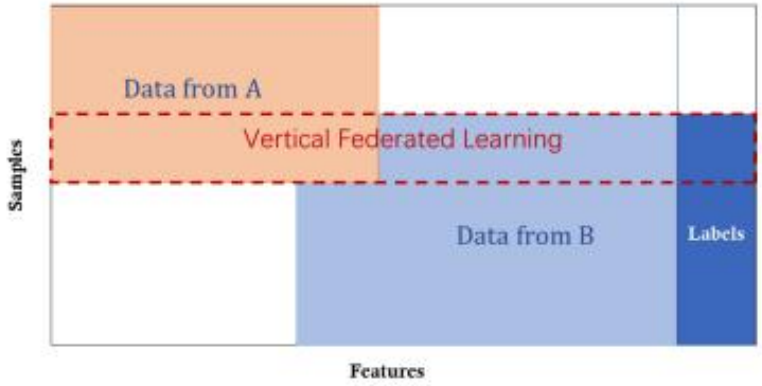
Concept Shift ($P_i(y|x)$): The input data with the same features is marked into different labels between different servers, this concept shift is raised because of system differences. For example, in the traditional image recognition task, the training data with particular feature is marked as y_1 in the $client_1$, and while the training data with the similar features is marked as y_2 in the $client_2$.

Concept Shift ($P_i(x|y)$): The input data with different features are marked with the same label between different clients. For another example using the traditional image recognition task discussed above, the training data with $feature_1$ is marked as label y_1 in the $client_1$, while the training data with $feature_2$ is also marked as y_1 in $client_2$.

Among these four status of non_iid data, label distribution is one of the most representative problems of the non_iid hypothesis of data distribution. In some general cases, focusing on the federated learning where training data between clients have same features but different labels, is called Horizontal Federated Learning, and for more details of the classification of different types of federated learning algorithm is shown in figure 1.



(a) Horizontal Federated Learning



(b) Vertical Federated Learning

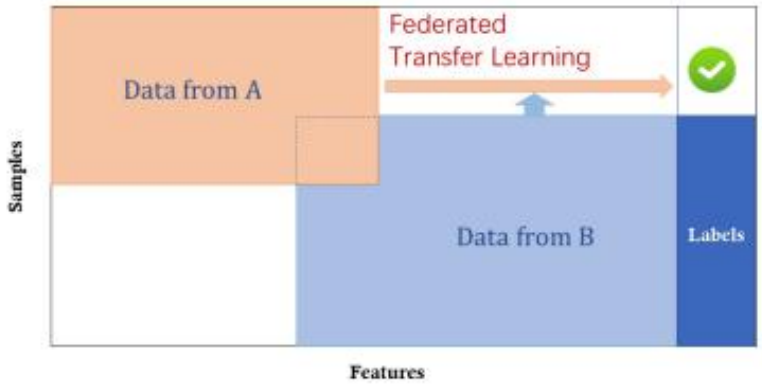


Figure 1. classification of different types federated learning algorithm

2.1.4 Performance Evaluation under Non_iid Data

The training data are defined as two different ways in data pre-processing stage in FedAvg algorithm, namely the standard iid hypothesis and non_iid hypothesis which is mutually exclusive to the former one.

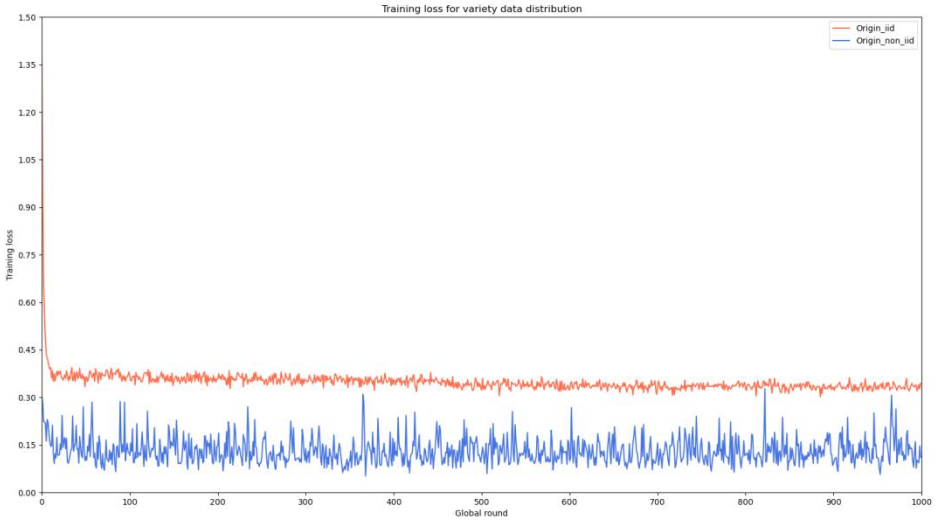


Figure 2(a). Training loss of origin Federated Learning algorithm

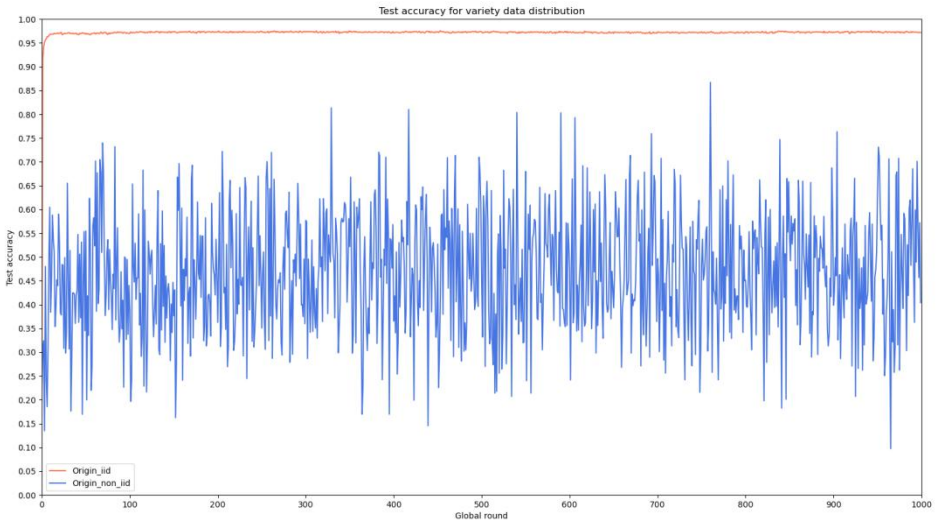


Figure 2(b). Test accuracy of origin Federated Learning algorithm

This pre-processing method can be intuitive in the model training process of federated learning. The influence of two different data distribution modes can be observed on training loss value and test accuracy curve: for original FedAvg training on iid distribution dataset, the training loss curve in figure 2(a) drop rapidly to the minimum value and goes steady as the training epoch increase, which indicate a quick convergence of the global model and the corresponding ideally test accuracy curve is shown in figure 2(b).

The loss curve for FedAvg training on non_iid dataset remains slightly vibration after hundreds of epochs training, and this phenomenon mainly because the divergence of the data distribution between different clients. It is difficult for FedAvg algorithm to calculate a global model that can cover every data diverge in the non_iid distribution status, which is also the major reason why test accuracy remains 0.45 - 0.55 as well as shaking severely in figure 2(b). The performance variation becomes the evidence that the original FedAvg algorithm cannot be the solution of the global model convergence of federated learning on non_iid data perfectly.

However, the only two extreme data distribution modes cannot regulate the degree of non_iid data distribution through hyper-parameters, which means it cannot quantify the influence of the degree of data distribution on federated learning in stages.

2.2 Dirichlet Data Partition and Visualization

2.2.1 Dirichlet Distribution

Dirichlet distribution is a continuous multivariate probability distributions controlled by a parameter vector “Alpha”, and it can be used in the data pre-process step to partition the training dataset in a verified way, using a hyper-parameter non_iid_alpha[48][49][50].

For a training dataset allocated with N labels, the allocation of labels follows a parameter vector q. Assuming that p represents the prior distribution of N labels of data, the Dirichlet distribution can be summarized as:

$$q \sim Dirichlet(\alpha p)$$

Where α represent a tunable hyper-parameter non_iid_alpha, and if the non_iid_alpha (>100) to be set in a larger value, the training data will be assigned evenly to each client according to the distribution of data label, which represent the homogeneous distribution. A smaller non_iid_alpha (<1) means the distribution of training data is non-uniform between clients, and since each client will hold part of training data in a overall dataset, only a small part of labels of the training data overlap between clients. In this way we can represent the week heterogeneous distribution and satisfies the assumption of label distribution discussed above.

2.2.1 Visualization on Mock Dataset

Suppose a virtual database has a total of 60,000 training data, which are respectively labeled as 10 labels, are assigned to 25 clients for federated learning through the Dirichlet distribution to be a simple simulation of training dataset like Cifar10. The visualization of Dirichlet data distribution is shown in figure 3.

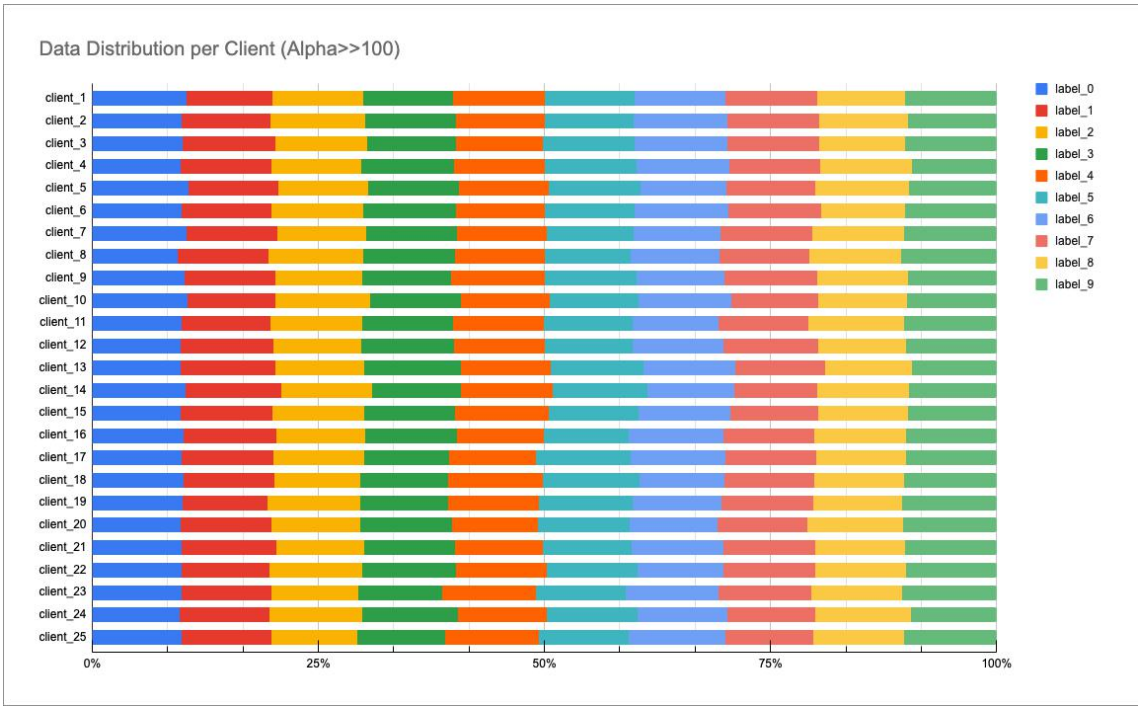


Figure 3(a). Data distribution per client on non_iid_alpha >> 100

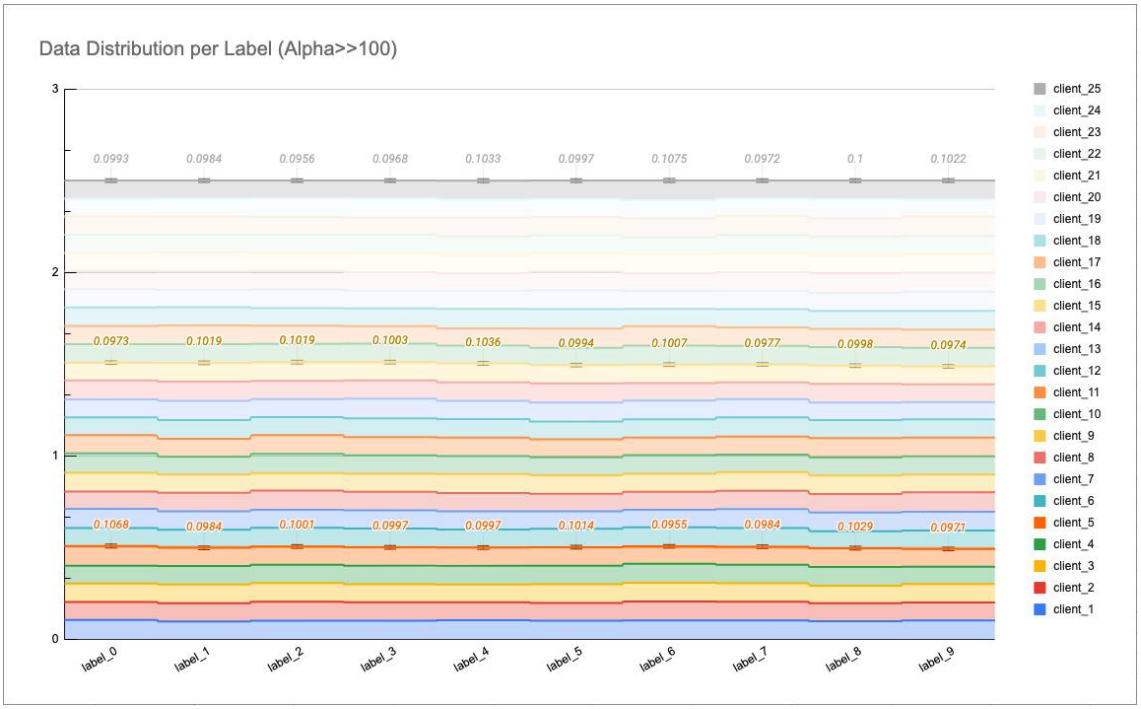


Figure 3(b). Data distribution per label on non_iid_alpha >> 100

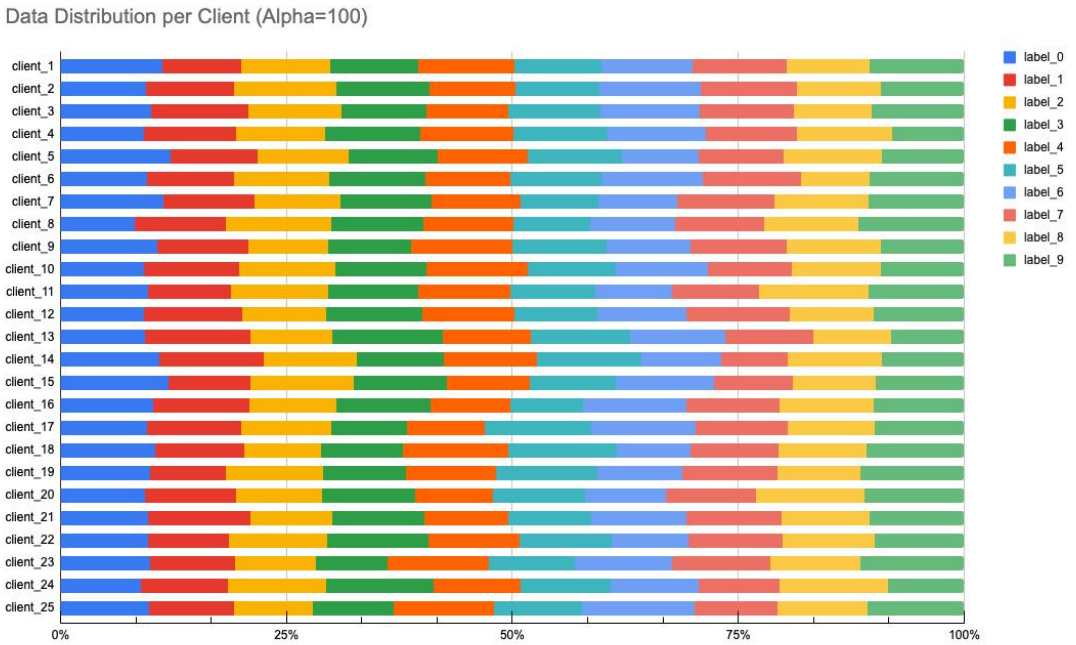


Figure 3(c). Data distribution per client on non_iid_alpha = 100

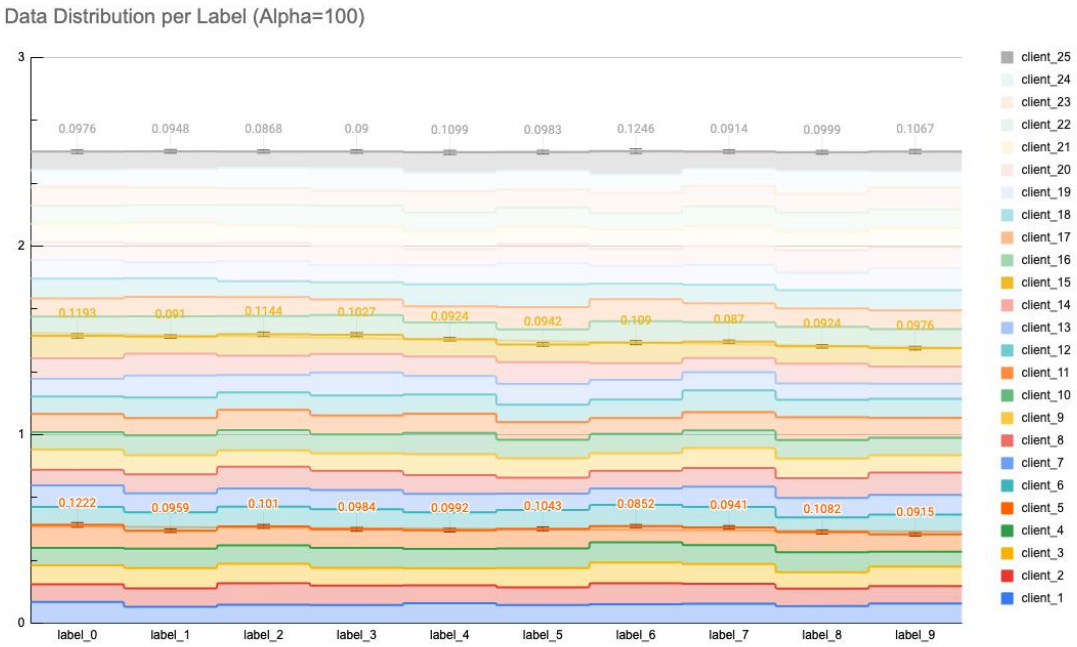


Figure 3(d). Data distribution per label on non_iid_alpha = 100

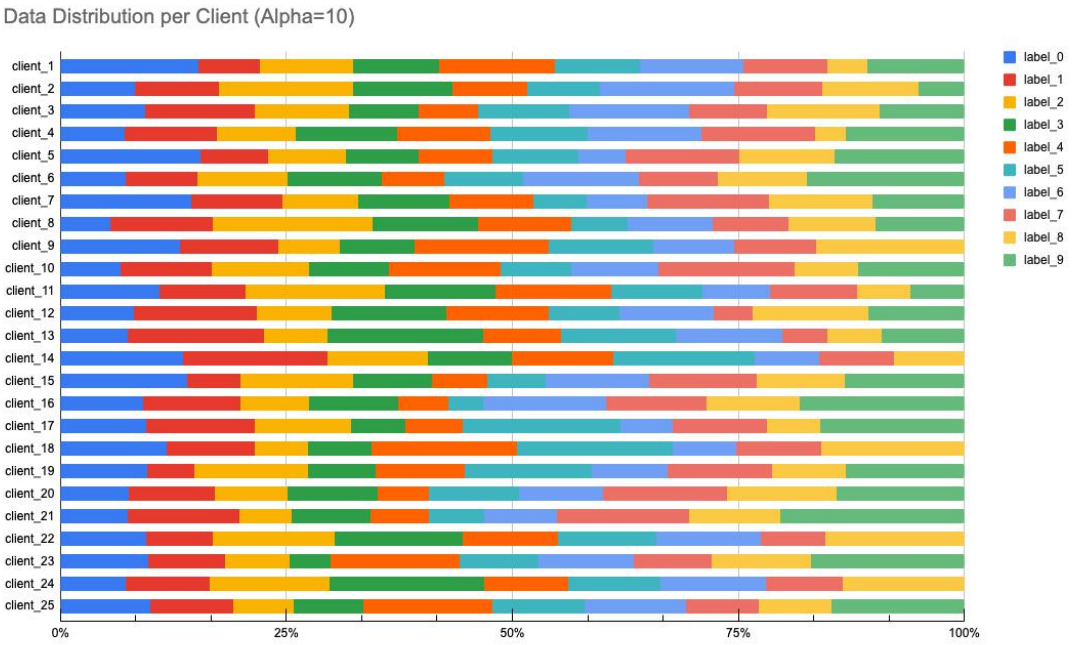


Figure 3(e). Data distribution per client on non_iid_alpha = 10

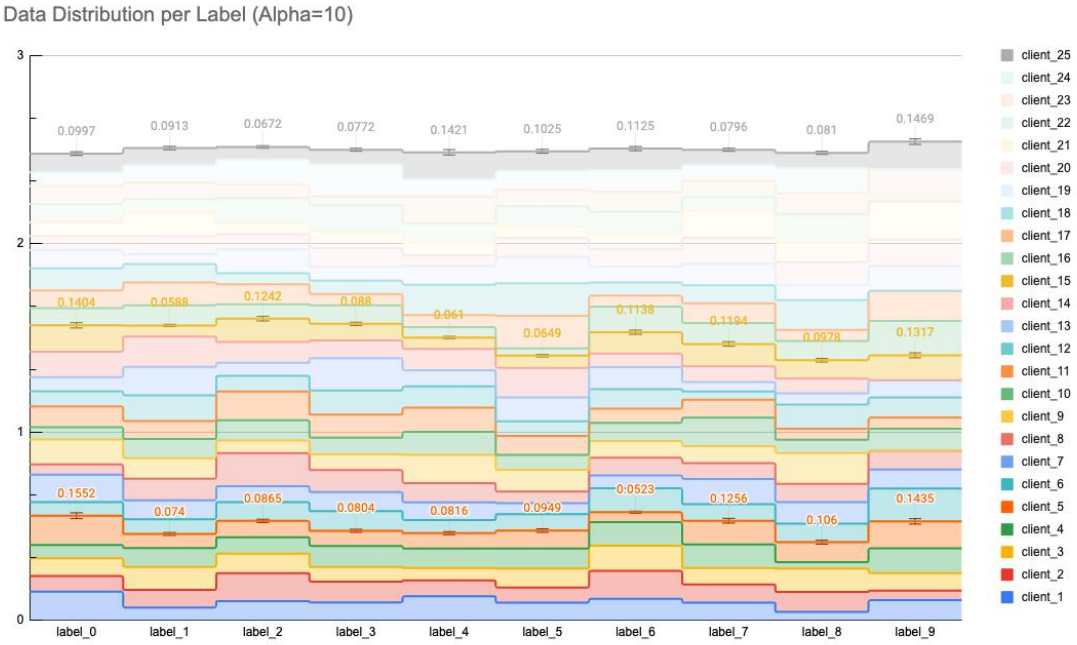


Figure 3(f). Data distribution per label on non_iid_alpha = 10

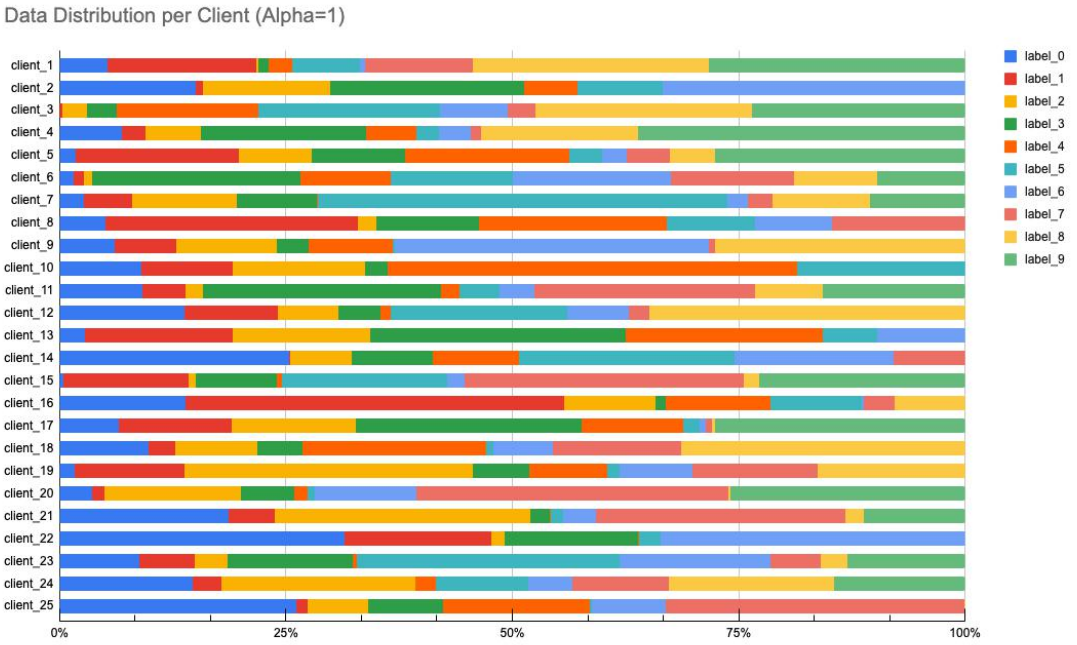


Figure 3(g). Data distribution per client on non_iid_alpha = 1

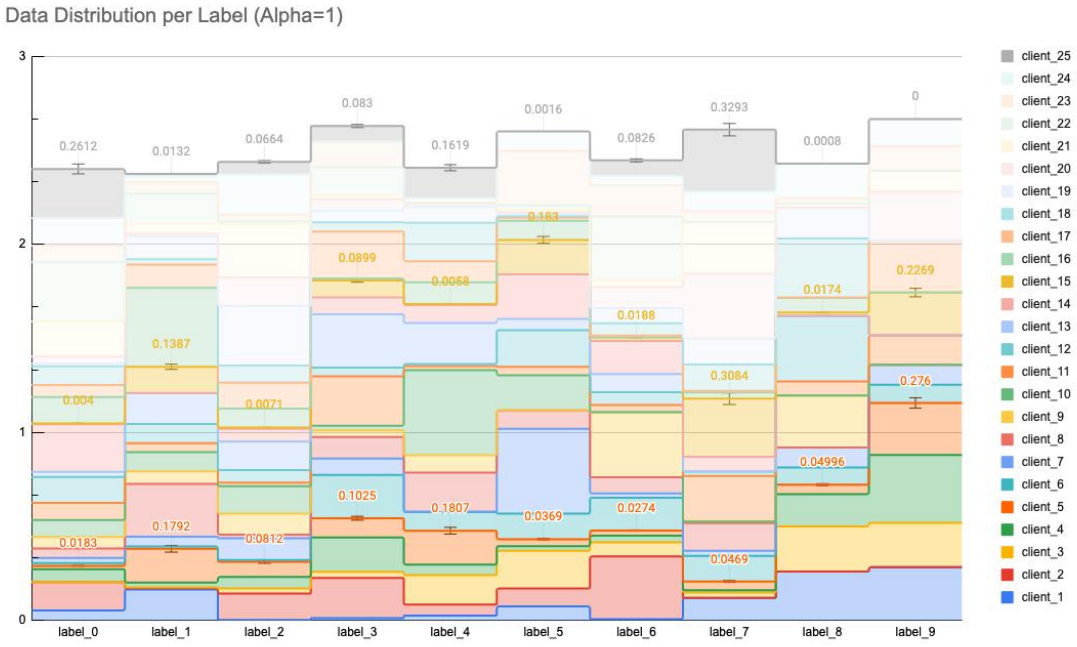


Figure 3(h). Data distribution per label on non_iid_alpha = 1

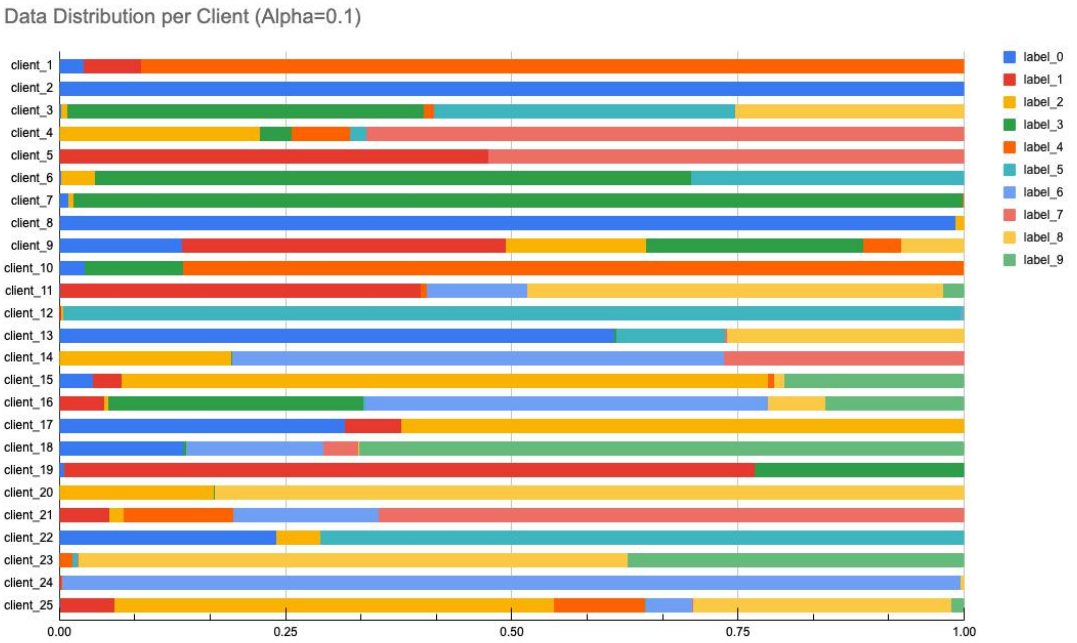


Figure 3(i). Data distribution per client on non_iid_alpha = 0.1

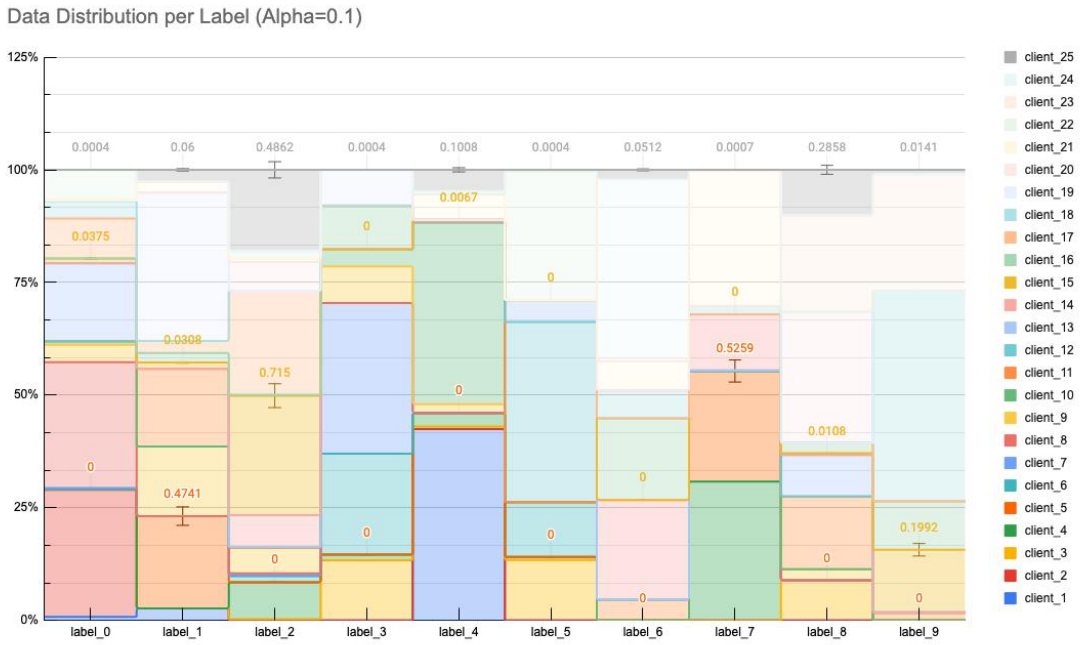


Figure 3(j). Data distribution per label on non_iid_alpha = 0.1

If the non_iid_alpha is set to be 0.01, it corresponds to the non_iid data distribution example shown in the FedAvg algorithm. In such case, each client's independent database can be assigned only one type of training data and have barely overlap between clients. This status represent the strong heterogeneous distribution of training data, and in traditional algorithms, the global model can hardly get convergence under this circumstance.

Also it should be claimed that the assignment of a single training data under Dirichlet distribution is randomized.

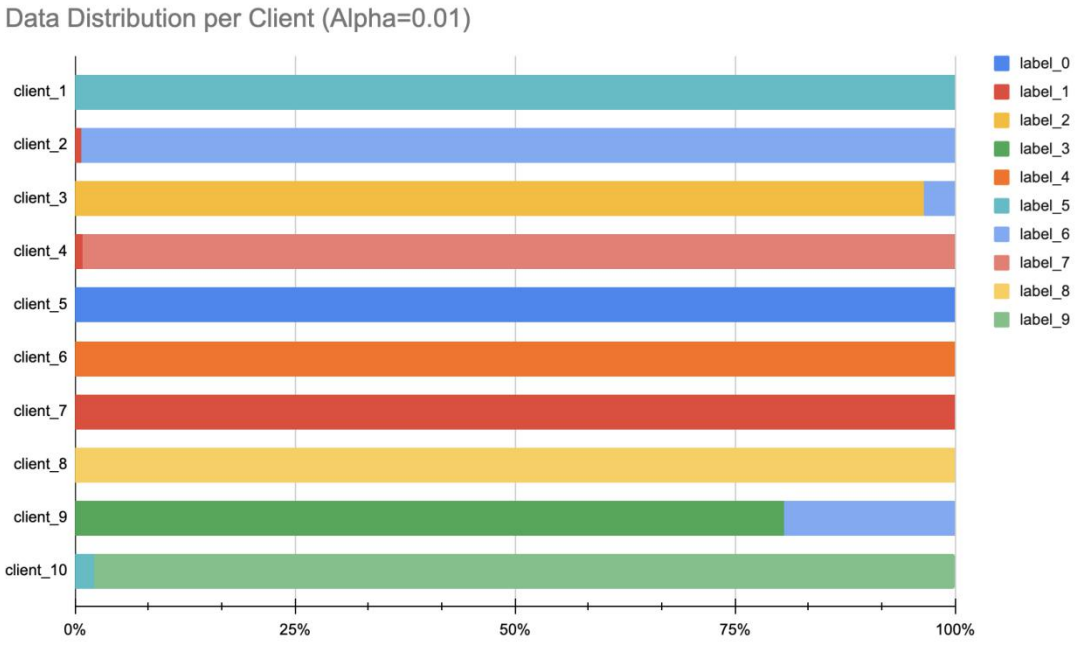


Figure 4(a). Data distribution per client on non_iid_alpha = 0.01

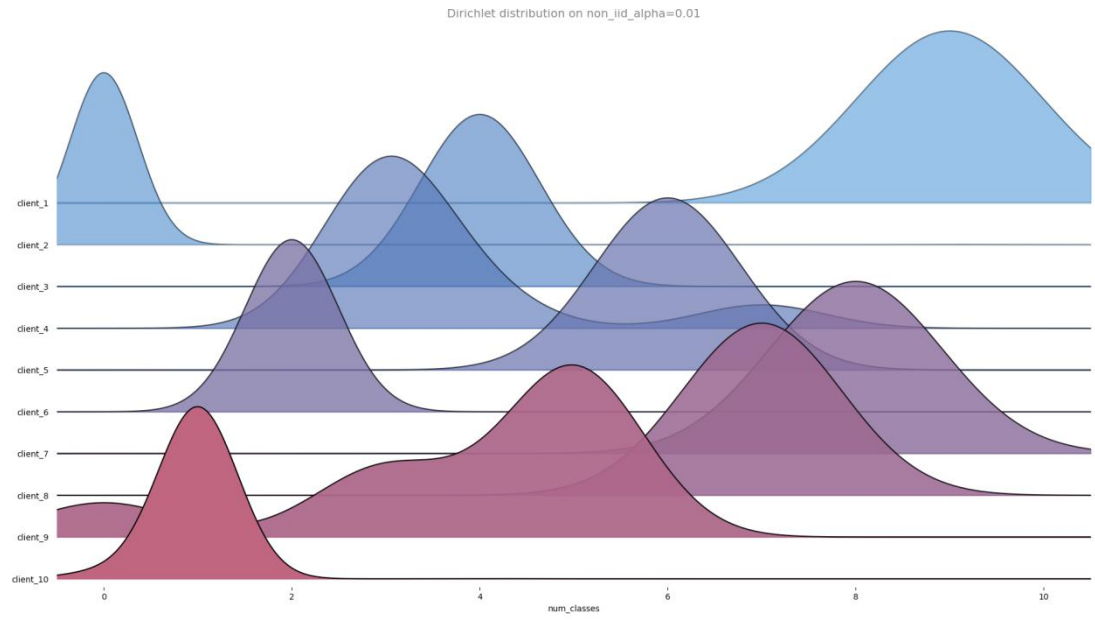


Figure 4(b). Data distribution per label on non_iid_alpha = 0.01

The sub-graph data distribution per client in figure 3 indicate the data distribution where training data are assigned into 25 independent clients, and non_iid_alpha is the

tunable hyper-parameter. The degree of iid is proportional increasing under the influence of the hyper-parameter non_iid_alpha. For a higher degree of non_iid assumption, some clients only hold part of the training data as expected, and this is consistent with the label distribution discussed above.

The sub-graph data distribution per label in figure 3 represent the data distribution status from the perspective of training data. The highlighted number indicate the normalized exact number of different labels of data in a single client after normalization: orange number represent Client_5, yellow number represent Client_15 and gray number represent Client_25. It can be intuitively observed that with non_iid_alpha decrease, the degree of the uneven distribution in training data which allocated to each client increase. Dirichlet distribution function achieve the quantitative processing from iid data to non_iid data, and to be more accurately, the degree of non_iid can be calculated by the variance of the number of training data, that a single client hold with different labels.

$$Variance = \frac{1}{n}\sum_{i=1}^{n}(x_i - x_n)^2$$

Variance	alpha >> 100	alpha = 100	alpha = 10	alpha = 1	alpha = 0.1
Client_5	10×10^{-6}	102.9×10^{-6}	1.1×10^{-3}	7.3×10^{-3}	0.045
Client_15	4.5×10^{-6}	119.2×10^{-6}	0.9×10^{-3}	12×10^{-3}	0.05
Client_25	12.5×10^{-6}	125.8×10^{-6}	0.7×10^{-3}	13.5×10^{-3}	0.026

Table 1. Variance of the training data number between clients

For the extreme case of non_iid_alpha = 0.01, it can be regarded as an instance of strongly non-identically distributed data distribution, where a single client only assigned by a single label of training data and the data distribution overlap between different clients can be hardly detected. According to the FedAvg algorithm, the local model of a single client is trained on its own independent database and no data exchange between clients and server is allowed because of the privacy restriction. Local model for each client would be unique and mutually exclusive, which is the critical reason why FedAvg algorithm can not reach the global convergence by uploading model parameters to the server and perform simple aggregation under non_iid assumption.

2.3 Performance Evaluation under Dirichlet Distribution

2.3.1 Performance Evaluation on MNIST Dataset

Theatrical analysis can be confirmed by federated learning performance characterization. In order to reflect the influence of different hyper-parameter non_iid_alpha to the global model in federated learning, a serious of experiments based on FedAvg algorithm is reproduced, and this performance evaluation will also be the baseline to the following work.

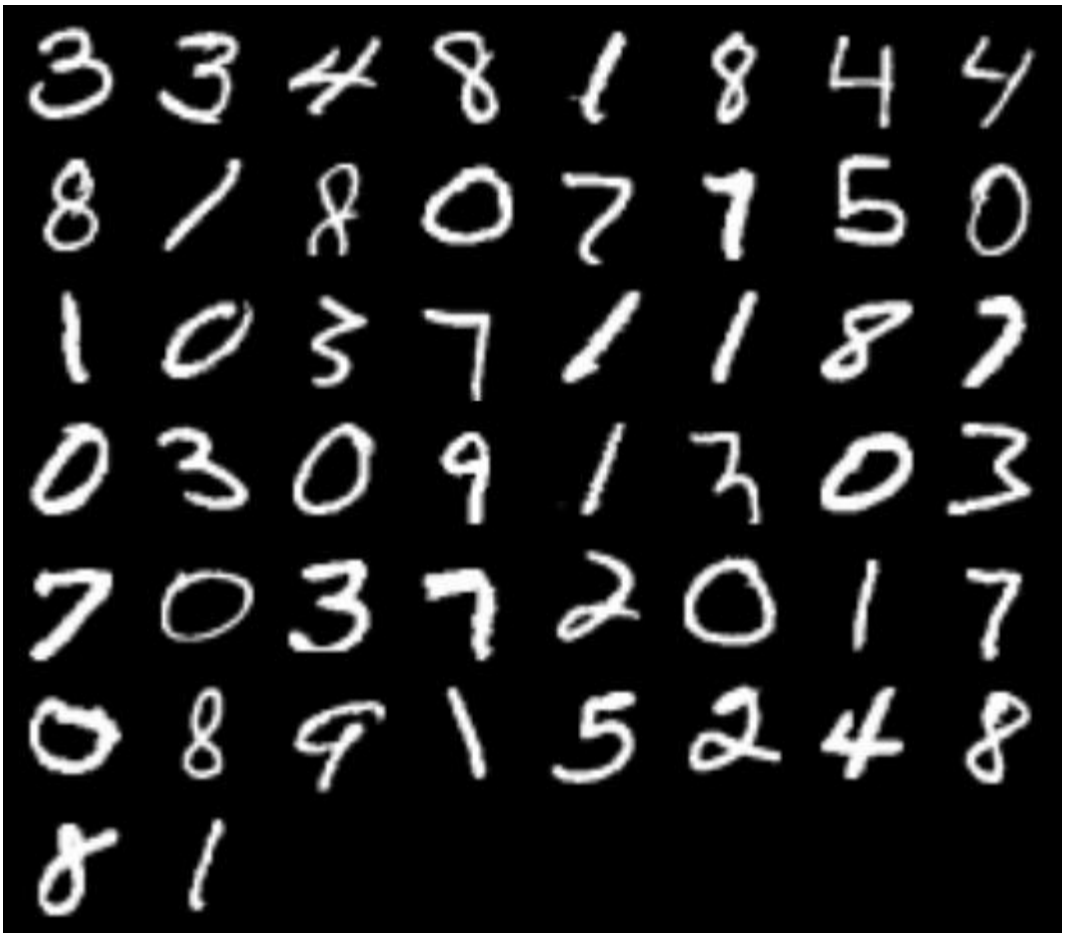


Figure 5. MNIST dataset training sample

The MNIST dataset is a widely used dataset for machine learning and it is also one of the most basic training sets in computer version. MNIST dataset has 60,000 training data and 10,000 test data, all the image are Arabic numerals written by authors with different

writing habits. The image is single channel with the size 28 * 28, and the training data has been pre-processed and formatted, which can be used for model training directly. Visualization of MNIST dataset is shown in figure 5.

Figure 6 is a simulation test of distributed learning on MNIST dataset with ten clients system, the training data is assigned to each client under different hyper-parameters non_iid_alpha. For a global model with lower non_iid_alpha, the training loss decreasing slower than that for a model with larger non_iid_alpha, and the test accuracy remains in a low status after rounds of training. This can be certification that FedAvg algorithm works well in the iid dataset, but can hardly get convergence in the non_iid dataset.

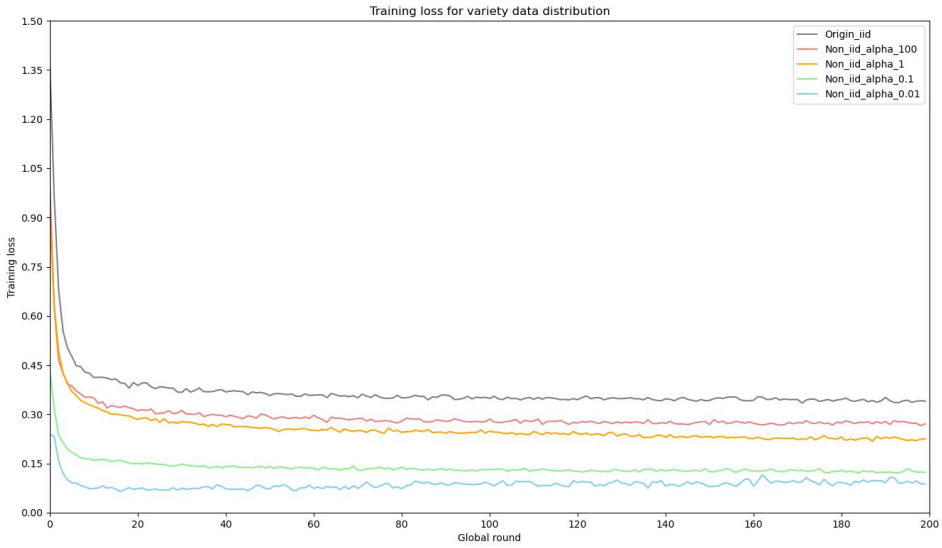


Figure 6(a). Training loss on MNIST dataset under Dirichlet distribution

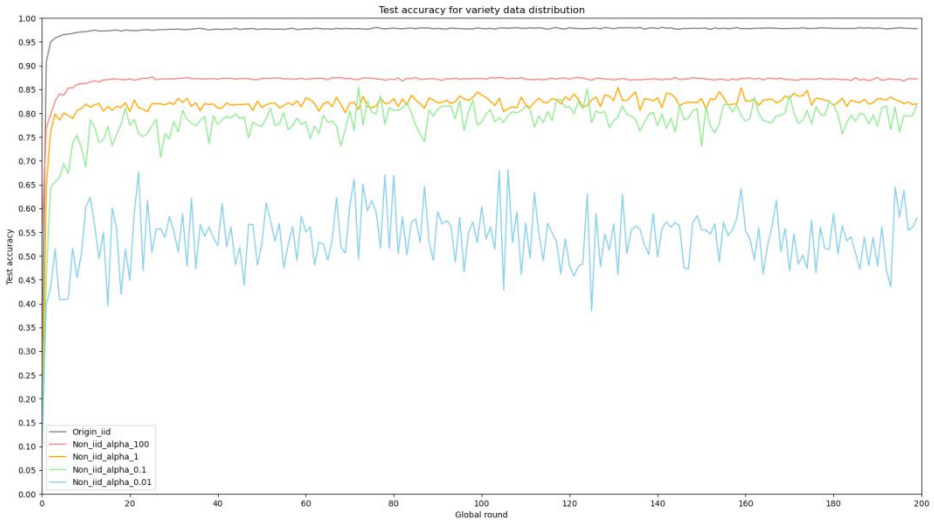


Figure 6(b). Test accuracy on MNIST dataset under Dirichlet distribution

In addition to the impact of non_iid degree on federated learning model training, other hyper-parameters will also show significant influence on the accuracy of the global model, such as local training epoch and local batch size. Some recent works have focused on the client model's parameters optimization, while others tries to train a independent model for each clients, which is known as personalized federated learning algorithm. However since we are focusing on improving the performance of federated learning under the assumption of non_iid dataset, using distributed GAN for data augmentation, the comparative experiments on other hyper-parameters will not be the core content.

2.3.2 Performance Evaluation on Cifar10 Dataset

Cifar10 is another training dataset which consist of 60,000 three-channel colorful images with the size 32*32, a little bit larger than MNIST. The dataset is classified into 50,000 training data and 10,000 test data, and the training data can be divided into ten categories with 6,000 image in a single category, such as airplanes, cars, trucks, etc. Visualization of Cifar10 dataset is shown in figure 7.

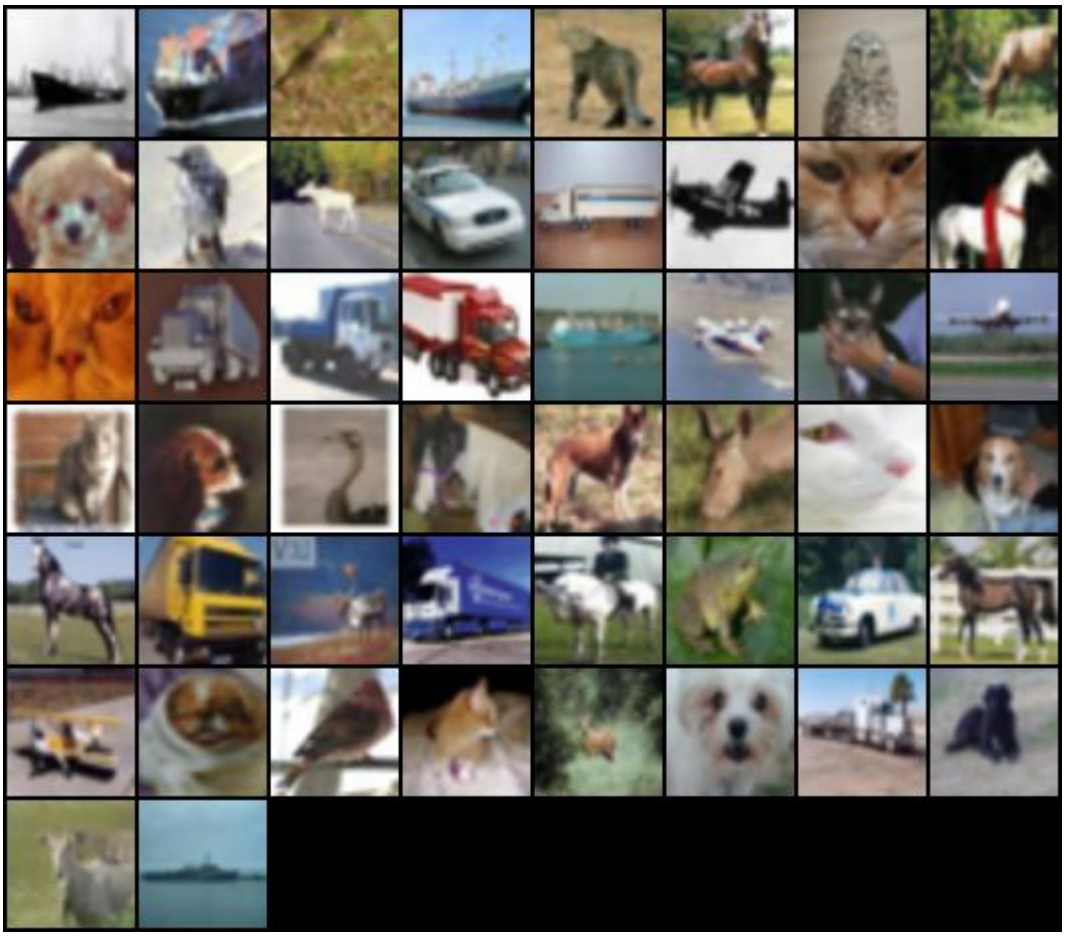


Figure 7. Cifar10 dataset training sample

The training of federated learning on Cifar10 would be more difficult to achieve convergence: firstly, the effective information of a single image in Cifar10 dataset is obviously much larger than that in MNIST dataset; secondly, the image in Cifar10 has three input channel to represent different colors, which is also more complicated than MNIST. Although the images in Cifar10 has been selected, it is not based on manual annotation. So we increase the global rounds of federated learning training process to ensure the convergence of the model as much as possible.

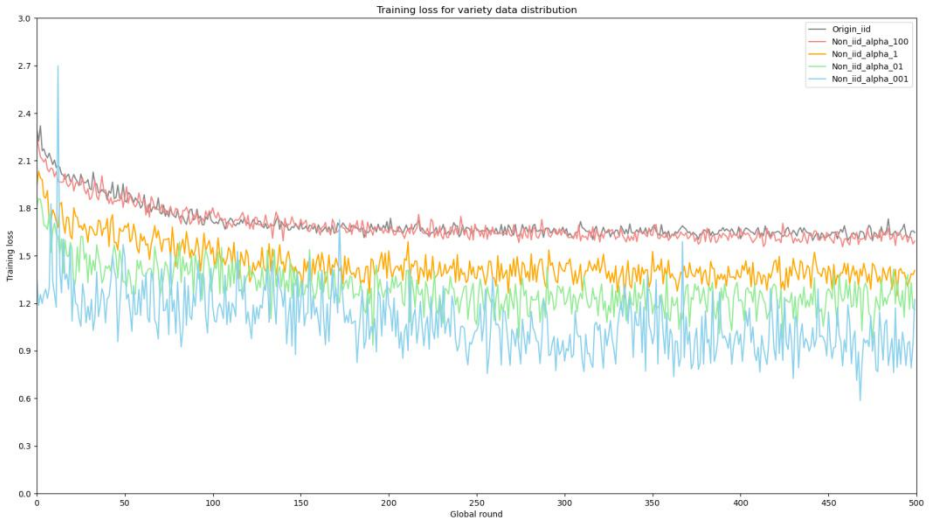


Figure 8(a). Training loss on Cifar10 dataset under Dirichlet distribution

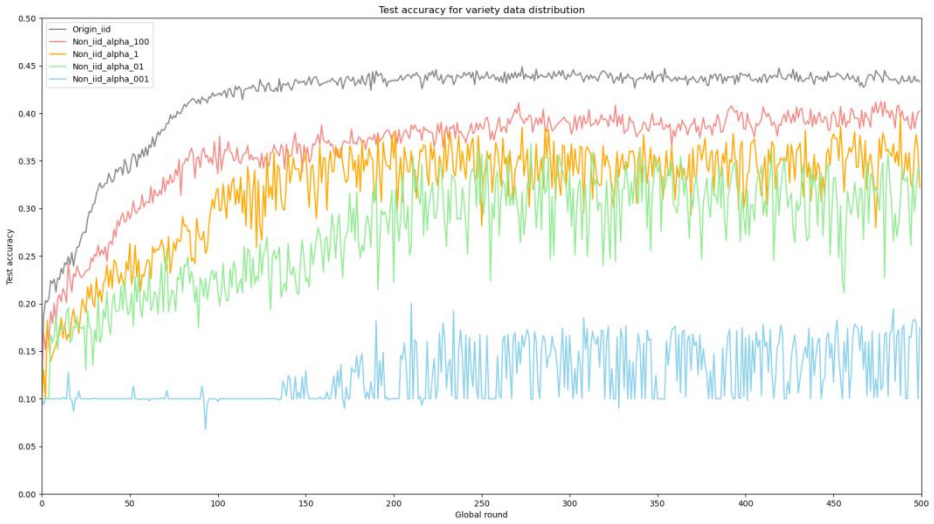


Figure 8(b). Test accuracy on Cifar10 dataset under Dirichlet distribution

The global model on Cifar10 reach convergence after 250 epochs training, while the training loss curve which represent the training on non_iid data remains fluttering in Figure 8(a). Considering the hyper-parameter fraction = 0.1, which means only 10% of the clients participated in the federated learning algorithm on each epoch, the degree of non_iid

distribution has a larger impact on the global model and for lower non_iid_alpha, the curve would be remain jittering. The negatively correlated relationship between model performance on test accuracy and non_iid_alpha can be seen from Figure 8(b), the stronger the degree of non_iid distribution, the lower accuracy of the model will be trained. This is a direct expression of the influence of non_iid distribution data on the performance of the federated learning model.

Chapter 3 GAN based Data Augmentation

3.1 Data Augmentation in Federate Learning

3.1.1 Model based Methodology on Migrating Non_iid Clients

The impact of non_iid data distribution on the accuracy of the federated learning model is majorly reflected in the weight divergence, which is raised by multiple rounds model training on the independent local dataset, and the system synchronization between clients and server under the premise of uneven data distribution[6]. Some researchers try to find a solution from a model based method, such as Model Parameter Regularization, which has been a common strategy to avoid overfitting in machine learning sector. The application of regularization algorithm in federated learning can achieve the stability of convergence and improve model generalization. Representative work as: FedProx[9], FedCurv[51] and FedCL[52]. Model regularization can be a important methodology in solving the problem like non_iid data caused training in-convergence, and it can also prevent catastrophic forgetting under the assumption of strong non_iid distribution.

Another model based approach is Clustering Algorithm, which is a basic unsupervised learning algorithm focusing on classify unlabeled training data automatically, using predefined distance equations to calculate the distance between each nodes. In federated learning, each client with independent training dataset can be considered as a node, and the distance between each node can be calculated by the predefined distance formula. After distance calculation and node aggregation, the randomized and uneven distribution of training data in a system can be transfer into several clients cluster, and the data distribution between each client which has been aggregated in a single cluster are biased toward iid states. Highlight works corresponding to clustering algorithm in federated learning as CFL[53], which use cosine function as the distance equation for clustering. FeSEM[54], which computes multiple global models using federated learning algorithm and then use Expectation Maximization algorithm to achieve the best match between each client and each cluster center.

3.1.2 Data based Methodology on Migrating Non_iid Clients

Although model based approach compensate the influence of non_iid distribution to the global model in a algorithmically way, it cannot guarantee to eliminate the negative effects, especially under the restriction of data privacy. Another methodology is to focus on the data itself and tries to enhance local training data using data augmentation method. Traditional data augmentation algorithms, such as SMOTE[55] and ADASYN[56], using over sampling algorithm can reduce the adverse effects of non_iid distribution theoretically.

To extent the method of data augmentation to federated learning, two critical issues need to be solved simultaneously: distributed machine learning structure and client's data privacy. Zhao[18] proposed the increase of earth mover's distance between clients caused by weight divergence in non_iid data distribution, which will make the model in-convergent eventually, and this phenomenon can be covered by a shared global dataset, so as to reduce weight divergence during federated model training and increase model accuracy. FAug algorithm[40] takes another approach. Following the inspiration of Knowledge Distillation(KD), FAug upload part of client's private data to the server side, and use the uploaded data as training data to train the discriminator of a GAN deployed on server. After GAN training is completed, the generator of GAN is allocated to each client, and the data generated by the generator is used for data augmentation, thereby improving the accuracy of the global model. This algorithm using GAN to generate fake data for data augmentation can objectively generate additional data and expand the client's local dataset, and the client's local dataset will show a more balanced data distribution theoretically.

However the disadvantage of this algorithm is that, although data encryption technologies, such as differential privacy can enhance the privacy of uploaded data, uploading the original data from client's local database to server still has the risk of leaking. As long as the client do process source data uploading, the privacy concern of federated learning will always be an unavoidable problem.

3.2 GAN Training on Distributed Framework

3.2.1 Multi_path Generator GAN on Synthesize Dataset

In order to prevent the privacy concern caused by data exchange, we consider using client data locally to train the discriminator of a GAN, since there will be no data permutation between clients and server. In the distributed federated learning framework with multiple clients, a multi_discriminator GAN will be trained. Under the assumption of non_iid data distribution, the label distribution between each client shows a significant difference, which means that for traditional single generator GAN cannot cover the multimodal distribution of the training data accurately. So we implement multi_path generator instead of simple generator to train the GAN which is deployed on server side.

Two obvious advantages for global data augmentation of federated learning can be obtained: firstly, the guarantee on privacy of client data. The local data of each client is strictly protected in the local database, and the local data does not participate in any data transmission. Secondly, the multi_path generator fits the data distribution of non_iid federated learning, where local data shows multi_peak distribution. It can significantly improve the efficiency and accuracy of the multi_discriminator GAN, compared with the traditional GAN model.

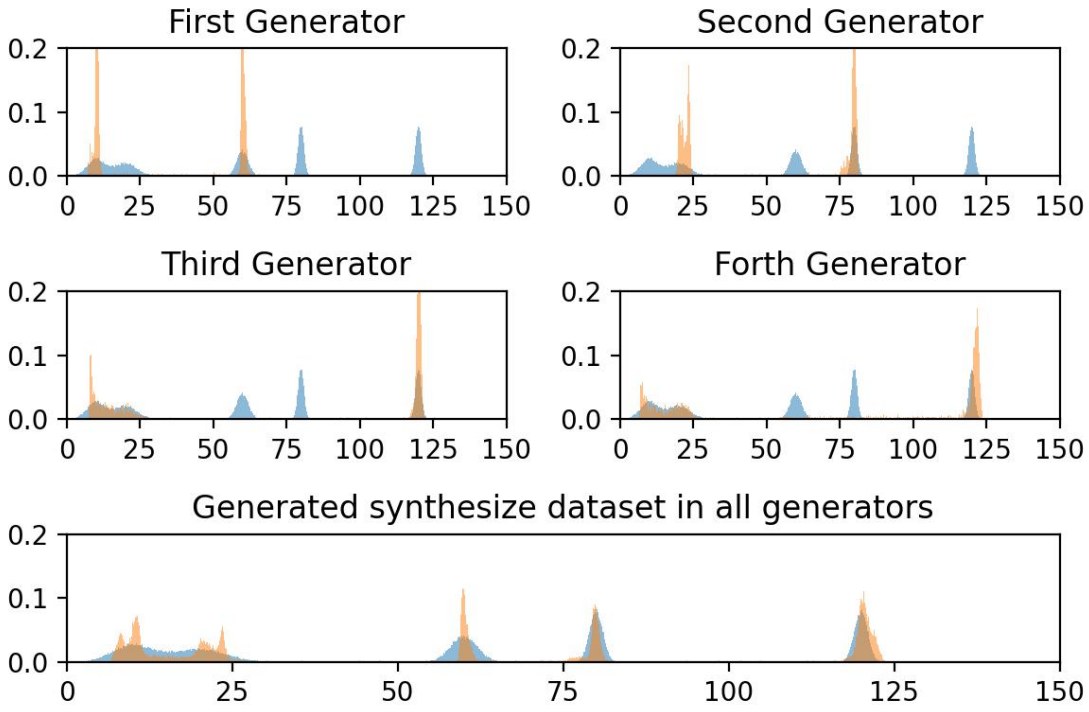


Figure 9. Multi_path GAN training on synthesize dataset

An experiment that simulates the multi_path generator GAN trained on a multimodal data source, to detect the accuracy of GAN's fitting performance on multimodal data source is shown in figure 9. The blue shadow is the training data generated by GaussianMixture function from Scikit-Learn machine learning library, and its peak data distribution is independent and diverse, which can simulate the non_iid distribution of the data in the actual application of federated learning.

For GAN training, a generator with four paths and a unified discriminator is trained in proper order. The loss function of generator is defined as the summarize of loss value of four paths, and the loss function of discriminator is the same as traditional GAN. In practical situation, the discriminators will be assigned to each client, and use client's local data to update a single discriminator. As a result, the loss function for discriminator should be set to be the sum of loss value of each distributed discriminator, and then do back propagation on the loss value of both discriminator and generator to update respective optimizers. Theoretically, the number of discriminator does not have any influence to the

model fitting on non_iid data distribution, so the only parameter that need to be discussed is the number of paths in generator.

The sub-graph above in figure 9, shows the output of each path in multi_path generator after 200 epoch GAN training, and the number of paths is set to be four. Although a single generator can generate multimodal data, considering the simplicity of structure of a single path generator model, it cannot cover the data distribution of a complex input data completely, and this is a problem that cannot be solved by increasing the training epoch. The sub-graph below shows the comparison of the output of multiple paths generator and the distribution of input data, with a higher coincidence degree compares to the output of a single path. Figure 9 shows the superior fitting performance of multi_path generator, and this phenomenon can also be found in non_iid training dataset.

3.2.2 Multi_path Generator GAN on iid MNIST Dataset

In order to simulate the data distribution in the federated learning environment, the Dirichlet distribution function is used to classify the training dataset. For the simulation of iid data, the hyper-parameter non_iid_alpha is set to be 100, and for the simulation of non_iid data, the hyper-parameter non_iid_alpha is set to be 0.1. Other parameters in the experiment are kept as consistent as possible with FedAvg algorithm. The total number of clients participating in the federated learning is set to be 100, and the hyper-parameter fraction = 0.1, which means that there are 10 random selected clients are participating in the model training for each round of federated learning training process.

The model of Multi_generator GAN adopts the DCGAN structure: Discriminator is a five layers Convolutional Neural Network with kernel size = (4, 4), stride = (2, 2) and the maximum output channel is 512 which can ensure an efficient feature extraction. Each kernel of the network is followed by a batch normalization and non_linear activation function LeakyRelu, which shall reduce the number of parameters in neural network and avoid saturation in GAN training. Visualization of Discriminator structure is shown in figure 10(a). Generator is also a five layers convolutional neural network with an opposite feature extraction order compares to Discriminator model, and the kernel size as well as non_linear function of each layer is consistent as well. More details is shown in figure

10(b).

It should be emphasized that the Generator model uses the pytorch.nn.ModuleList() function. According to the predefined hyper-parameter Generator_paths, each independent neural network is trained by the input data stream in turn, and the training result of all paths are set to update the model using back propagation after evaluated by loss function. The difference between Multi_generator GAN and traditional GAN model is the generated data through a multiple path model can cover a differentiated data distribution, which is often shown in non_iid data distribution in federated learning algorithm, and to avoid model collapse during training.

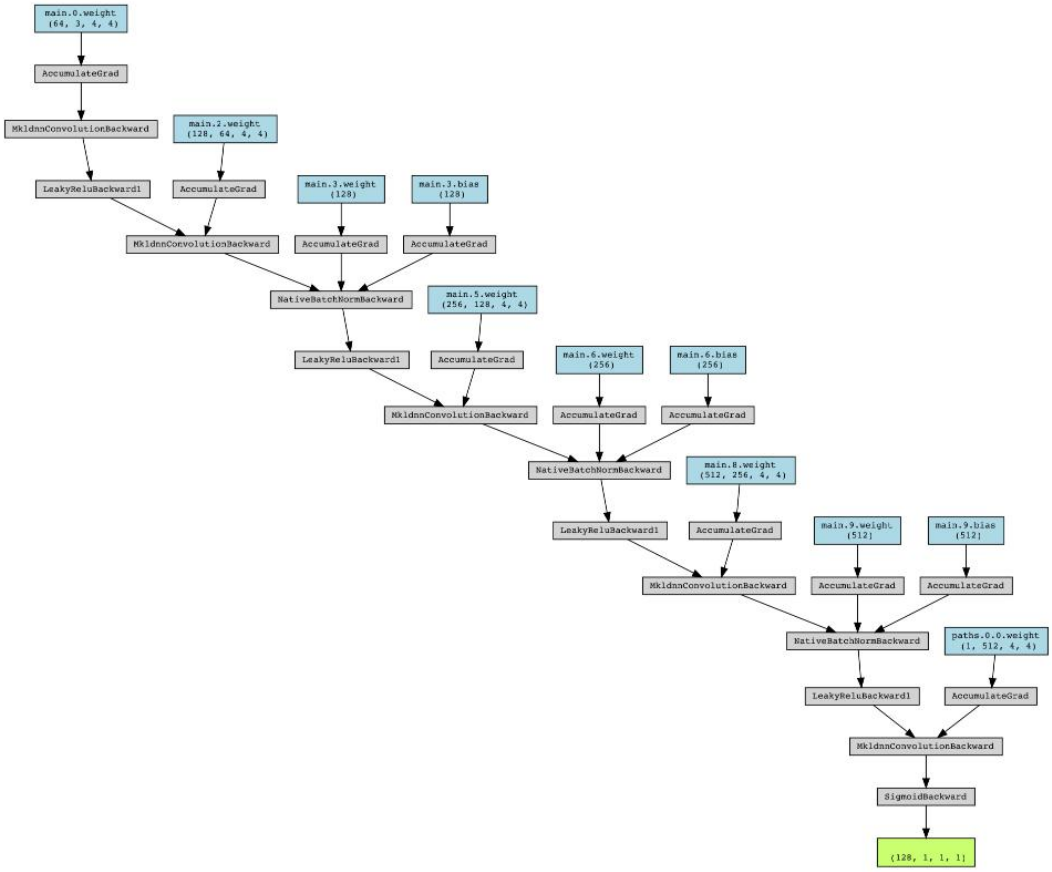


Figure 10(a). Visualization of Discriminator model

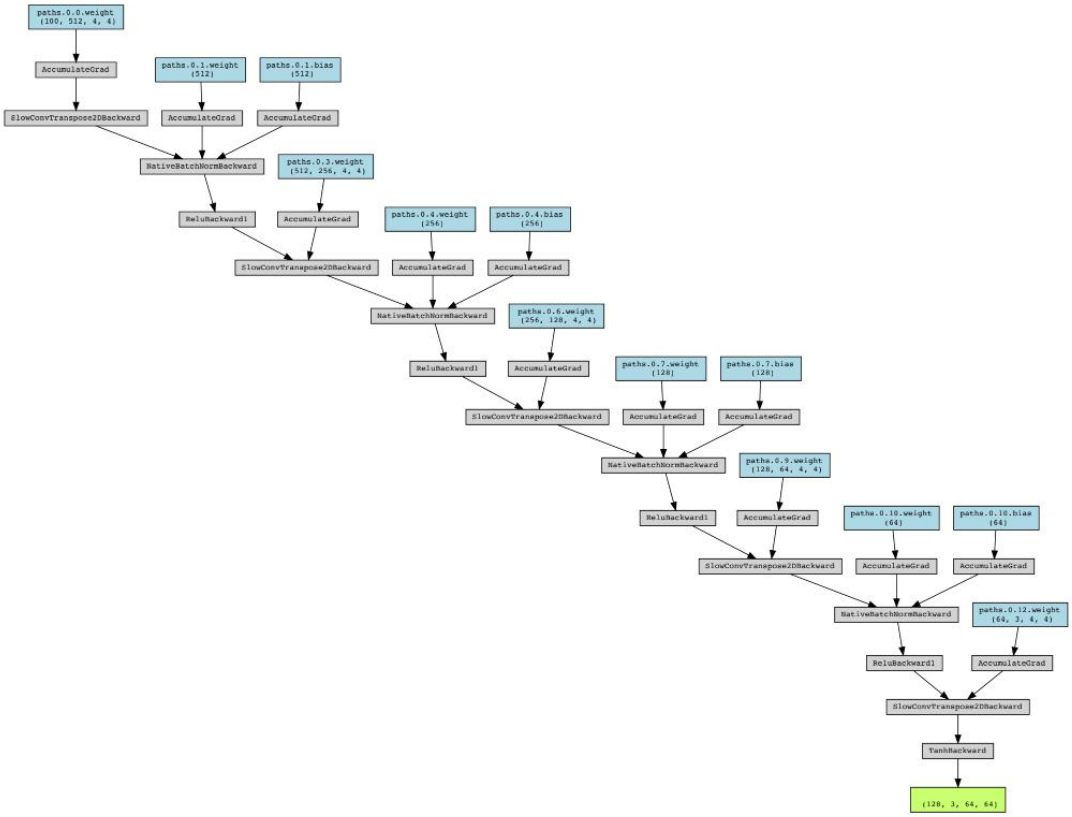


Figure 10(b). Visualization of a single path in Multi_path generator

Multi_path generator GAN is trained under the distributed machine learning framework, sharing the consistent hyper-parameter setting with federated learning algorithm as much as possible. Set the number of discriminator participating in overall GAN training to be 100, and the number of discriminator participating in each epoch of GAN training to be 10, which is also consistent with FedAvg algorithm where hyper-parameter fraction = 0.1. Each discriminator only pick the corresponding training data to update model, which can ensure that the training data assigned to each client in federated learning algorithm is also the data allocated to the discriminator for GAN training. Thereby we can guarantee the integrity and privacy of the source data.

In each round of GAN training, only 10 discriminator participate in training and updating parameters, while each path of the multi_path generator is trained in turn, which is an ideal training process for the generator.

Figure 11(a). Real sample for discriminator, non_iid_alpha = 100

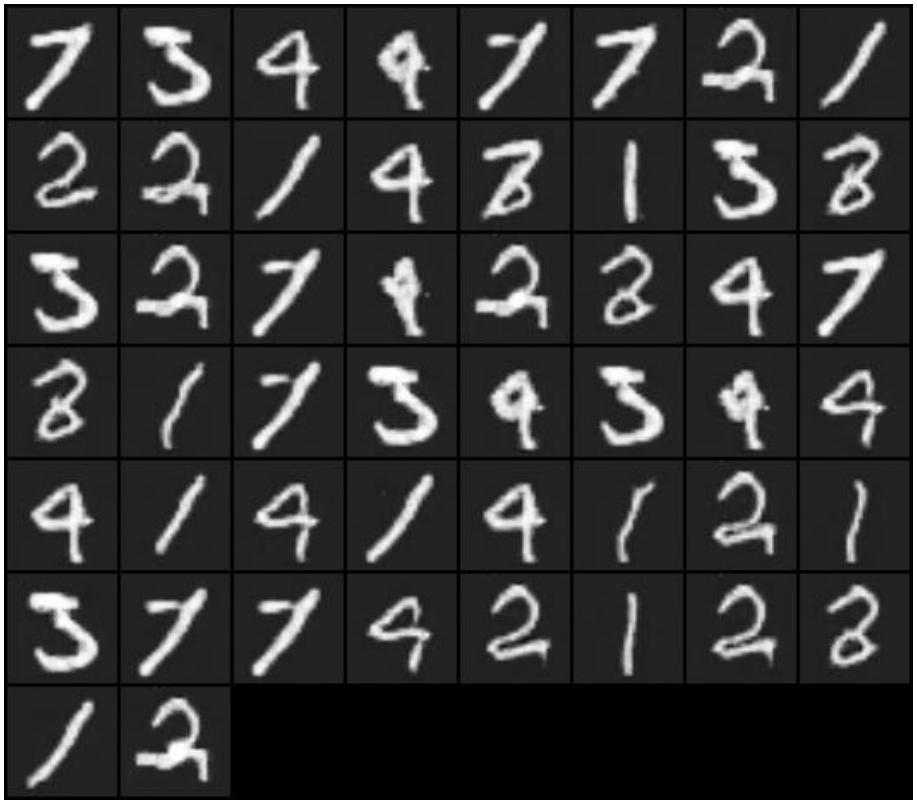


Figure 11(b). Generated sample form single_path generator, non_iid_alpha = 100

Figure 11 shows the output visualization image of the single_path generator GAN trained in the distributed framework. Figure 11(a) is the visualization of original training data partitioned by Dirichlet distribution function, and it is also can be considered as the input data for a random discriminator. The label distribution of the partitioned data is relatively average which conform to the characteristics of the iid data distribution. Figure 11(b) is the visualization of the generated image from generator after 50 epoch GAN training, using Gaussian Noise as input data. Output image can be identified manually as valid images, showing a steady generator status after epochs training. The output visualization of single_generator GAN trained in a uniformly distributed framework can be the proof that single_generator GAN can finished the simple training task as expected.

3.2.3 Multi_path Generator GAN on Non_iid MNIST Dataset

For the simulation of non_iid distribution of training data, we set the hyper-parameter non_iid_alpha to be 0.1 in the data partition process. The visualization of partitioned training data is shown in Figure 12(a), also it shall be the input data assigned to a specified discriminator as training data. The input data under the condition non_iid_alpha = 0.1 is not evenly distributed. As a result, a single discriminator has only a few categories of training data, as the same categories participating into the federated learning algorithm. Since data exchange between clients is strictly prohibited, data distribution between clients will show a multimodel mode, which has been discussed above.

Compare with iid data, the training data partitioned under non_iid distribution is unlikely to train an ideally GAN model in single_path generator training process. Figure 12(b) shows a typical example of training failure. Although manually recognized images can be generated successfully, the image type is simple and cannot cover multimodel mode training data. Figure 12(c) shows another example of failure. In this case, single_path generator cannot generate images that can be recognized as Arabic numerals by human being. GAN model presents a model collapse statement, and this kind of failure can not be carried out by increasing training epoch.

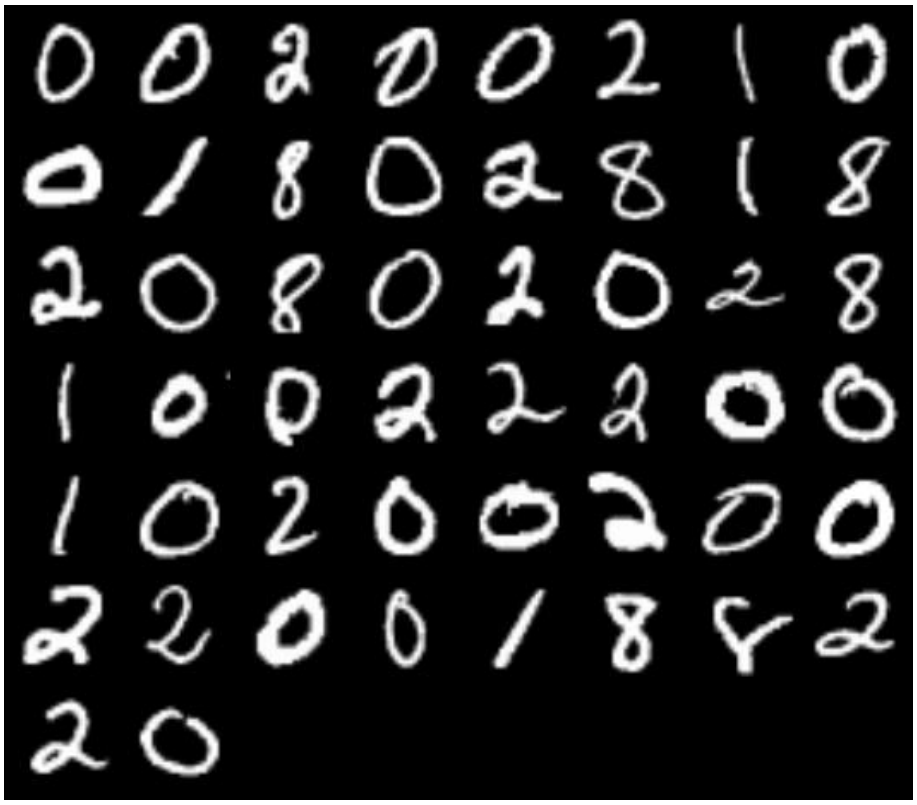


Figure 12(a). Real sample for discriminator, non_iid_alpha = 0.1

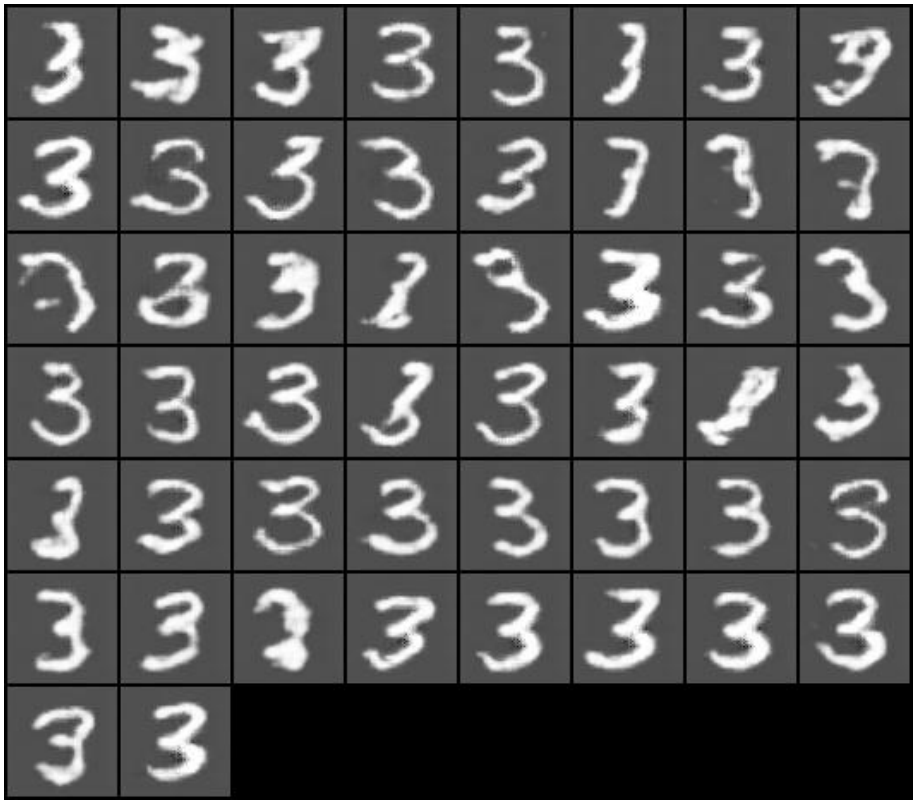


Figure 12(b). Generated sample from single_path generator, non_iid_alpha = 0.1

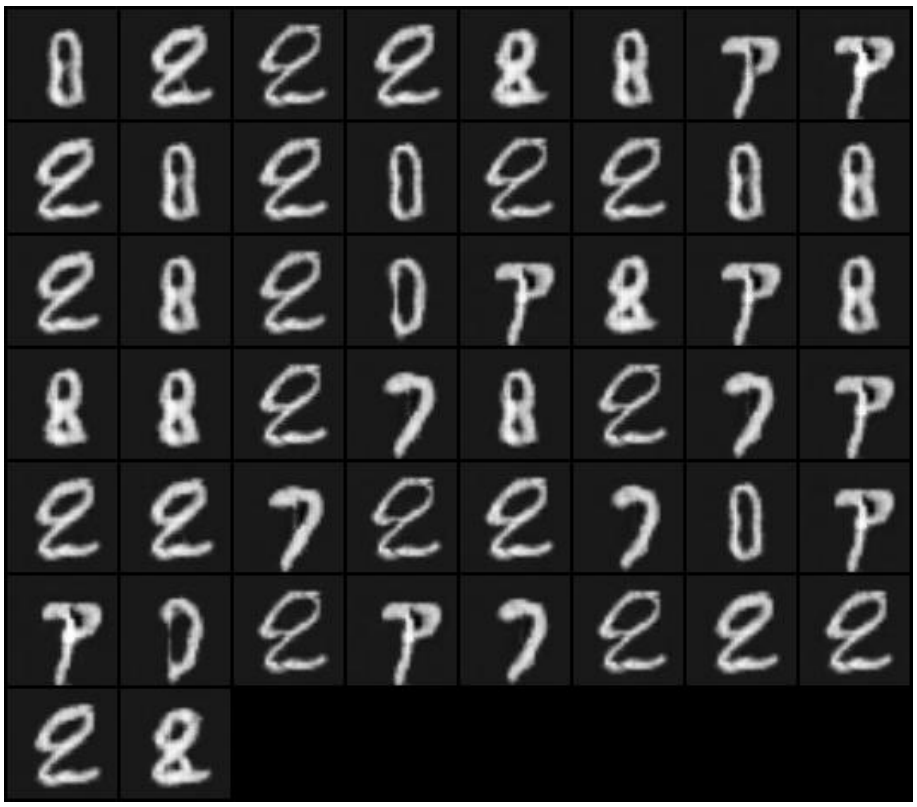


Figure 12(c). Generated sample from single_path generator, non_iid_alpha = 0.1

The optimized GAN model use a multi_path generator for training, and the hyper-parameter Generator_paths is set to be four. Figure 13 shows the visualization of generated images from the generator on four paths in proper order.

Although there are still some images that cannot be manually recognized as Arabic numerals, most of the generated image can be considered as valid outputs. The output of four paths in a generator basically covers all the labels of the original training data in MNIST. The output image in Figure 13 indicate that the multi_path generator has more advantages than the single_path generator in GAN training for the non_iid database, and the former structure meet the requirements of a model used in federated learning data augmentation: client's data is protected in the local database and no data exchange between clients, which can ensure privacy concern, and the data generated by the multi_path generator covers the non_iid data distribution of the input training data.

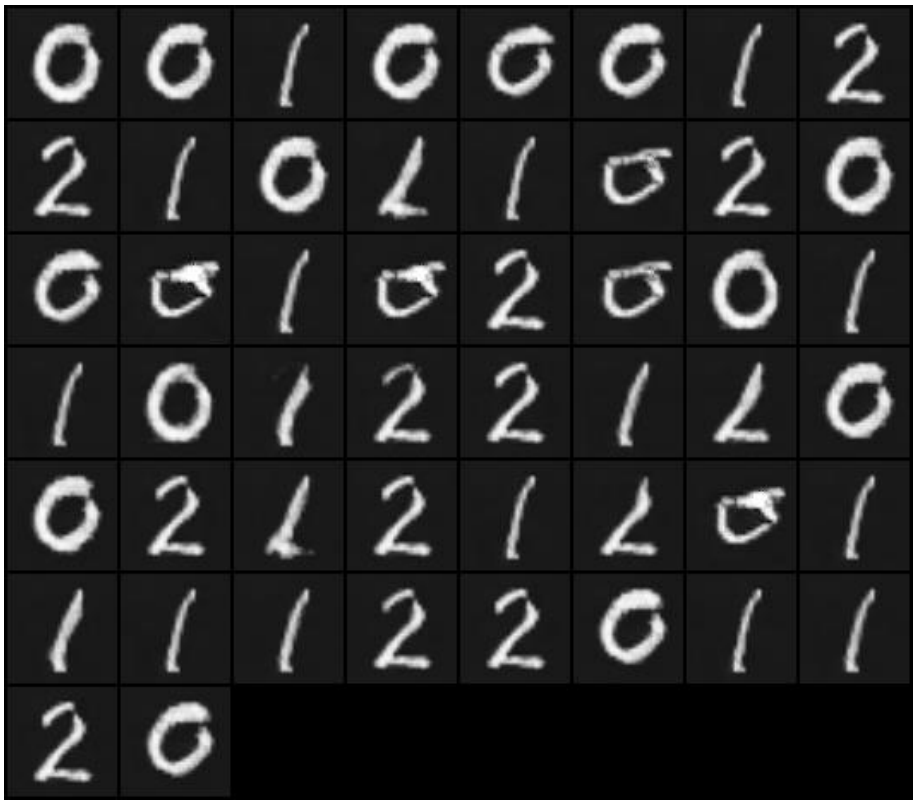


Figure 13(a). Generated sample from multi_path generator, num_path = 1

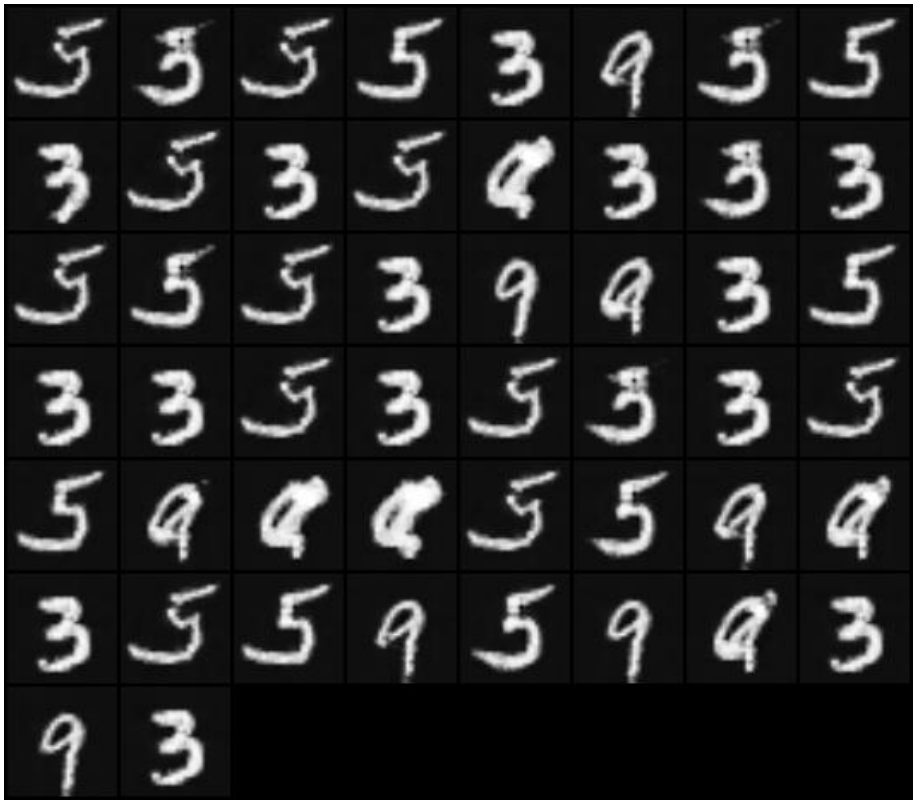


Figure 13(b). Generated sample from multi_path generator, num_path = 2

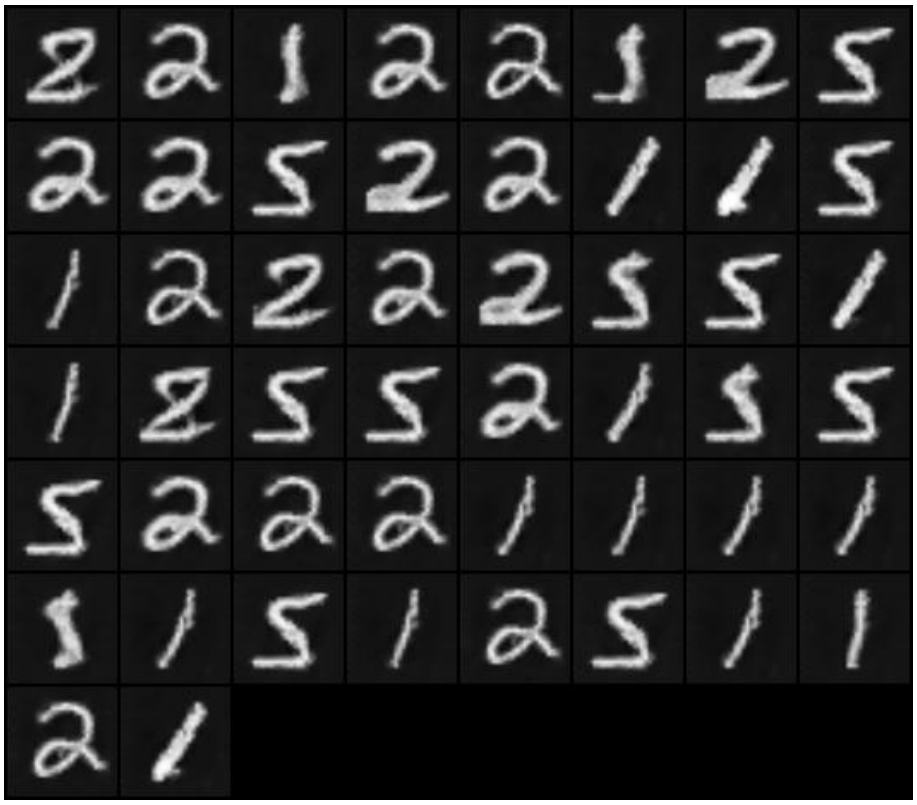


Figure 13(c). Generated sample from multi_path generator, num_path = 3

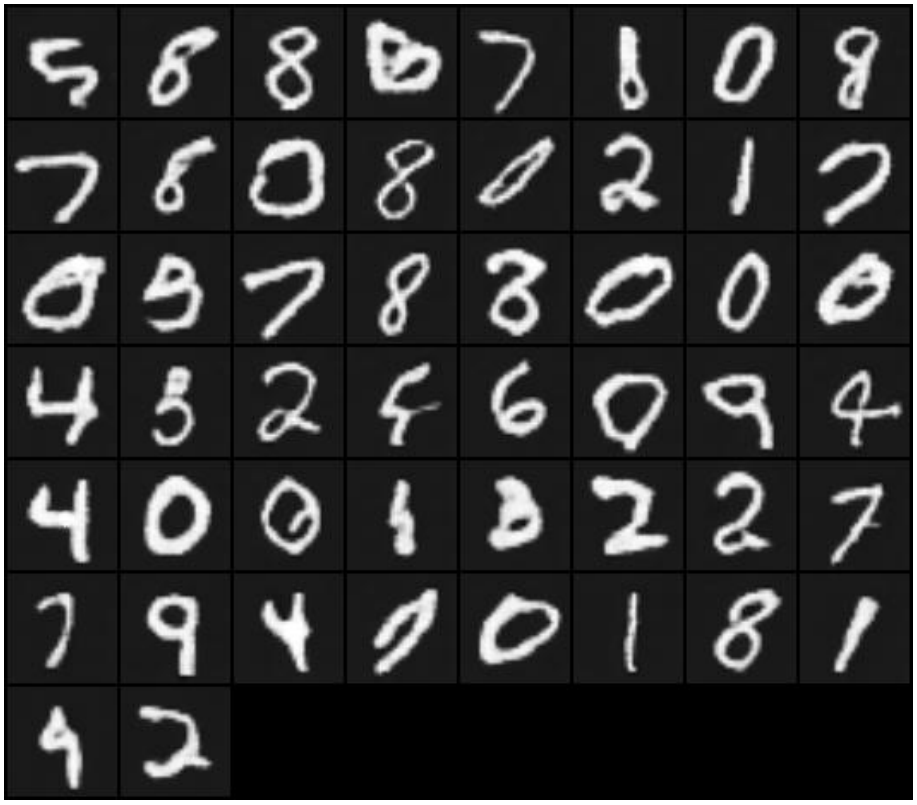


Figure 13(d). Generated sample from multi_path generator, num_path = 4

3.2.4 Multi_path Generator GAN on Non_iid Cifar 10 Dataset

An intuitively analyze of training result of the multi_path generator on multimodal training data is discussed above. The major reason for using MNIST dataset for multi_path generator GAN performance analysis under the condition of non_iid data distribution, is that the Arabic numerals image is highly recognizable than colorful images, and it is much easy to observe the difference of generated images between single_path and multi_path generator GAN, with different hyper-parameter non_iid_alpha.

In fact, the multi_path generator GAN can also be trained on other databases, such as Cifar10 dataset.

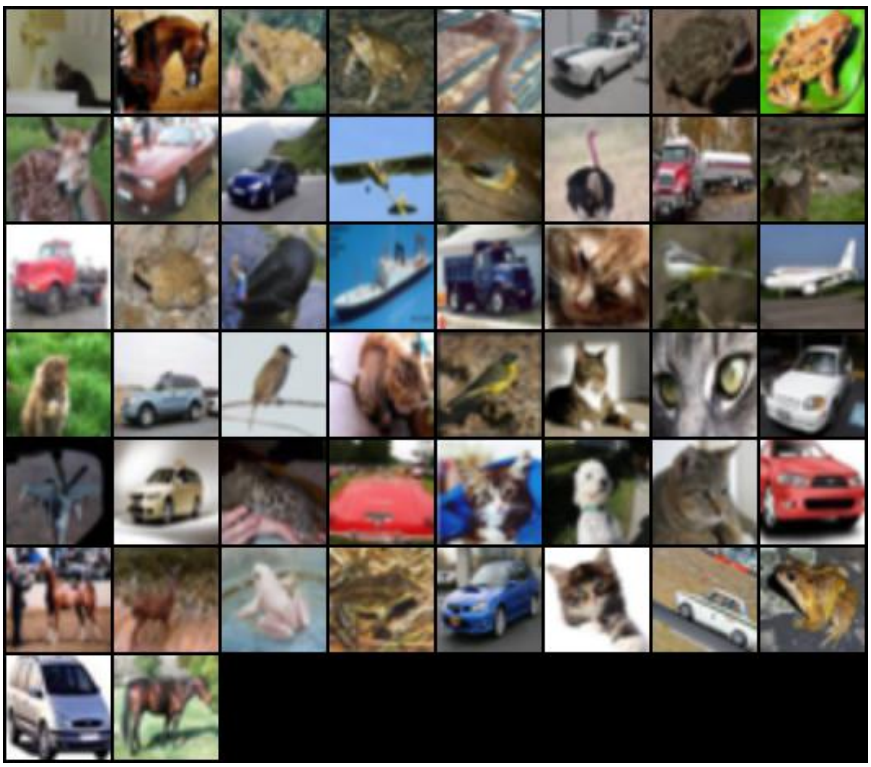


Figure 14(a). Real sample for discriminator, non_iid_alpha = 100

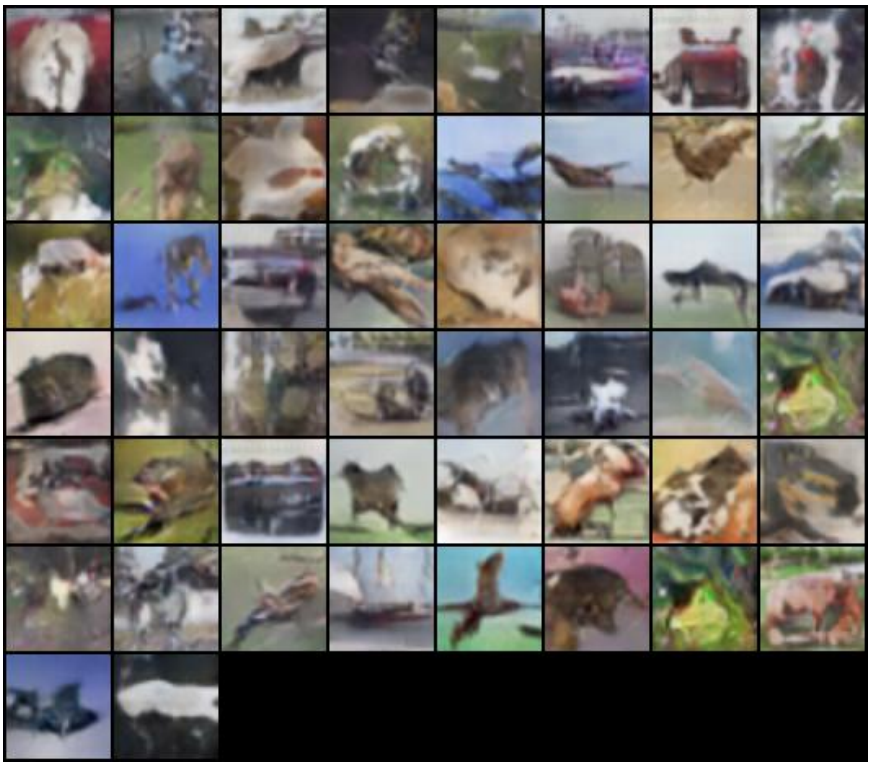


Figure 14(b). Generated sample from single_path generator, non_iid_alpha = 100

Figure 14(a) is the input training data of a randomly chosen discriminator, under the consideration of iid data distribution, which means the hyper-parameter non_iid_alpha in the Dirichlet distribution function is set to be 100. Figure 14(b) is the data generated by the well trained single_path generator, and the generated image shows a highly degree of recognition which means GAN is trained successfully as expected.

Figure 15(a) shows the input data of a discriminator randomly selected in non_iid data condition. Despite of the low clarity of the image in Cifar10 dataset, it can still be figured out that the only corresponding labels are Horse, Car and Ship, which belong to the data partition result of non_iid_alpha = 0.1. Figure 15(b-e) is the visualization of generated images from well trained multi_path generator GAN. The generated image shows a high degree of discrimination and basically covers all labels of the Cifar10 dataset.

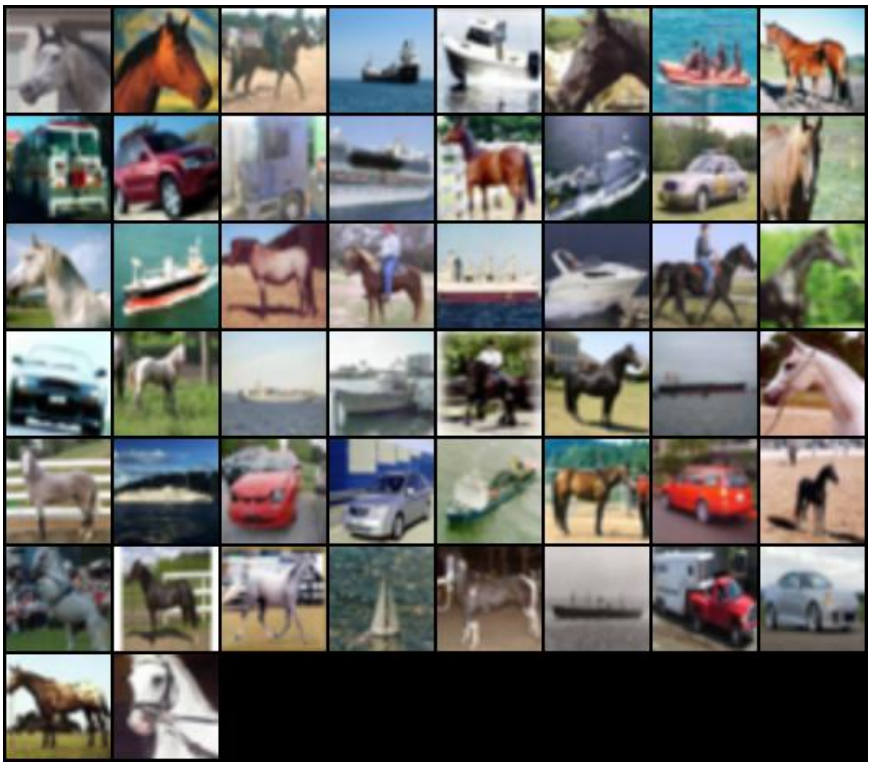


Figure 15(a). Real sample for discriminator, non_iid_alpha = 0.1

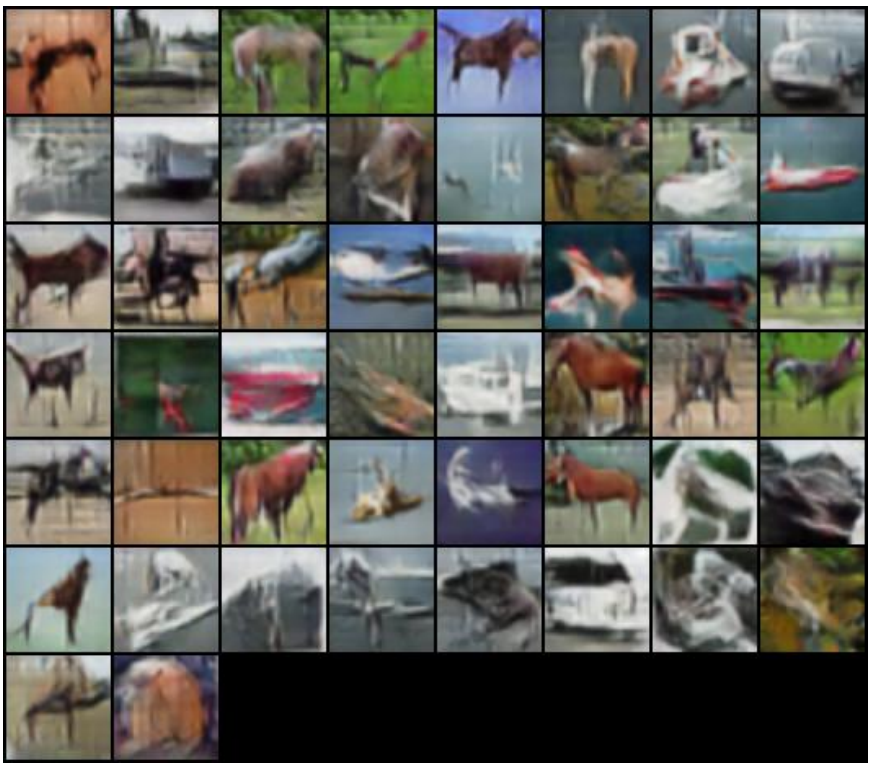


Figure 15(b). Generated sample from multi_path generator, num_path = 1

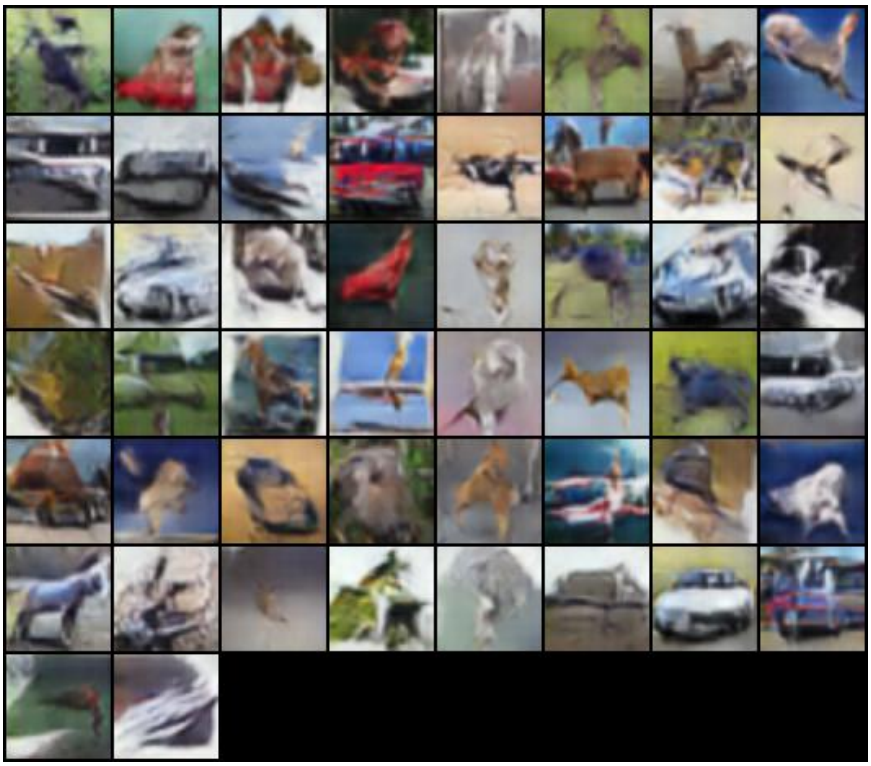


Figure 15(c). Generated sample from multi_path generator, num_path = 2

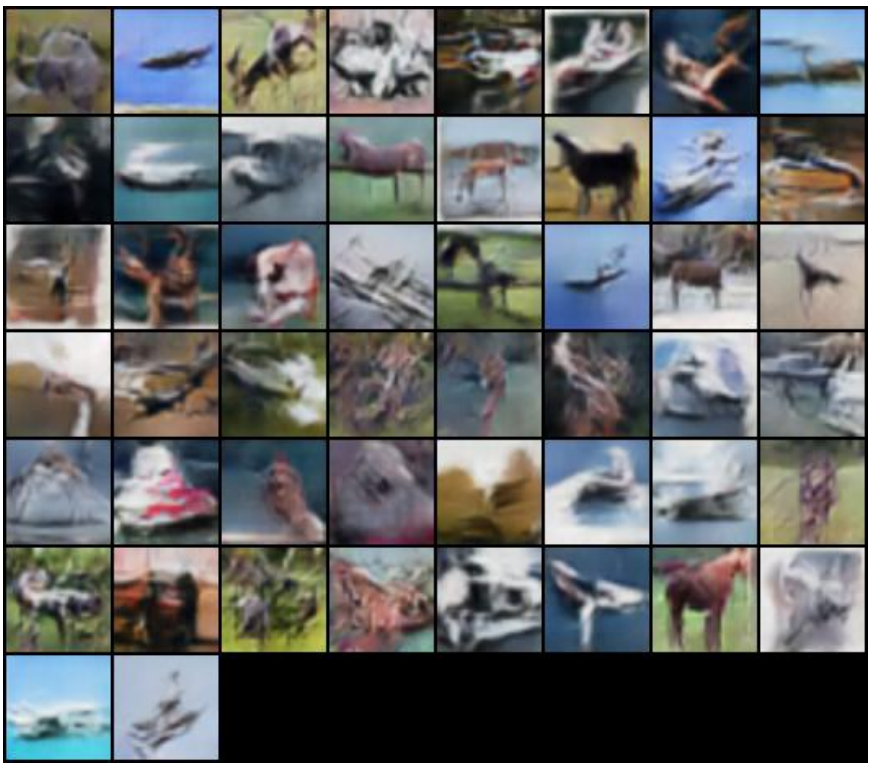


Figure 15(d). Generated sample from multi_path generator, num_path = 3

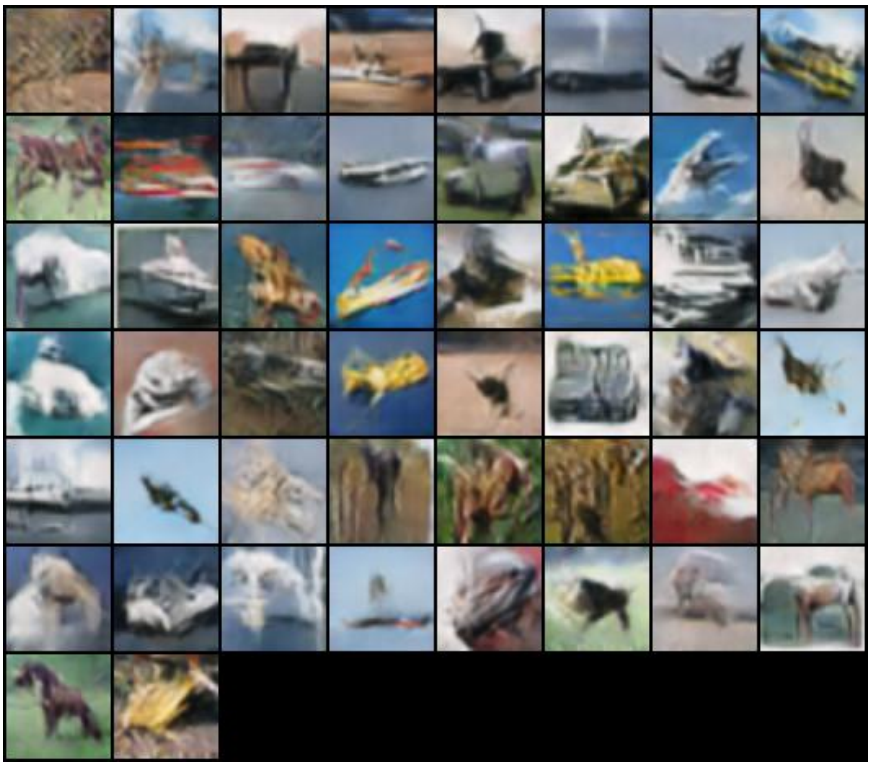


Figure 15(e). Generated sample from multi_path generator, num_path = 4

3.2.5 Training Loss Analysis in Multi_path Generator GAN

The ideal statement when processing GAN training is to achieve a balance between generator and discriminator, where the discriminator cannot distinguish between real data and fake data generated by the generator, since the generator has been well trained to generate fake images. Figure 16 shows the loss value calculated by the Binary Cross Entropy(BCE) loss function. In each training epoch, 10 discriminators are selected and trained randomly, and then calculate loss value for each discriminator for back propagation, which means the 100 scale value on the abscissa corresponding to 10 epochs in GAN training process.

Since the training process is triggered, the rapidly drop of the loss value from discriminator indicating the gradually converge of discriminator model. As the training progress carries on, the loss value approach 0 after around 20 epochs training. While the loss value of discriminator remain at a low level approximately, occasional jump occurs

several times during training process, which is exactly the phenomenon can be visualized when data distribution of the input data change temporarily. Strong non_iid distribution has a certain negative impact on the distributed GAN training, and the multi_path generator can quickly adapt itself to the sudden changes from data distribution of input data, which is the reason loss value of discriminator drop back to 0 after a few following epochs.

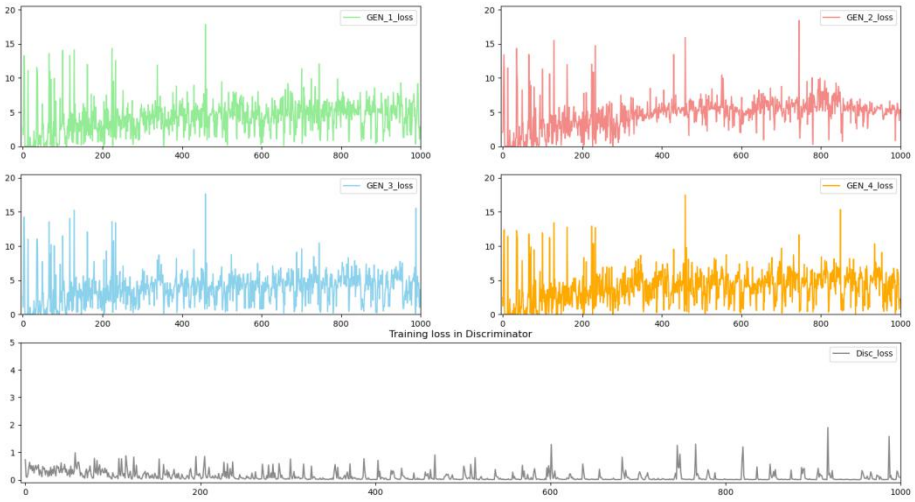


Figure 16(a). Loss value for GAN trained on MNIST dataset

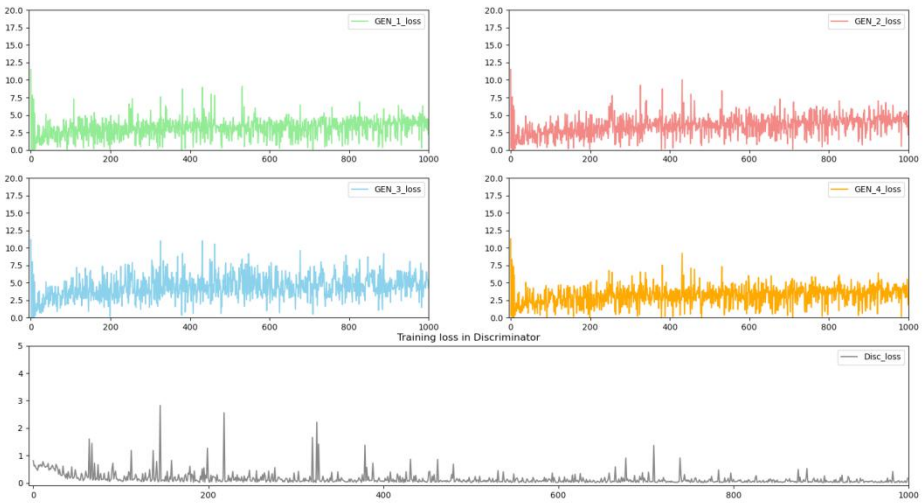


Figure 16(b). Loss value for GAN trained on CIFAR10 dataset

The back propagation pass the sum of four paths' loss values in one training epoch, there is no significant differences between each paths in one generator. With the process of GAN training, the loss value gradually reach stable level as well: it is normal for the generator loss value to oscillate slightly near a limit average value, and it is also an ideal situation for GAN training. The generator model no longer requires large-scale parameter updates to meet the need on cheating discriminator, and the data distribution of generated data has already reached the multimodal distribution state.

Chapter 4 Federated Learning using Data Augmentation

4.1 Data Augmentation using Generated Image

Since we concentrate on the performance evaluation of federated learning algorithm on non_iid dataset, the generated data from multi_path generator GAN will be assigned to each client's dataset and participate in local training. The assignment of generated images shall be directly and randomly, and from the perspective of clients, they should not be able to find out or inference raw training data from any other clients.

Considering the strict constrain of privacy in federated learning algorithm, where raw data exchange is prohibited between clients, data augmentation in this work using distributed discriminators GAN trained on the basis of local dataset for each client. An overall multi_path generator should be ready for data generation after 30-50 epochs training until the generated images can be manually recognized as well as the data distribution of generated data covers the non_iid distributed dataset as expected.

Figure 17 shows a visualization of training dataset for a random chosen specific client in one local training batch, the images which have been fuzzy processed are the generated images from multi_path generator, while the others are part of raw training data in MNIST and Cifar10 dataset. The training dataset has been partitioned by Dirichlet distribution function, and non_iid_alpha is set to be 0.1 indicating a strong non_iid distribution. With the augmentation function processed with generated images, the independent database of this client has more categories of labels of training data to participate in federated learning algorithm.

Figure 17(a). Augmented dataset using multi_path generator GAN on MNIST

Figure 17(b). Augmented dataset using multi_path generator GAN on Cifar10

4.2 Performance Evaluation on Augmented Dataset

The performance evaluation of federated learning algorithm using data augmentation based on multi_path generation GAN, is performed on two experiments. In order to emphasize the impact of data augmentation on test accuracy, the experiment set the only tunable parameter to be the proportion between generated image and original training data. Considering the dataset to be MNIST, where the number of images in training set to be 60,000, are assigned to 10 clients using Dirichlet distribution function after separated into training_set, valid_set and test_set, for full batch gradient descent. The number of training data for a single client is 4800, and we set hyper-parameter Augmentation_fraction(Af) to be 0.01, 0.1 and 0.5, corresponding size of the generated augmentation dataset should be 48, 480 and 2400.

Also consider the FedAvg algorithm where hyper-parameter fraction C = 0.1, indicating 10 clients out of 100 total clients are selected to participating in the local updating in each training epoch randomly. The size of training set for one single client after partition should be 480, which means that in our data augmentation algorithm, when we set Af = 0.1, the size of augmentation dataset using multi_path generator GAN should be 48. Model performance for full batch gradient descent is shown in Figure 18.

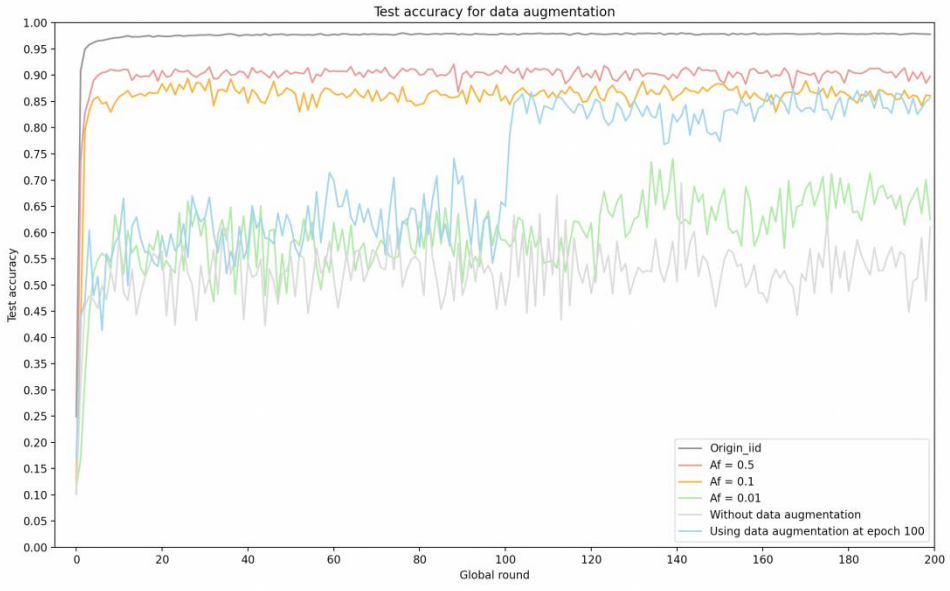


Figure 18. Performance evaluation on Augmented MNIST dataset

Figure 18 shows classification performance based on the original non_iid data, using GAN for data augmentation. With the increase of hyper-parameter Af, the overall test accuracy of the federated learning model improves significantly. For the control group partitioned by original method in FedAvg algorithm, the average test accuracy for the last 20 training epochs is 0.538, which is much lower than the augmented group Af = 0.5 with an average test accuracy 0.904. This accuracy value for augmented dataset reach the level of model whose training dataset is partitioned by Dirichlet distribution function on non_iid_alpha = 100 as shown in Chapter 2, while the test accuracy for the control group match the model performance trained on dataset which partitioned on non_iid_alpha = 0.01, indicating the significant effect on offset of model performance drop on the non_iid data distribution.

Besides, the adverse impact of non_iid distribution training data is not only shown in the test accuracy value of the model evaluation, but also performance stability. It can be seen subjectively from figure 18 that with the decrease of the degree of non_iid data distribution, the accuracy curve becomes more stable. More quantitative analysis based on test data is shown in Table 2, where the test accuracy calculates the average value of the last 20 epochs.

	Original_iid	Af = 0.5	Af = 0.1	Af = 0.01	non_iid data
Test_accuracy	0.979	0.904	0.860	0.662	0.538
variance	0.4×10^{-6}	62.2×10^{-6}	61.5×10^{-6}	0.7×10^{-3}	1.6×10^{-3}

Table 2. Model performance using GAN based data augmentation

Figure 18 also provide a dynamic method data augmentation, where Af was set to be 0.01 at the first 100 epochs and then turn to be 0.1 at the leftover 100 training epochs. A significantly gap of increasement of model accuracy appears at around 100 epoch, while the test accuracy of global model reach 0.85 at the end of training process. The phenomenon that model accuracy improved with the change of hyper-parameter of data augmentation function, can be the solid evidence that GAN based data augmentation is an efficient method to improve the performance of global model in non_iid data distributed federated learning algorithm.

Federated learning algorithm provide a heuristic method to train a global model without raw data exchange between clients and server, and this methodology works well on both full batch gradient descent and FedAvg algorithm. GAN based data augmentation using Multi_path generator to fulfill a better performance on non_iid dataset can also achieved when hyper-parameter fraction is set to be 0.1.

Figure 19 shows test accuracy of original FedAvg model using data augmentation, where the only control variable Af is the proportion of generated image for single client as discussed before. The value of accuracy curve in figure 19 indicating the average value of test accuracy around ± 25 epochs based on the abscissa content, and a significant improvement can be achieved by using data augmentation.

It should be announced that, although test accuracy shall approach the level of iid data group by increasing the size of augmentation dataset, this approach is still not recommended. The motivation of data augmentation is to cover the shortage of uneven data distribution in federated learning algorithm, not to offset the drop of model performance fundamentally. Besides, with a large size of augmentation dataset, the possibility of being attacked from an untrusted third party ascent simultaneously, which shall increase the risk on privacy concern.

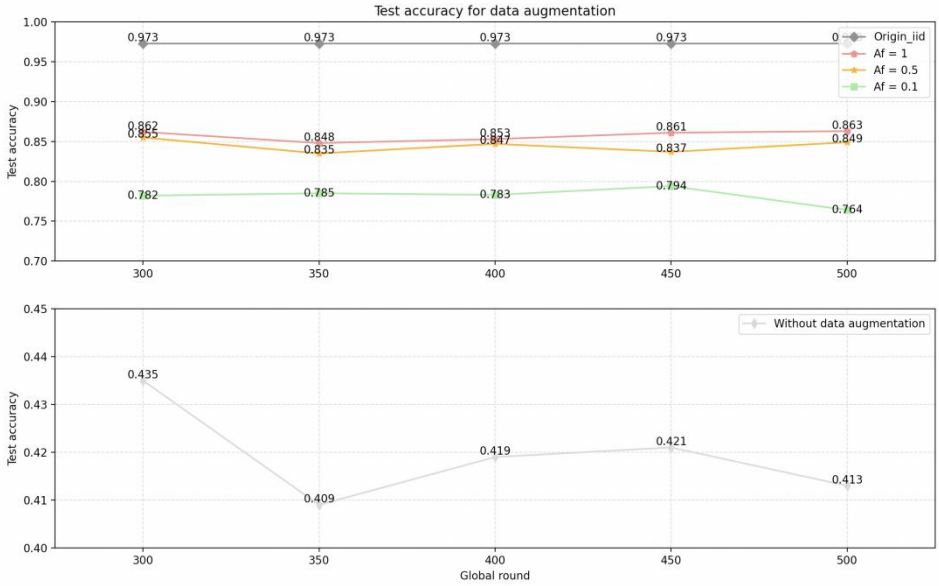


Figure 19. Performance evaluation of FedAvg on Augmented MNIST dataset

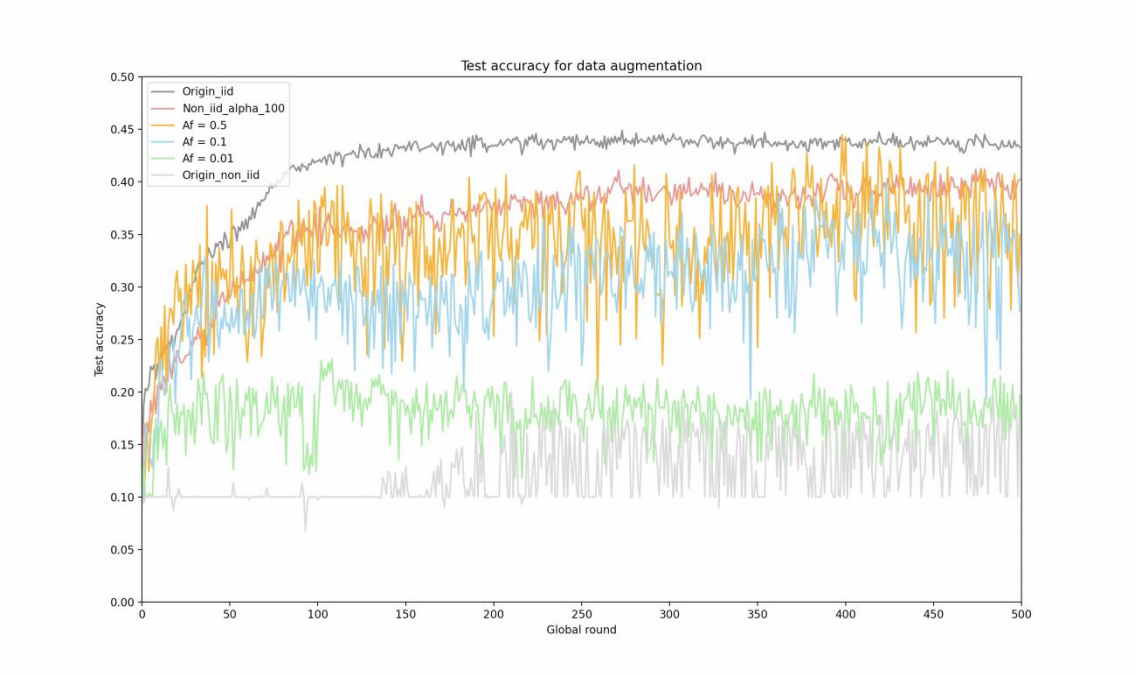


Figure 20. Performance evaluation on Augmented Cifar10 dataset

The model performance evaluation on Cifar10 dataset, where the training dataset is augmented using using multi_path generator GAN is shown in Figure 20. Classification accuracy of different models between using augmentation dataset or not can be seen obviously. The model performance on augmented dataset Af = 0.5 approach the red curve, which training data is partitioned by Dirichlet distribution function on non_iid_alpha = 100. This observation is similar to the accuracy curve on MNIST dataset shown in figure 18, indicating the improvement of model performance using GAN based data augmentation.

However, the test accuracy of model on Cifar10 dataset is not as steady as that trained on MNIST dataset. The major reason is the difficulty of training GAN model on distributed Cifar10 dataset, and the relatively poor accuracy curve of federated learning algorithm trained on non_iid data distribution Cifar10 dataset, is another possible reason. Model performance on Cifar10 dataset or even larger scale training dataset can be improved by implement a more complicate deep neuron network, or separate GAN training process from federated learning algorithm to obtain a better augmentation dataset. This can be some direction for further research, and significant accuracy improvement on non_iid data by

using multi_path generator GAN based data augmentation have been proved above.

4.3 Performance Evaluation on Local Model

Since we set as much number as parameters in our federated learning evaluation, to fit FedAvg algorithm for a intuitive comparison, the local training epoch for client's model update in one global round is also set to be five. Figure 21 is a visualization of the local training loss of one specific client at different global rounds, and from which we can illustrate how global model work on a single client model on both iid and non_iid dataset.

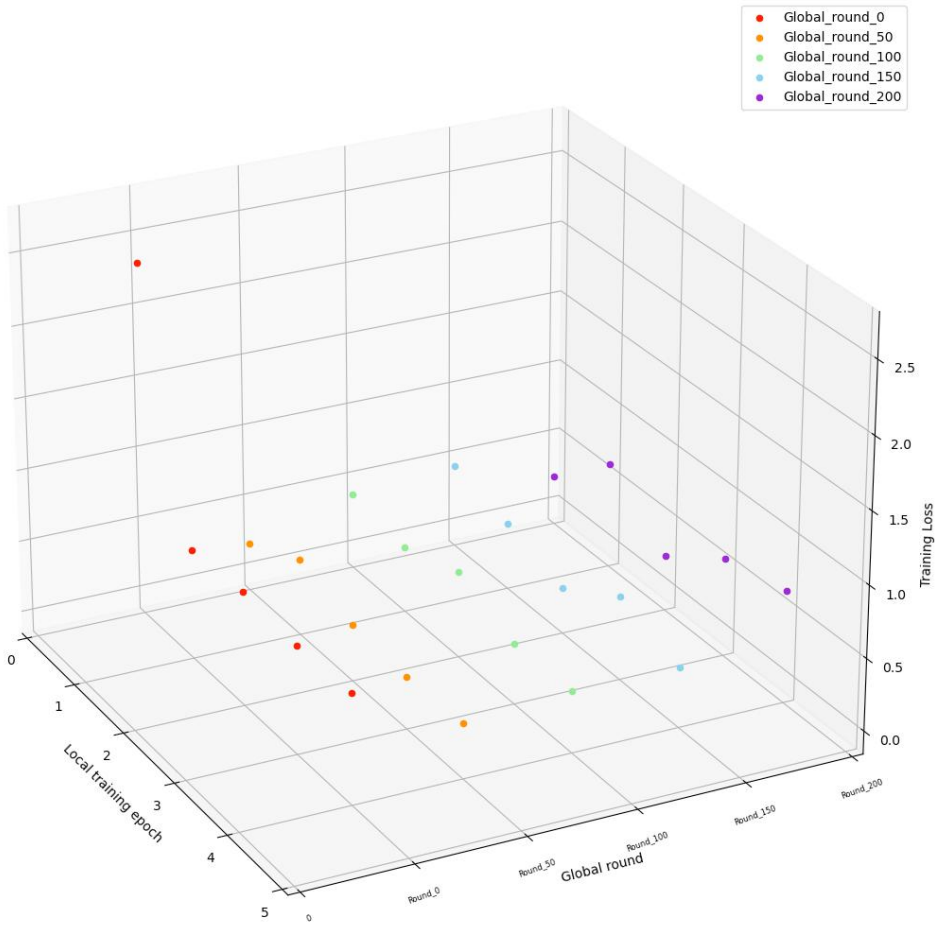


Figure 21(a). Local model training loss on iid dataset

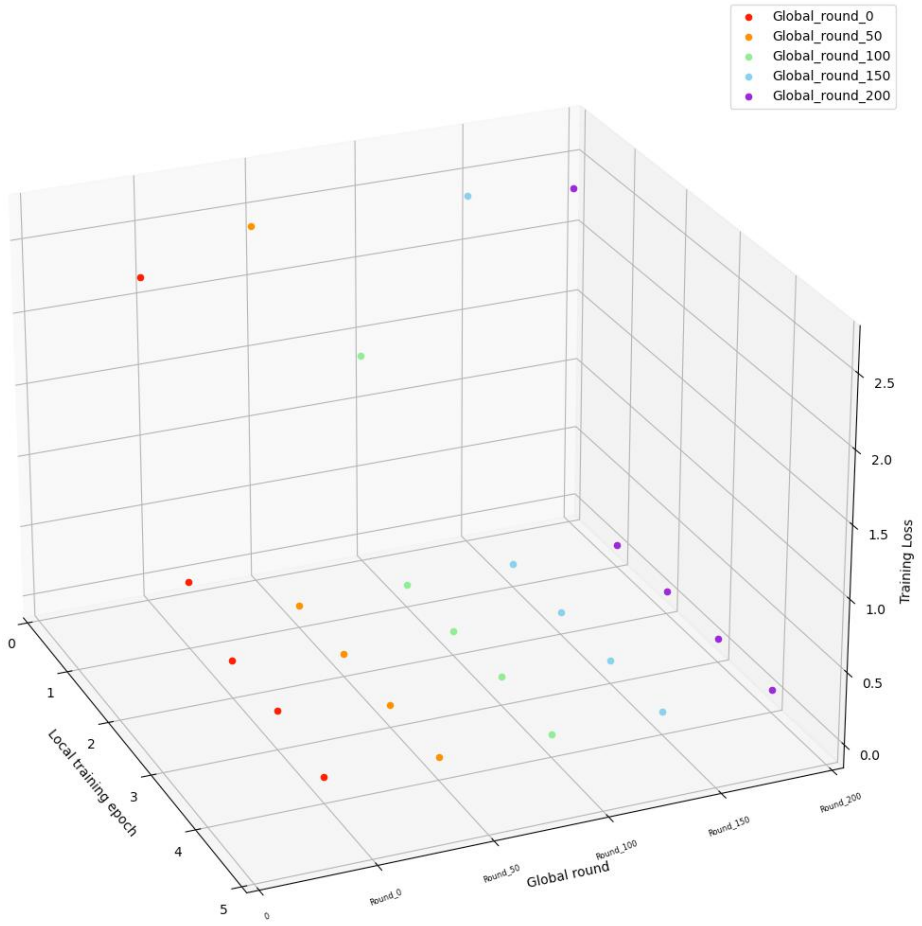


Figure 21(b). Local model training loss on non_iid dataset

Figure 21(a) shows the dot plot of training loss on each local epoch of a single client. Loss value at the first section indicating the first global round, and an obvious drop express the local model is getting convergence on local dataset. Based on FedAvg algorithm, the parameters of local model will be updated to server side for aggregation, and then assign back to random selected clients. The following sections after 50 epochs training, showing the value of loss function to be at a relatively low level even at the beginning of local training, which means the global model has been well trained to fit each single clients on iid dataset. So that the global model get converged at the beginning of the local training process.

However global model on non_iid dataset reveals a opposite behavior. Figure 21(b) shows the loss value on first local epoch is much larger than that on other epochs at each

sections. This means the local model, aggregated from last global round, is trained to converge on client's local data at every epoch, and is what happened when the global model cannot fit the local dataset well. Considering the loss value of the four later local epochs is extremely low, the model may be overfitting at client's dataset and this phenomenon will further reduce global accuracy when the parameters of a single client model is updated to server side and then broadcast to other clients on non_iid assumption.

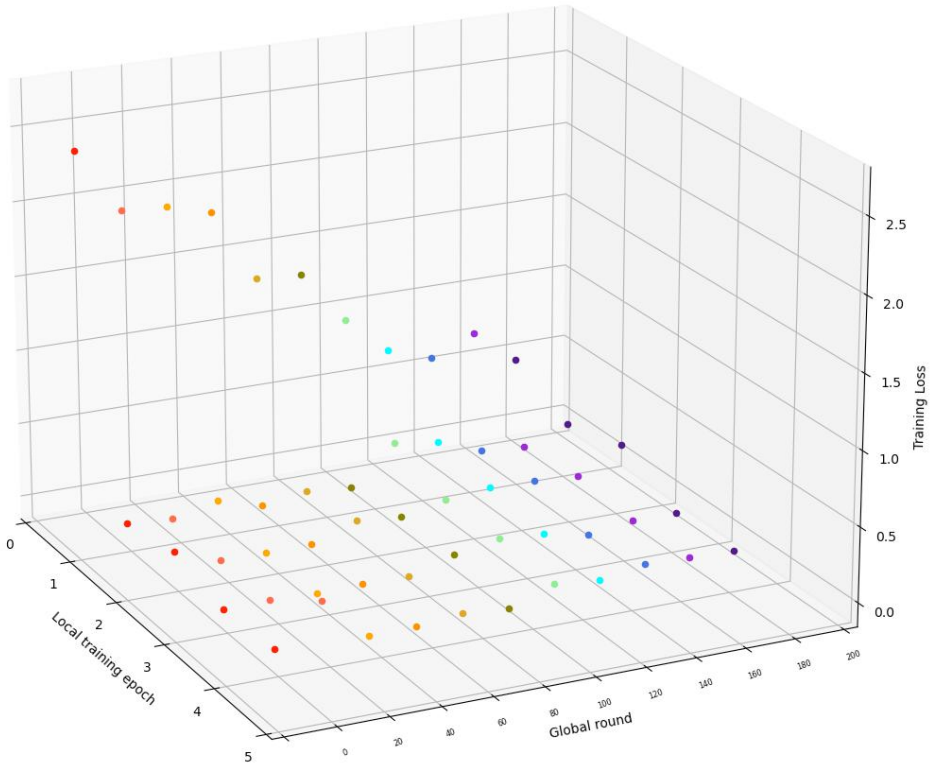


Figure 22. Local model training loss on dynamic dataset

Figure 22 recording local loss value every 20 global epoch on MNIST dataset, where GAN based data augmentation is set to be dynamic augmentation mode as shown in the blue curve at figure 18. The reduce of local loss value as the global training process, corresponding to the increasing test accuracy curve at around 100 epoch. This can be considered as another evidence on the effectiveness of data augmentation method on improving model performance of federated learning algorithm on non_iid dataset.

Conclusion

This work focus on solving the impact of statistical heterogeneity on data distribution between clients in Federated Learning. Theoretically, by generating simulated data using the Adversarial Generation Network (GAN), defects of statistical heterogeneity between clients should be smoothed in the distributed model, thereby improve the test accuracy as well as the convergence rate in distributed learning framework

In this work, a distributed GAN is trained to process data augmentation on federated learning algorithm under non_iid dataset, using multi_path generator to fit the uneven data distribution between clients. Training data is partitioned by Dirichlet distribution function with hyper-parameter non_iid_alpha for a quantitative analysis of the relationship between non_iid data distribution and the global model test accuracy in federated learning algorithm. Model performance can be mitigated by implementing a multi_path generator GAN based data augmentation, from a strong non_iid status (non_iid_alpha = 0.01) to an approximately iid status (non_iid_alpha = 100). Performance evaluation is presented on full batch gradient descent as well as FedAvg framework on both MNIST and Cifar10 dataset.

Reference

[1]. Voigt P, Von dem Bussche A. The eu general data protection regulation (gdpr)[J]. A Practical Guide, 1st Ed., Cham: Springer International Publishing, 2017, 10: 3152676.

[2]. McMahan B, Moore E, Ramage D, et al. Communication-efficient learning of deep networks from decentralized data[C]//Artificial Intelligence and Statistics. PMLR, 2017: 1273-1282.

[3]. Li T, Sahu A K, Talwalkar A, et al. Federated learning: Challenges, methods, and future directions[J]. IEEE Signal Processing Magazine, 2020, 37(3): 50-60.

[4]. McMahan H B, Ramage D, Talwar K, et al. Learning differentially private recurrent language models[J]. arXiv preprint arXiv:1710.06963, 2017.

[5]. Li L, Fan Y, Tse M, et al. A review of applications in federated learning[J]. Computers & Industrial Engineering, 2020: 106854.

[6]. Zhao Y, Li M, Lai L, et al. Federated learning with non-iid data[J]. arXiv preprint arXiv:1806.00582, 2018.

[7]. McDonald R, Hall K, Mann G. Distributed training strategies for the structured perceptron[C]//Human language technologies: The 2010 annual conference of the North American chapter of the association for computational linguistics. 2010: 456-464.

[8]. Woodworth B, Patel K K, Srebro N. Minibatch vs local sgd for heterogeneous distributed learning[J]. arXiv preprint arXiv:2006.04735, 2020.

[9]. Li T, Sahu A K, Zaheer M, et al. Federated optimization in heterogeneous networks[J]. arXiv preprint arXiv:1812.06127, 2018.

[10]. Wang J, Liu Q, Liang H, et al. Tackling the objective inconsistency problem in heterogeneous federated optimization[J]. arXiv preprint arXiv:2007.07481, 2020.

[11]. Reddi S, Charles Z, Zaheer M, et al. Adaptive federated optimization[J]. arXiv preprint arXiv:2003.00295, 2020.

[12]. McMahan H B, Streeter M. Adaptive bound optimization for online convex optimization[J]. arXiv preprint arXiv:1002.4908, 2010.

[13]. Kingma D P, Ba J. Adam: A methodfor stochastic optimization[C]//International Conference onLearning Representations (ICLR). 2015.

[14]. Reddi S, Zaheer M, Sachan D, et al. Adaptive methods for nonconvex optimization[C]//Proceeding of 32nd Conference on Neural Information Processing Systems (NIPS 2018). 2018.

[15]. Deng Y, Kamani M M, Mahdavi M. Distributionally Robust Federated Averaging[J]. arXiv preprint arXiv:2102.12660, 2021.

[16]. He C, Li S, So J, et al. Fedml: A research library and benchmark for federated machine learning[J]. arXiv preprint arXiv:2007.13518, 2020.

[17]. Karimireddy S P, Kale S, Mohri M, et al. SCAFFOLD: Stochastic controlled averaging for federated learning[C]//International Conference on Machine Learning. PMLR, 2020: 5132-5143.

[18]. Zhao Y, Li M, Lai L, et al. Federated learning with non-iid data[J]. arXiv preprint arXiv:1806.00582, 2018.

[19]. Reisizadeh A, Farnia F, Pedarsani R, et al. Robust federated learning: The case of affine distribution shifts[J]. arXiv preprint arXiv:2006.08907, 2020.

[20]. Ghosh A, Chung J, Yin D, et al. An efficient framework for clustered federated learning[J]. arXiv

preprint arXiv:2006.04088, 2020.
[21]. He C, Annavaram M, Avestimehr S. Group Knowledge Transfer: Federated Learning of Large CNNs at the Edge[J]. Advances in Neural Information Processing Systems, 2020, 33.
[22]. Briggs C, Fan Z, Andras P. Federated learning with hierarchical clustering of local updates to improve training on non-IID data[C]//2020 International Joint Conference on Neural Networks (IJCNN). IEEE, 2020: 1-9.
[23]. Goodfellow I J, Pouget-Abadie J, Mirza M, et al. Generative adversarial networks[J]. arXiv preprint arXiv:1406.2661, 2014.
[24]. Gui J, Sun Z, Wen Y, et al. A review on generative adversarial networks: Algorithms, theory, and applications[J]. arXiv preprint arXiv:2001.06937, 2020.
[25]. Mirza M, Osindero S. Conditional generative adversarial nets[J]. arXiv preprint arXiv:1411.1784, 2014.
[26]. Zhu J Y, Park T, Isola P, et al. Unpaired image-to-image translation using cycle-consistent adversarial networks[C]//Proceedings of the IEEE international conference on computer vision. 2017: 2223-2232.]
[27]. Arjovsky M, Chintala S, Bottou L. Wasserstein generative adversarial networks[C]//International conference on machine learning. PMLR, 2017: 214-223.
[28]. Radford A, Metz L, Chintala S. Unsupervised representation learning with deep convolutional generative adversarial networks[J]. arXiv preprint arXiv:1511.06434, 2015.
[29]. Radford A, Metz L, Chintala S. Unsupervised representation learning with deep convolutional generative adversarial networks[J]. arXiv preprint arXiv:1511.06434, 2015.
[30]. Hoang Q, Nguyen T D, Le T, et al. MGAN: Training generative adversarial nets with multiple generators[C]//International conference on learning representations. 2018.
[31]. Fan C, Liu P. Federated generative adversarial learning[C]//Chinese Conference on Pattern Recognition and Computer Vision (PRCV). Springer, Cham, 2020: 3-15.
[32]. Yonetani R, Takahashi T, Hashimoto A, et al. Decentralized Learning of Generative Adversarial Networks from Non-iid Data[J]. arXiv preprint arXiv:1905.09684, 2019.
[33]. Rasouli M, Sun T, Rajagopal R. Fedgan: Federated generative adversarial networks for distributed data[J]. arXiv preprint arXiv:2006.07228, 2020.
[34]. Augenstein S, McMahan H B, Ramage D, et al. Generative models for effective ML on private, decentralized datasets[J]. arXiv preprint arXiv:1911.06679, 2019.
[35]. Zhuo H H, Feng W, Xu Q, et al. Federated reinforcement learning[J]. arXiv preprint arXiv:1901.08277, 2019, 1.
[36]. Zhan Y, Zhang J. An Incentive Mechanism Design for Efficient Edge Learning by Deep Reinforcement Learning Approach[C]//IEEE INFOCOM 2020-IEEE Conference on Computer Communications. IEEE, 2020: 2489-2498.
[37]. Zheng H, Wei P, Jiang J, et al. Cooperative Heterogeneous Deep Reinforcement Learning[J]. arXiv preprint arXiv:2011.00791, 2020.
[38]. Wang H, Kaplan Z, Niu D, et al. Optimizing federated learning on non-iid data with reinforcement learning[C]//IEEE INFOCOM 2020-IEEE Conference on Computer Communications. IEEE, 2020: 1698-1707.
[39]. Hardy C, Le Merrer E, Sericola B. Md-gan: Multi-discriminator generative adversarial networks for distributed datasets[C]//2019 IEEE international parallel and distributed processing symposium

(IPDPS). IEEE, 2019: 866-877.
[40]. Jeong E, Oh S, Kim H, et al. Communication-efficient on-device machine learning: Federated distillation and augmentation under non-iid private data[J]. arXiv preprint arXiv:1811.11479, 2018.
[41]. Shahbazi M, Huang Z, Paudel D P, et al. Efficient Conditional GAN Transfer with Knowledge Propagation across Classes[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021: 12167-12176.
[42]. Seo H, Park J, Oh S, et al. Federated knowledge distillation[J]. arXiv preprint arXiv:2011.02367, 2020.
[43]. Li D, Wang J. Fedmd: Heterogenous federated learning via model distillation[J]. arXiv preprint arXiv:1910.03581, 2019.
[44]. Romera-Paredes B, Torr P. An embarrassingly simple approach to zero-shot learning[C]//International conference on machine learning. PMLR, 2015: 2152-2161.
[45]. Zhang L, Yuan X. FedZKT: Zero-Shot Knowledge Transfer towards Heterogeneous On-Device Models in Federated Learning[J]. arXiv preprint arXiv:2109.03775, 2021.
[46]. Vyas M R, Venkateswara H, Panchanathan S. Leveraging seen and unseen semantic relationships for generative zero-shot learning[C]//European Conference on Computer Vision. Springer, Cham, 2020: 70-86.
[47]. Smith V, Chiang C K, Sanjabi M, et al. Federated multi-task learning[J]. arXiv preprint arXiv:1705.10467, 2017.
[48]. Hsu T M H, Qi H, Brown M. Measuring the effects of non-identical data distribution for federated visual classification[J]. arXiv preprint arXiv:1909.06335, 2019.
[49]. Yurochkin M, Agarwal M, Ghosh S, et al. Bayesian nonparametric federated learning of neural networks[C]//International Conference on Machine Learning. PMLR, 2019: 7252-7261.
[50]. Lin T, Kong L, Stich S U, et al. Ensemble distillation for robust model fusion in federated learning[J]. arXiv preprint arXiv:2006.07242, 2020.
[51]. Shoham N, Avidor T, Keren A, et al. Overcoming forgetting in federated learning on non-iid data[J]. arXiv preprint arXiv:1910.07796, 2019.
[52]. Yao X, Sun L. Continual Local Training for Better Initialization of Federated Models[C]//2020 IEEE International Conference on Image Processing (ICIP). IEEE, 2020: 1736-1740.
[53]. Sattler F, Müller K R, Samek W. Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints[J]. IEEE transactions on neural networks and learning systems, 2020.
[54]. Xie M, Long G, Shen T, et al. Multi-center federated learning[J]. arXiv preprint arXiv:2108.08647, 2021.
[55]. Chawla N V, Bowyer K W, Hall L O, et al. SMOTE: synthetic minority over-sampling technique[J]. Journal of artificial intelligence research, 2002, 16: 321-357.
[56]. He H, Bai Y, Garcia E A, et al. ADASYN: Adaptive synthetic sampling approach for imbalanced learning[C]//2008 IEEE international joint conference on neural networks (IEEE world congress on computational intelligence). IEEE, 2008: 1322-1328.