



电子科技大学
格拉斯哥学院
Glasgow College, UESTC

FINAL YEAR PROJECT REPORT
BACHELOR OF ENGINEERING

FedtriGAN: Federated Semi-supervised Learning based on Triple GAN

Student: Liwei CHE

GUID: 2357574C


1st Supervisor: Junming Shao

2nd Supervisor: Joao Ponciano

2020-2021

Coursework Declaration and Feedback Form

This Student should complete and sign this part

Student Name: Liwei CHE	Student GUID: 2357574C
Course Code : UESTC4006P	Course Name : INDIVIDUAL PROJECT 4
Name of 1st Supervisor: Junming Shao	Name of 2nd Supervisor: Joao Ponciano
Title of Project: FedtriGAN:Federated Semi-supervised Learning based on Triple GAN	
Declaration of Originality and Submission Information	
<p><i>I affirm that this submission is all my own work in accordance with the University of Glasgow Regulations and the School of Engineering requirements</i></p> <p><i>Signed (Student) :</i></p> <p><i>I affirm this submission is completed by the student independently and the quality of the submission meets the requirements for graduation. I consent to the student taking part in the oral presentation.</i></p> <p><i>Signed (1st Supervisor) :</i></p>	 <p>UESTC4006P TEC-IT.COM</p>
Date of Submission :	
<i>Feedback from Lecturer to Student – to be completed by Lecturer or Demonstrator</i>	
Grade Awarded: Feedback (as appropriate to the coursework which was assessed):	
Lecturer Demonstrator:	Date returned to the Teaching Office:

Abstract

Federated learning has shown great potentials for security and privacy protection and dealing with the distributed data. While two significant problems still exist in federated learning: (1) The scarce of both amount and label rate of data is still severe in many of the circumstances, which as the solution to the actual deployment application and the problem is fatal; and (2) dispersion of distributed data and random updating by clients caused weight divergence in server updating, making the optimization process of a global model less stable and efficient than that of the local setting. To tackle the federated semi-supervised learning problems under the scenario that labeled data is exceptionally scarce, we proposed `FedtriGAN` with triple generative adversarial networks and `FedtriGAN-CE` to solve the communication cost issue. Our results on various datasets demonstrate that (1) `FedtriGAN` with utilizing a small amount of labeled data can achieve the state-of-the-art classification results among the existing federated semi-supervised methods; (2) `FedtriGAN-CE`, as can be applied to almost all existing federated GANs methods, significantly improves the communication efficiency and privacy preservation ability of the global model in the federated learning framework

Acknowledgements

I want to express my sincere thanks to my supervisor, Prof. Junming Shao, who provided valuable advice and patient guidance on my work. Given sufficient freedom to finish the project at my own pace, I have been provided with detailed solutions and in-depth insights from my supervisor to many problems I encountered in completing my work, which helped me avoid many detours.

Contents

Coursework Declaration and Feedback Form	i
Abstract	ii
Acknowledgements	iii
1 Introduction	1
2 Related Work	3
2.1 Federated Learning	3
2.2 Semi-supervised Learning	4
2.3 Generative Adversarial Networks	4
3 Federated Semi-supervised Learning	6
4 Methodology	8
4.1 FedtriGAN training process	8
4.1.1 Local Update	8
4.1.2 Server update	9
4.2 FedtriGAN with communication efficiency	10
4.2.1 Local Training Process	11
4.2.2 Server Training Process	12
5 Experiment	14
5.1 Experimental Setting	14
5.1.1 Datasets	14
5.1.2 Data Distribution Setting	14
5.1.3 Baselines	15
5.1.4 Data Augmentations	15
5.2 Implementation	15
5.3 Performance Evaluation for the IID Setting	15
5.4 Performance Evaluation for the Non-IID Setting	16
6 Discussion	19
7 Conclusion	20
7.1 Suggestions for further work	20

1 Introduction

Federated Learning [1], [2], [3] has furnished a concrete solution to the training of machine learning models among decentralized data deployment networks with relative stable privacy preservation. A central server helps multiple clients to collaborate on ensemble a global model, which performs better than any of the local models. This distributed framework contributes a series of advantages on protection of data privacy, access rights and security.

Several practical issues still shackle federated learning aggregation effect performance. For instance, the clients tend to generate large amounts of data which lacks of labels or only contains few of them. In local devices, the data labels are not only scarce in number, but also occupies a very small percentage of the total local private data. While existing federated methods mainly focus on the supervised scenario where client data is well labeled. It is crucial to get full access to the information included inside the unlabeled data in order to improve the global model performance.

The traditional semi-supervised algorithms and data enhancement methods are difficult to play an effective role under the limitation of extreme scarcity of data tags and amount. For example, the semi-supervised algorithm represented by pseudo label methods [4], due to the poor performance of the pre-trained model after supervised learning and training, it would introduce error messages in the subsequent marking process of pseudo label, seriously affecting the learning effect.

On the other hand, the distribution of local data and global updating significantly impact the performance of ensemble models. Most of federated learning work focus on the independent identically distributed(IID) data among clients, while those may results in an unpleasant result on Non-independent identically distribution(Non-IID).The federated learning aggregation method represented by FedAvg [1] is still defective compared with the Centralized Setting, which is largely due to the problems such as the random upload and update of clients and the gradient divergence caused by different data distribution.Different data distributions can greatly affect the aggregation effect of federated learning global models, especially under the Non-IID condition, where model training is highly unstable.

While a few challenges are still in need of solving in the field of federated learning. We list them as below:

- *Communication efficiency and effectiveness of the algorithm.* Considering many clients in most of the application for federated learning, a stable and robust communication connection between the server and user devices is highly costly and rigorous. The algorithm must be effective enough to handle the learning process under the limitation of the computation capacitance and internet transmission ability for the hardware.
- *Statistical discrepancy between the local and global views.* A fundamental difference between federated learning and the centralized learning method is that the server can not access the global data distribution. In other words, the global parameters optimization is achieved through local optimization of multiple clients, which causing the performance gap between the centralized learning and federated one. Besides, the NonIID challenges and the introduction of unlabeled data both make the federated learning problem more complicated and challenging.
- *Applicability and generalization of the designed framework.* Simply porting a previous semi-supervised learning effort to a federated learning framework often underperforms or is difficult to adapt to a distributed framework. One of our design purposes is to make

the model have strong adaptability and generalization ability so that it can be easily transplanted while performing well on the data set.

To address those problems, it is efficient to add unsupervised learning methods represented by generative adversarial networks, effectively derive information from unlabeled data, and achieve beyond limitations of rarely labeled data.[5] proposed a three-player model for improving the stability of the generative adversarial networks performance on semi-supervised learning. Based on that, we propose `FedtriGAN`, an efficient method to solve the federated semi-supervised problems under both IID and Non-IID scenarios. The discriminator and generator will assist the classifier in the classification training of local private data with limited labels. The proposed model has been validated on several benchmark image data sets and achieved a state-of-art classification result among the existing federated semi-supervised learning methods. Besides, the additional training and ensemble techniques have been explored to leverage the divergence and instability during global model training. The contributions of this paper are listed as below:

- `FedtriGAN`: A federated semi-supervised learning method, which performs well for both IID and NonIID scenarios, especially for a circumstance that the labeled data is extremely rare. For each client, a classifier model will collaborate with a discriminator model and a generator model to continuously dig the hidden information inside the unlabeled data and scarce labeled data.
- `FedtriGAN-CE`: Considering the adversarial process of the GANs, the generator model is deployed in the server. While for each client, a discriminator and a classifier model are training locally. Except for the parameters of the classifier model, the generated data and output of the discriminator are required to be transmitted between the server and the client, which significantly reduced the communication efficiency.
- This work validated `FedtriGAN` over three image data sets under four different data distribution settings. Also, the further discussion has been done for the mitigated global model update instability caused by Non-IID and random update frequency of clients

2 Related Work

2.1 Federated Learning

Federated Learning provides an efficient and privacy-preserved collaboration strategy for the mutually training between different data owners, such as distributed data center, customers and diverse institutions. The majority of the Federated Learning works focus more on supervised learning scenario and solving three challenges: statistical heterogeneity [6, 7, 8], system constraints [9, 10, 11], and trustworthiness [12, 13, 14].

Federated Learning (FL) is committed to solving the problem of data islands, realizing joint modeling under the premise of meeting user privacy protection, data security, data confidentiality and government regulations, and improving the effectiveness of AI models. Its predecessor was initially proposed by Google in 2016 to solve the problem of the local update model of Android mobile phone end users.

FL is a framework for machine learning. Virtual models are designed to solve the problem of collaboration between different data owners without sharing data under this framework. The model is the best way for all participants to combine data, and their respective regions use the model to serve local goals. Federated learning generally requires that the modeling outcome be sufficiently close to the centralized model, i.e., the result of aggregating data from multiple data owners in one location for modeling. Each member has the same identity and status under the federation mechanism, and a data-sharing strategy can be developed. Since the information is not really transferred, it will not compromise user privacy or alter data requirements.

A FL framework for deep learning models usually updates the virtual models, aka global model, by performing an aggregation method on the server using the parameters uploaded from local clients. An update of the global model is called a communication round which contains local training and server aggregation processes. In the following part, three baseline works will be introduced for further comparison with the proposed work in this paper.

FedAvg [1], proposed by McMahan et al., gives the first presentation on how to conduct federated learning of deep networks when data is deployed in distributed condition based on iterative model averaging under the communication cost constraints. The server performs model averaging, and each client updates local models using stochastic gradient descent. This method has been empirically shown to be resilient to unbalanced and Non-IID data distributions.

FedSem [4], proposed by Abdullatif Albaseer, et al., combines pseudo labeling idea with federated semi-supervised learning problems in the smart city application. The training process is divided into two phases in this research. The local model obtains a certain classification capacity in step one, with the current labeled data to supervise the training process. The model uses a pseudo-label to label the local unlabeled data with the expected value. In phase two, the entire federated system will repeat the same training process as in phase one, using both real and pseudo-label data.

FedMatch [15] adopts the idea of consistency regularization and designs two kinds of loss functions to guide the training of the model, namely Inter-client Consistency Loss and Data-level Consistency Regularization. The concept behind consistency regularization is that the predictor's performance should be as consistent as possible between an original sample and the data-enhanced version (the idea of consistency). Several versions of other clients will be sent to the client as helper agents during the server parameter delegation process, in addition to the original client model. The goal of local unsupervised training is to keep the difference between the local model's prediction results and the labels given by each consensus model to a minimum. In this paper, we mainly focus on the statistical heterogeneity challenge on semi-supervised learning scenarios in both IID and Non-Iid settings, aiming to achieve better model performance

on classification.

Previous work has made a series of attempts to tackle this challenge from different angles. [6] use a shared server-stored dataset to help the clients achieve higher performance in Non-iid setting; [8] applied adjustment on the SGD convergence of FL; [7] add regularization terms on the loss function during the local training process on the clients to constrain the divergence between the global model and local ones.

While, in a semi-supervised scenario, the introduction of the unlabeled data will significantly increase the difficulty of the problem. For Federated Learning, compared with other fields, the work and attention on semi-supervised learning are relatively less popular. [4] proposed a simple two-phase pseudo-labeling based method for semi-supervised learning application; [15] introduced the Inter-Client Consistency Loss and Additive Parameter Decomposition to handle the unlabeled data; [16] proposed Fedswap exchange local models at the cloud. However, the existing methods are efficient in a way while may violate the privacy of the clients or have poor performance with scarce labeled data, which are severe disadvantages for a federated semi-supervised learning problem.

2.2 Semi-supervised Learning

Semi-supervised Learning is a research field of practical significance and value to extract effective information from unlabeled data and help the model achieve better training effect and performance. [17]

The previous SSL work shows a series of diverse and coherent solutions. A very intuitive approach is Pseudo label, which uses pre-trained model to denote the unlabeled data to labeled one. [18] add a dynamic decision threshold to help the model labeling the data. Another effective and well-known strategy is the add consistency regularization on the training loss[19, 20, 21, 22, 23]. Besides, Generative Adversarial Networks, as a trendy architecture with the combination of unsupervised and supervised learning, shows great potentials on the SSL problems.[24] used discriminator network to play the role of classifier; [25] proposed Bad GANs to generate negative samples to help the discriminator achieve more robust and accurate decision boundary; [5] presented an SSL method based on three neural networks, which characterize the conditional distributions between images and labels.[26] suggests that the flat platform of SGD leads to the convergence dilemma of consistency-based SSL.UDA [27],ReMixMatch [28], and Fixmatch [29] mixed plenty of practical methods and did further exploration.

2.3 Generative Adversarial Networks

GANs[30], or Generative Adversarial Networks, is a form of unsupervised learning that involves pitting two neural networks against each other. A generator network and a discriminator network make up the GANs. The generator network takes random samples from the latent space as input, and the output must closely resemble the actual samples in the training set. The discriminator network takes the real sample or the generator network's output as input, and its aim is to separate the generator network's output from the real samples as much as possible. As much as possible, the generator network must deceive the discriminator network. The two networks are actively adjusting their parameters in order to avoid colliding. The ultimate aim is to render the discriminator network incapable of determining whether the generator network's output is correct. Usually, D and G play the following two-player minimax game with value function V (G, D)

$$\min_G \max_D U(D, G) = E_{x \sim p(x)} [\log(D(x))] + E_{z \sim p_z(z)} [\log(1 - D(G(z)))] \quad (1)$$

Semi-supervised GAN [24] made D to predict which of $N+1$ classes the input belongs to, during training, with an extra class added to correspond to G 's outputs. It demonstrates that this approach can be used to construct a more data-efficient classifier and can generate higher-quality samples than a traditional GAN.

Triple GAN [31] proposed triple generative adversarial net (Triple-GAN), a three-player system that includes a generator, a discriminator, and a classifier. The discriminator focuses solely on defining fake image-label pairs, while the generator and classifier describe the conditional distributions between images and labels. They designed compatible utilities to ensure that the distributions characterized by the classifier and the generator both converge to the data distribution, which greatly enhanced the performance of the GAN on semi-supervised learning scenarios.

Training Federated GAN [32] provides a theoretical guarantee for the training of a federated GAN that jointly trains a centralized generator network and multiple private discriminator networks hosted at different clients. The proposed Universal Aggregation simulates a centralized discriminator via carefully aggregating the mixture of all private discriminators. This gives confidence on the training of the federated GAN can allow the generator learnt the desired data distribution. In this paper, our work mainly focuses on the take advantage of the collaboration between the GANs model and the classifier model.

In the following sections of this paper, the definition of FedSSL problems will be introduced in Section 3. And the detail of the proposed methods `FedtriGAN` and `FedtriGAN-CE` will be presented with their training process in Section 4. The experiment results and the analysis will follow in Section 5.

3 Federated Semi-supervised Learning

Aiming to learn an intensified global model without violation of the data privacy, **Federated Learning** provides an efficient collaboration learning strategy for distributed data distribution scenarios.

Let G denotes the global model and $\mathcal{L} = \{l_k\}_{k=1}^N$ denote a set of local models for N clients. For the k -th client, $\mathcal{D}_L^k = \{(\mathbf{x}_1^k, y_1^k), \dots, (\mathbf{x}_n^k, y_n^k)\}$ represents a set of labeled data, where \mathbf{x}_i^k ($i \in \{1, \dots, n\}$) is a data instance, $y_i^k \in \{1, \dots, C\}$ is the corresponding label, and C is the number of label categories.

Common **Federated Learning** usually uses two stages as a communication round, **Local Training** and **Server update**.

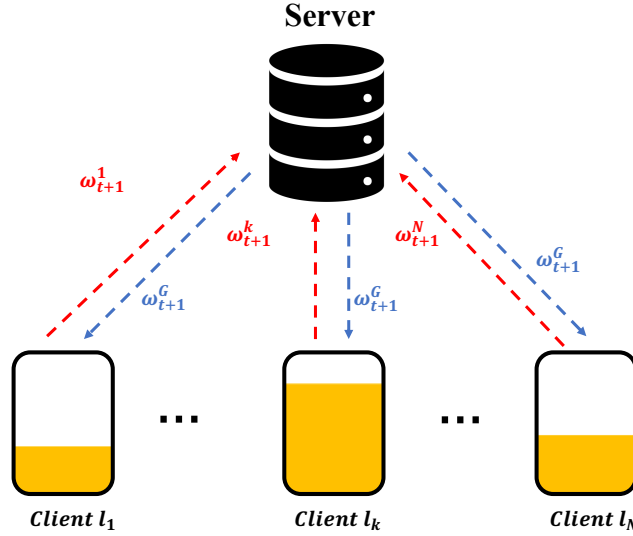


Figure 1: Federated Learning Framework

Local Training Represented by FedAvg, in local training phase, each client will utilize their own local dataset to train the local model initialized by the same server. We use $[\omega_k^i]$ to represent the i -th round local model parameter of client k .

Server Update For each communication round, the server will randomly pick up K clients for participating the parameter aggregation in the server to update the global model.

$$\omega^G = \sum_{k=1}^K \frac{\omega^k}{K} \quad (2)$$

where ω^G denotes the global model parameter, k denoting the index of the client; K denoting the total number of the clients that participate in the aggregation process.

Federated Semi-supervised Learning (FedSSL) investigates the hidden information contained in the unlabeled data aiming to enhance the model performance above the federated supervised learning. This goal provides an explicit target to reduce the total training loss of labeled and unlabeled data.

$$\min Loss_{total} = Loss_l(\mathcal{D}_L^k) + Loss_u(\mathcal{D}_U^k) \quad (3)$$

Here, \mathcal{D}_L^k and \mathcal{D}_U^k denote labeled dataset and unlabeled dataset of client k separately; $Loss_l$ is the loss of labeled data, and $Loss_u$ is for the unlabeled data.

In the server, a parameter aggregation process for the update of global model will be delivered similarly to Federated Supervised Learning

$$\omega^G = \sum_{k=1}^K \alpha^k \omega^k \quad (4)$$

k denotes the index of the client; K is the total number of the clients that participate in the aggregation process; α^k , represents the aggregation weight of client k , which may be decided by the importance of the client or the data distribution.

A complete communication round starts from the download of the global networks and ends with the aggregation for the updated global networks. FedSSL usually makes adjustments and modulations based on this framework to realize the joint learning on labeled data and unlabeled data.

While the challenges for FedSSL is not only limited in the existence of the labels for the data but also the data distribution. Traditional machine learning methods usually have a common hypothesis that all the data utilized in their work are under independent and identically distribution (IID), while in more realistic consideration the data is more common in the presentation of Non-IID.

4 Methodology

In order to tackle the federated semi-supervised learning problems under the scenario that tag data is especially scarce, we proposed FedtriGAN, a three-player game deployed in the local client where generator model, discriminator model and classifier model collaborate to utilize unlabeled data efficiently. Considering the communication efficiency, it could be a significant cost for the server if we upload all the parameters of the three models. To solve this problem, we also proposed a communication cost-friendly deployment strategy for FedtriGAN. We will introduce the detail of our model in the following sections.

The whole training process of FedtriGAN is similar to the previous federated learning work, divided into two important stages, local training process and global ensemble process.

4.1 FedtriGAN training process

4.1.1 Local Update

The local training process contains three phases changed depending on the global communication rounds number. Two preset communication rounds numbers decided by experimental analysis will be the division point of different training phases. In the first training phase, the three networks focus on the labeled data to do the pre-training on both the classifier model and two adversarial networks. After that, the Discriminator network and Generator network will gradually get involved in the training process of Classifier to improve its performance via utilizing the unlabeled information.

Phase I: Pre-training. For the first phase, the Classifier will be trained using labeled data. Discriminator and Generator will be able to access both labeled and unlabeled data. In this phase, the local GANs model in the client will not get involved in the training of the Classifier.

In this phase, the classifier model will be trained with the below loss function:

$$L_l(D_L^k) = \min \left[E_{p_l} \left(\log C(f(\mathbf{x}_i^k; \omega^k), y_i^k) \right) \right], \quad (5)$$

where, D_L^k is the labeled training data of client k ; x_i^k is the i -th labeled data and y_i^k is the corresponding label. And, the adversarial losses between the Generator model and Discriminator model is defined as below:

$$\begin{aligned} \min_{C, G} \max_D U(C, G, D) = & E_{(x, y) \sim p(x, y)} [\log D(x, y)] + \alpha E_{(x, y) \sim p_c(x, y)} [\log(1 - D(x, y))] \\ & + (1 - \alpha) E_{(x, y) \sim p_g(x, y)} [\log(1 - D(G(y, z), y))], \end{aligned} \quad (6)$$

where $\alpha \in (0, 1)$ controls the significance of classification and generation. It was as 0.5 for all the experiment of this paper.

Phase II: Discriminator Collaboration. When the total local training epochs reached a pre-defined threshold value, the client model will get into the second phase where the discriminator would take part into the training of the classifier. A regularization term related to Discriminator is added into the loss of the Classifier. This term is used to help identify the data that is difficult to classify, thus promoting the further convergence of the network. Thus, the loss function of Classifier model is updated as below:

$$L_C = \min \left[\alpha \sum_{(x_c, y_c)} p_c(y_c | x_c) \log(1 - D(x_c, y_c)) \right] + L_l, \quad (7)$$

Phase III: Generator Collaboration. When the total local training epochs reached the second predefined threshold value, the client model will get into the third phase where another regularization item related to the fake image generated by the Generator are involved in the training of Classifier – using the relatively trained-well Generator, generated data is added to the training process, so as to improve the Classifier’s robustness.

$$\begin{aligned} \min_{C,G} \max_D \tilde{U}(C,G,D) = & E_{(x_l,y_l) \sim p(x,y)} [\log D(x,y)] + \alpha E_{(x_u,y_u) \sim p_c(x,y)} [\log(1 - D(x,y))] \\ & + (1 - \alpha) E_{(x,y) \sim p_g(x,y)} [\log(1 - D(G(y_g,z), y_g))] \\ & + E_{(x_l,y_l) \sim p(x,y)} [-\log p_c(y | x)], \end{aligned} \quad (8)$$

which is added with L_l in the base of 6. The target function of classifier model is:

$$L_C = \alpha \sum_{(x_u,y_u)} p_c(y_u | x_u) \log(1 - D(x_c, y_c)) + L_l + \alpha_{\mathcal{D}} L_P, \quad (9)$$

$$L_P = E_{p_g} [-\log p_c(y_g | x_g)], \quad (10)$$

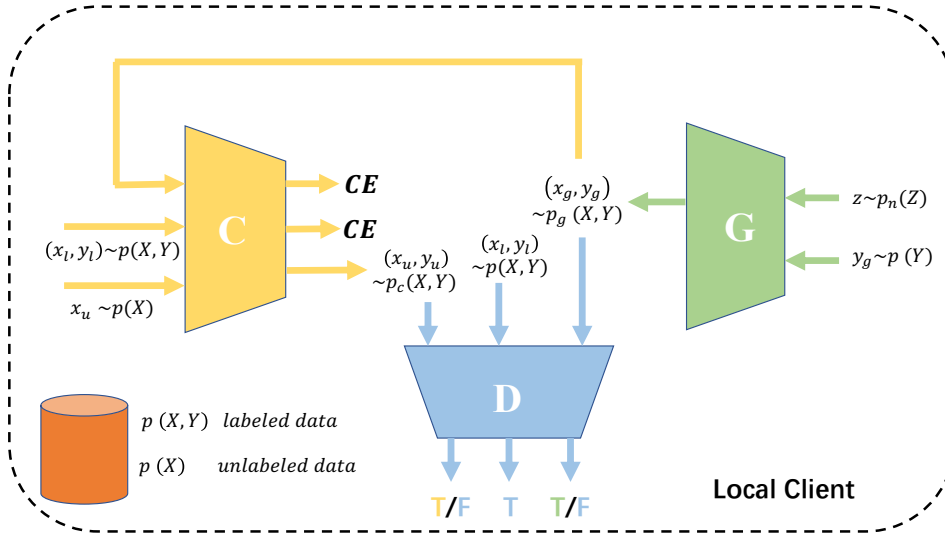


Figure 2: Local model of FedtriGAN. Networks of D, C and G are colored in blue, yellow and green respectively, with “T” denoting true, “F” denoting false and “CE” denoting the cross entropy loss for supervised learning. “T”s and “F”s are the adversarial losses and “CE”s are unbiased regularizations that ensure the consistency between p_g , p_c and p , which are the distributions defined by the generator, classifier and true data generating process, respectively.

4.1.2 Server update

. For each client k , let $[\omega_{Gi}^k, \omega_{Di}^k, \omega_{Ci}^k]$ denotes the learned parameters of the local Generator model, Discriminator model and Classifier model, separately in the i -th communication round, which will be uploaded to the server and used to aggregate the global models with

$$\omega_{model}^G = \sum_{k=1}^K \alpha_k \omega_{model}^k \quad (11)$$

where `model` is denoting the uploaded model type, which are Generator model, Discriminator model and Classifier model; k denoting the index of the client; N denoting the total number of the clients that participate into the aggregation process; α^k , which is smaller than 1, here for $1/K$, represents the aggregation weight of client k

After the weighted averaging, the server will send the new global models' parameters to all the clients, after that a full communication round finished.

Algorithm 1 FedtriGAN

Require: \mathcal{D}_L and \mathcal{D}_U

```

1: procedure COMMUNICATION ROUND
2:   Initialization:  $\omega_{G0}^k, \omega_{D0}^k, \omega_{C0}^k$  ▷ initialize weights
3:   for each communication round  $t = 1, \dots, T$  do
4:      $\mathcal{L}_t = \{l^k\}_{k=1}^{N_t} \leftarrow \mathcal{L} = \{l^k\}_{k=1}^N$  ▷ random selection of clients for server aggregation
5:     for each client  $k \in \mathcal{L}_t$  in parallel do
6:        $\Delta\omega_{NET(t+1)}^k, l_{NETt}^k \leftarrow \text{Local Update}(\omega_{NETt}^k, D_L^k, D_U^k, t)$ 
7:       where  $NET \in \{G, D, C\}$ 
8:     end for
9:      $\omega_{NET(t+1)}^G \leftarrow \omega_{NETt}^G + \frac{1}{N_t} (\sum_{i=1}^{N_t} \Delta\omega_{NET(t+1)}^k)$  ▷ server aggregation
10:  end for
11: end procedure
12:
13: function LOCAL UPDATE( $\omega_{Gt}^k, \omega_{Dt}^k, \omega_{Ct}^k, D_L^k, D_U^k, t$ )
14:   if  $t \geq T_2$  then ▷ Training Phase Judgement
15:      $L_C = \alpha \sum_{(x_u, y_u)} p_c(y_u | x_u) \log(1 - D(x_c, y_c)) + L_l + \alpha \mathcal{D} L_p$ 
16:   else if  $t > T_1$  then
17:      $L_C = \alpha \sum_{(x_c, y_c)} p_c(y_c | x_c) \log(1 - D(x_c, y_c)) + E_{p_l}(\log C(f(\mathbf{x}_i^k; \omega^k) y_i^k))$ 
18:   else
19:      $L_C = E_{p_l}(\log C(f(\mathbf{x}_i^k; \omega^k) y_i^k))$ 
20:   end if
21:   for  $i$  in local epochs do
22:     for  $B_1, B_2$  in  $\mathcal{D}_L^k, \mathcal{D}_U^k$  do
23:        $l_{NETt}^k \leftarrow L_{NET}(\omega_{NETt}^k, B_1, B_2)$  ▷ pseudo labeled loss computation
24:        $\omega_{NET(t+1)}^k \leftarrow \omega_{NETt}^k - \eta \nabla \mathcal{L}_{NETt}^k$  ▷ mini batch gradient descent
25:     end for
26:   end for
27:   return  $\omega_{G(t+1)}^k, \omega_{D(t+1)}^k, \omega_{C(t+1)}^k, l_{Gt}^k, l_{Dt}^k, l_{Ct}^k$  ▷ return updated weights and client loss
28: end function

```

4.2 FedtriGAN with communication efficiency

The normal training process of FedtriGAN is no doubt a really communication costing method though achieving impressive performance on results. We assume the use of the normal method is mainly based on a low-cost communication scenario with a robust and stable internet connection between the server and the clients. With the consideration of communication cost, we further proposed a communication efficiency friendly deployment for the three models, whose training detail is described as below.

Model Deployment Considering the adversarial training process, we realized that there are no requirements for the Generator model to get access to the real datasets. The essential exchanging information between Generator and Discriminator for training is mainly generated data, discriminator output and related loss regularization. This deployment can also make sure that the distribution learned by the global generator model is the overall data distribution combined by all the data shades located in clients instead of the local one.

Based above analysis, the significant difference between FedtriGAN-CE and FedtriGAN is that one global Generator model is put in the server in the duty of generation of fake data and adversarial training with the local Discriminator model, except one for each client. In this context, the clients will not upload model parameters of the local Discriminator. Instead, the output of Discriminator when a batch of fake data is put into it would be upload to the server for the training of the global Generator model.

Server Generation The global Generator model will produce a batch of fake data, denoted as \mathcal{D}_g , based on a random sequence of data label and noise input. Those generated data will be sent to every client in a batch for their local training process.

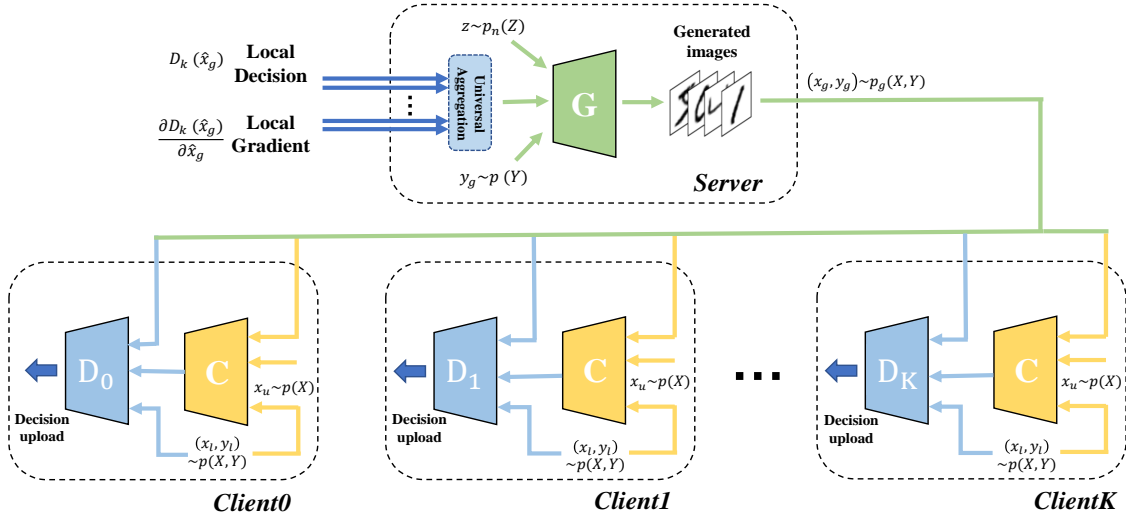


Figure 3: Overall framework of FedtriGAN-CE. Networks of D, C and G are colored in blue, yellow and green respectively. The output of the local discriminator models is uploaded to the server for the adversarial training of global generator model.

4.2.1 Local Training Process

Similar to the regular version, in FedtriGAN-CE, the Classifier model in the clients will still obey the uploading and server aggregation strategy for its update. Simultaneously, the training strategy is counted and controlled by the communication rounds with two thresholds for adding the loss regularization related to the output of the local discriminator model and global generator model.

The Discriminator model in the client will utilize both the labeled data and the generated data to enhance its ability to distinguish the real and fake data, which in return will finally improve the performance of the Classifier and the Generator. The loss function of the classifier will stay nearly constant with only changing the generated data distribution symbol from the local one to

the global one. Moreover, the $k - th$ local discriminator model has the training target function as:

$$\max_{D_k} U(G, D_k) = \mathbb{E}_{x \sim p_k(x)} [\log D_k(x)] + \mathbb{E}_{z \sim \mathcal{N}(0, I_d)} [\log (1 - D_k(G(z)))] \quad (12)$$

4.2.2 Server Training Process

At the end of the local training process, the clients who participated in the global communication of this round will upload the corresponding gradients and output or decision value from their discriminator model under the input of generated data batch. The uploaded output of the local Discriminator model will be synthesized based on the data the corresponding client owns and a transform of odds values. Thus, the global Generator model can obtain a global view for the total data distribution of the whole federated system. When the generation is mature and the generated data is well enough, the fake data will benefit the classifier.

As for the training of global generator model, the Universal Aggregation[32] should be introduced firstly. We define the Odds Value as:

$$\Phi(\phi) \triangleq \frac{\phi}{1 - \phi} \quad (13)$$

where $\phi \in (0, 1)$ which is same with the output decision of discriminator model. Further, it can be gotten the simulated global discriminator output:

$$\Phi(D_G(x_{gt}^i)) = \sum_{k=1}^K \alpha_k \Phi(D_k(x_{gt}^i)), \quad x_{gt}^i \in \mathcal{D}_{gt} \quad (14)$$

Here, \mathcal{D}_{gt} is the $t - th$ round generated data batch, and x_{gt}^i is the $i - th$ generated image data of the batch, where $i = 1, \dots, m$. Based on above equation, the universal aggregated discriminator output can be calculated as:

$$\hat{D}_G(x_{gt}^i) = \frac{\Phi(D_G(x_{gt}^i))}{1 + \Phi(D_G(x_{gt}^i))} \quad (15)$$

Thus, a series of transformation has aggregate the decision output from the client discriminator model into a simulated global one. This result could be used to get involved in the training of generator model in the server. The training loss function of the generator is:

$$\min_G U(G, \hat{D}_G) = \mathbb{E}_{x \sim p(x)} [\log \hat{D}_G(x)] + \mathbb{E}_{z \sim \mathcal{N}(0, I_d)} [\log (1 - \hat{D}_G(G(z)))] \quad (16)$$

When the server training process for the global generator model is finished, the server will generate a group of new fake images as the next-round training information for the clients to download.

Algorithm 2 FedtriGAN-CE

Require: \mathcal{D}_L and \mathcal{D}_U **Require:** Batch size: m

```
1: Initialization:  $\omega_{G0}, \omega_{D0}^k, \omega_{C0}^k$  ▷ initialize weights
2: for each communication round  $t = 1, \dots, T_1$  do
3:
4:   procedure SERVER GENERATION
5:     G generates synthetic data:  $\hat{x}_i = G(z_i)$  for  $i = 1, \dots, m$ 
6:     Delegate generated batch data to all the client:  $\mathcal{D}_{gt} = \{\hat{x}_1, \dots, \hat{x}_m\}$ 
7:   end procedure
8:
9:    $\mathcal{L}_t = \{l^k\}_{k=1}^{N_t} \leftarrow \mathcal{L} = \{l^k\}_{k=1}^N$  ▷ random selection of clients
10:  for each client  $k \in \mathcal{L}_t$  in parallel do
11:     $\Delta\omega_{NET(t+1)}^k, l_{NETt}^k \leftarrow \text{Local Update-CE}(\omega_{NETt}^k, \mathcal{D}_L^k, \mathcal{D}_U^k, \mathcal{D}_{gt}, t)$ 
12:    where  $NET \in \{D, C\}$ 
13:  end for
14:   $\omega_{NET(t+1)}^G \leftarrow \omega_{NETt}^G + \frac{1}{N_t} (\sum_{i=1}^{N_t} \Delta\omega_{NET(t+1)}^k)$  ▷ server aggregation
15:
16:  procedure SERVER TRAINING
17:     $\Phi(D_G(\mathcal{D}_{gt})) = \sum_{k=1}^K \alpha_k \Phi(D_k(\mathcal{D}_{gt}))$  ▷ global Odds Value computation
18:     $\hat{D}_G(\mathcal{D}_{gt}) = \frac{\Phi(D_G(\mathcal{D}_{gt}))}{1 + \Phi(D_G(\mathcal{D}_{gt}))}$  ▷ Universal Aggregation
19:     $l_{Gt} \leftarrow \mathcal{L}_G = \mathbb{E}_{x \sim p(x)} [\log D_{ua}(x)] + \mathbb{E}_{z \sim \mathcal{N}(0, I_d)} [\log (1 - D_{ua}(G(z)))]$ 
20:     $\omega_{G(t+1)} \leftarrow \omega_{Gt} - \eta \nabla \mathcal{L}_{Gt}$ 
21:  end procedure
22: end for
23:
24: function LOCAL UPDATE-CE( $\omega_{Dt}^k, \omega_{Ct}^k, \mathcal{D}_L^k, \mathcal{D}_U^k, t$ )
25:   if  $t \geq T_2$  then ▷ Training Phase Judgement
26:      $L_C = \alpha \sum_{(x_u, y_u)} p_c(y_u | x_u) \log(1 - D(x_c, y_c)) + L_l + \alpha_{\mathcal{D}} L_p$ 
27:   else if  $t > T_1$  then
28:      $L_C = \alpha \sum_{(x_c, y_c)} p_c(y_c | x_c) \log(1 - D(x_c, y_c)) + E_{p_l}(\log C(f(\mathbf{x}_i^k; \omega^k) y_i^k))$ 
29:   else
30:      $L_C = E_{p_l}(\log C(f(\mathbf{x}_i^k; \omega^k) y_i^k))$ 
31:   end if
32:   for  $i$  in local epochs do
33:     for  $B_1, B_2$  in  $\mathcal{D}_L^k, \mathcal{D}_U^k$  do
34:        $l_{NETt}^k \leftarrow L_{NET}(\omega_{NETt}^k, B_1, B_2)$  ▷ pseudo labeled loss computation
35:        $\omega_{NET(t+1)}^k \leftarrow \omega_{NETt}^k - \eta \nabla \mathcal{L}_{NETt}^k$  ▷ mini batch gradient descent
36:     end for
37:   end for
38:   return  $\omega_{D(t+1)}^k, \omega_{C(t+1)}^k, l_{Dt}^k, l_{Ct}^k$  ▷ return updated weights and client loss
39: end function
```

5 Experiment

5.1 Experimental Setting

5.1.1 Datasets

In our experiments, we use four public available datasets, including MNIST[33], Fashion-MNIST[34] and SVHN[35]. Both MNIST and Fashion-MNIST dataset are divided into a training set of 60,000 iamges and a test set of 10,000 images. There are 73,257 digits used for training and 26,032 digits for testing in the SVHN dataset. These three datasets are all used for the image classification task with 10 categories (i.e., $C = 10$).



Figure 4: Dataset samples of MNIST, SVHN and Fashion-MNIST

5.1.2 Data Distribution Setting

For three image datasets, each of them will be randomly shuffled and divided into 10 shards for different clients which will have $\frac{D}{N}$ samples separately, where D is the total number of training data and N is the number of clients. We use α to represent the labeled data ratio of all the clients. In other words, there are $\frac{D}{N} * \alpha$ labeled data in every client as well as $\frac{D}{N} * (1 - \alpha)$ unlabeled ones. To fully estimate the performance of our model, we use labeled data amount and classes contained in each client to control the data distribution of our federated framework, resulting in three distribution settings:

IID setting Both labeled and unlabeled data of the train set will be shuffled randomly then allocated to each client. Under this condition, every client will have nearly same data distribution either on labeled data amount or classes number.

Non-IID-I setting In this setting, after random shuffling, we divided the dataset in an extreme unbalanced distribtuion, where each client only contains two classes of data with a universal labeled data ratio α .

Non-IID-II setting We adapted different division strategy for labeled data and unlabeled data. For unlabeled data distribution, every client contains all categories of data, which is exactly same with the IID condition. While for labeled data distribution, similar to Non-IID-I setting, there are only two categories of them stored in each client with a labeled data ratio α to control label amount.

Non-IID-III setting Different client will have different ratios of labeled data. In our experiments, 1 client own 55% labeled data, and 9 clients only own 5% labeled data.

5.1.3 Baselines

. To fairly validate the proposed `FedtriGAN` framework, we use the following state-of-the-art baselines.

- *Federated supervised learning models:*
`FedAvg` [1].
- *Federated semi-supervised learning models:*
`FedSem` [4],
`FedMatch` [15],
`FedsemiGAN` FL adaptation of semi-supervised GANs [24],

5.1.4 Data Augmentations

. We perform weak data augmentation techniques on the three image datasets, including resize, rotation and pixel value normalization. A randomly selected patch of the image is also resized to 32x32 with a random horizontal flip, accompanied by a color distortion consisting of a random sequence of brightness, contrast, saturation, hue, and optional grayscale conversion. The patch is then given a Gaussian blur and solarization effect.

In the following subsections, we will compare the performance of our model `FedtriGAN` with the above baselines under four different data distribution settings.

5.2 Implementation

In all baselines and `FedtriGAN`, we use the same local model for each client. Two Convolutional Neural Network is used for the image classification tasks of three datasets. The first one consists of two convolutional layers and two linear layers for the classification of MNIST and Fashion-MNIST. While for SVHN, we adopt another network with a more complex structure, including a convolutional block and a full-connection block. The former consists of 6 convolutional layers and related batch normalization layers, max-pooling, and ReLU operation to improve the model’s generalization ability.

Wasserstein GAN with Gradient Penalty (WGAN-GP)[36] architecture is employed, which is proved to be more stable during the training process and to provide similar or better results than other GAN architectures.

We adopt the weak data argumentation technique on the three datasets for all the baselines and `FedtriGAN`, where the main process contains random reflect, flip, contrast adjustment, grayscale, and crop.

For all the IID experiments, we set the local training epochs as $R_l = 3$, and communication rounds as $R_g = 300$. For the Non-IID setting, the local training epoch is set to 1 with other parameters constant. Besides, the client number is fixed as 10, the local training batch size for labeled data is 50, for unlabeled data is 100.

5.3 Performance Evaluation for the IID Setting

Table 1 shows that under IID setting, `FedtriGAN` outperforms all the baselines on different labeled data ratio, where the total labeled data number are 60,600 and 6000 for MNIST, 600, 1000 and 3000 for Fashion-MNIST, 1000, 3000 and 6000 for SVHN.

Dataset	MNIST			Fashion-MNIST			SVHN		
Labeled data number	60	600	6000	600	1000	6000	1000	3000	6000
FedAvg	74.45%	92.75%	97.25%	77.69%	80.26%	86.91%	71.67%	84.37%	88.50%
Fedsem	70.98%	94.04%	98.43%	74.04%	78.44%	87.37%	68.36%	75.65%	89.14%
FedsemiGAN	71.77%	93.58%	97.25%	72.56%	76.67%	87.67%	67.38%	77.62%	78.32%
FedMatch	74.94%	94.37%	98.11%	79.64%	85.78%	87.81%	76.47%	87.21%	91.04%
FedtriGAN	80.67%	94.98%	99.26%	82.26%	87.66%	89.95%	80.88%	90.60%	92.25%
FedtriGAN-CE	77.31%	94.11%	98.52%	80.47%	84.42%	87.39%	77.02%	86.95%	90.37%

Table 1: Accuracy on the three datasets under the IID setting, where all clients have the same distribution.

Here FedAvg is a federated supervised learning model. FedSem, FedsemiGAN and FedMatch are three baseline models of federated semi-supervised learning. From the accuracy value of the table, it can be observed that our method FedtriGAN outperforms all the baseline models. It even obtains an accuracy above the supervised model for nearly 4 percent on SVHN with 6000 labeled data. FedsemiGAN uses discriminator network to play the role of classifier model, resulting in its instability when the labeled data is extremely scarce or the image data is complex. It performs badly especially in SVHN dataset, where it has a 10% lower gap with the supervised model performance. Thus, its results are not always better than the baseline model. FedMatch presents robust and efficient performance which does not drop significantly when the labeled data number is extremely small. It also has the highest classification accuracy in baseline models.

FedSem utilizes a two-phase process to perform pseudo-labeling on the unlabeled data. This makes it heavily depend on the first phase, supervised learning stage, to get an effective pre-trained model. While the pseudo-labeled data may be denoted with wrong labels, which may introduce harmful information for model training, resulting in a sensitivity for labeled data amount and quality. Thus, we can see that it outperforms the supervised learning method when the labeled data number is comparatively high, while it performs poorly when the number drops. For instance, on the MNIST dataset, it has better classification accuracy than the supervised model when the labeled data number is greater than or equal to 600, while it gets the inverse result when the number reduces to 60.

As for the communication efficient version of FedtriGAN, aka FedtriGAN-CE, its performance drops compared to the original method, while it still outperforms the FedAvg among all the IID settings. It surpasses the FedMatch for three datasets with minimum labeled data numbers, achieving 77.31% accuracy with only 60 labeled MNIST image data.

5.4 Performance Evaluation for the Non-IID Setting

From Table 2, we can observe that our proposed approach FedtriGAN still outperforms all the baselines. Compared with the results listed in Table 1, we find that all the accuracy drops in all the three Non-IID result tables. This observation is in accord with the fact that the Non-IID setting is more challenging than the IID setting for federated learning.

In Table 2, it can be seen that the classification accuracy of all the methods on Fashion-MNIST, SVHN and the small labeled data setting of MNIST has a significant decrease up to 12% compared with the result in Table 1. While the proposed method FedtriGAN obtained a result of 97.84% for 6000 MNIST labeled data which has only a drop by 2% compared to the IID condition.

Dataset	MNIST			Fashion-MNIST			SVHN		
Labeled data number	60	600	6000	600	1000	6000	1000	3000	6000
FedAvg	65.22%	88.81%	96.98%	72.17%	74.27%	76.79%	65.44%	70.01%	78.26%
Fedsem	62.68%	81.03%	94.27%	60.48%	76.57%	73.14%	65.25%	66.74%	78.81%
FedsemiGAN	57.46%	80.82%	92.41%	61.50%	64.26%	69.37%	54.51%	66.22%	76.29%
FedMatch	72.11%	90.34%	97.30%	72.46%	74.36%	76.47%	66.87%	73.46%	78.61%
FedtriGAN	72.16%	93.51%	97.84%	72.98%	75.45%	77.68%	68.73%	74.58%	79.33%
FedtriGAN-CE	70.48%	90.11%	96.12%	71.97%	74.28%	76.49%	66.02%	72.17%	78.31%

Table 2: Accuracy on the three datasets under the Non-IID-I setting

Dataset	MNIST			Fashion-MNIST			SVHN		
Labeled data number	60	600	6000	600	1000	6000	1000	3000	6000
FedAvg	74.48%	88.69%	95.78%	71.27%	71.60%	74.26%	64.78%	74.24%	80.56%
Fedsem	73.69%	84.20%	95.02%	65.04%	65.71%	69.89%	64.28%	66.02%	73.26%
FedsemiGAN	76.45%	85.74%	95.28%	66.56%	68.57%	71.24%	62.47%	69.89%	75.28%
FedMatch	78.67%	88.24%	96.34%	70.34%	72.27%	73.47%	67.56%	71.78%	76.17%
FedtriGAN	84.37%	89.81%	97.56%	72.56%	74.37%	75.31%	71.25%	76.59%	83.20%
FedtriGAN-CE	81.44%	88.79%	96.01%	71.32%	72.15%	72.37%	66.97%	72.31%	81.04%

Table 3: Accuracy on the three datasets under the Non-IID-II setting

In all the three tables 2 3 4, Fedsem shows great decrease of classification accuracy on all the three datasets under Non-IID setting. For FedsemiGAN, it has the worst performance among all the baseline methods, especially on Fashion-MNIST.

FedtriGAN performs better on the setting where the given labeled data is scarce. It achieves the accuracy of 72.16%, 72.98% and 68.73% separately for 60 labeled MNIST data, 600 labeled Fashion-MNIST data, and 1000 labeled SVHN data. FedtriGAN-CE shows unsatisfying performance on Fashion-MNIST, which is 71.97% on 600 labeled data setting (74.28% and 76.49% for 1000 and 6000 number settings) slightly lower than the supervised learning result, while competing on the FedSSL baseline models.

Table 4 tells that the performance of most approaches in the Non-IID-III setting is better than those in the Non-IID-I and Non-IID-II setting, as the Non-IID-III setting has the slightest unbalance of data distribution. We can see that the missing part of the data categories can critically influence the generalization ability of the models.

FedtriGAN still has the best performance, which is 91.02% on SVHN setting, 98.34% for MNIST and 89.88% for Fashion-MNIST with 6000 labeled data. FedMatch has the best overall

Dataset	MNIST			Fashion-MNIST			SVHN		
Labeled data number	60	600	6000	600	1000	6000	1000	3000	6000
FedAvg	69.14%	89.46%	97.31%	68.79%	80.32%	86.46%	76.47%	78.98%	82.62%
Fedsem	70.33%	86.37%	96.56%	66.45%	76.00%	84.42%	69.48%	80.47%	84.08%
FedsemiGAN	70.76%	85.34%	98.34%	68.92%	73.48%	85.34%	70.26%	81.33%	86.01%
FedMatch	75.22%	91.27%	98.13%	69.45%	81.56%	88.08%	81.23%	86.78%	90.47%
FedtriGAN	77.24%	96.28%	98.34%	71.26%	83.25%	89.88%	83.46%	88.45%	91.02%
FedtriGAN-CE	74.28%	90.82%	98.00%	70.16%	81.43%	87.87%	81.34%	85.45%	90.36%

Table 4: Accuracy on the three datasets under the Non-IID-III setting

result among all the baseline models in this setting especially on SVHN dataset, where its accuracy is higher than the second one by *4to9%*. FedsemiGAN shows similar stability with the IID condition. It even obtains the same accuracy with FedtriGAN. As for FedtriGAN-CE, it surpasses the supervised model on all the settings while slightly lower than FedMatch.

Based on these results, it can get the conclusion that the proposed method FedtriGAN has more advantages than all the baseline models on both IID and Non-IID performance.

6 Discussion

From the above evaluation sections, it can be seen that the proposed method `FedtriGAN` achieves the best performance among all the testing models under both IID and Non-IID settings. This shows the robustness of it to resist the challenge of statistics discrepancy and heterogeneity.

Besides, when the labeled data number drops to a comparatively low level, it maintains a stable performance without collapse, similar to `Fedsem` and `FedsemiGAN`. This is because the GANs structure provides a steady stream of unsupervised information for the classifier and fully captures and utilizes unlabeled data. The well-trained GANs will help the classifier get feedback on its prediction in regularization terms for loss computing. The generator model could also produce extra fake image data to enhance the generalization ability of the classifier model.

Considering the communication efficiency of `FedtriGAN`, the local generator models are replaced by a global server generator to form a new framework, namely `FedtriGAN-CE`. Based on the experiment results, we can see that the `FedtriGAN-CE` has surpassed the supervised baseline methods and most of the semi-supervised learning baseline under most settings. It significantly reduced the communication loss by avoiding uploading two deep networks' parameters to the server with the cost of a slight drop in the performance. Although, new communication content, generated data, is introduced to the communication loop while based on the relationship of data size and parameter numbers of the networks. Intuitively, this cost is still far below the previous version.

The performance drop problems could be caused by the divergence between the overall optimization and local optimal points. In `FedtriGAN`, the local generator and discriminator has simultaneous training paces, ensuring the consistency of local optimization on each client. While for `FedtriGAN-CE`, the divergence between the global optimization required by the server generator model and the sum of local optimization for all the clients.

The conclusion and future direction will be discussed in the next section.

7 Conclusion

In this paper, we aim to solve the challenge caused by the scarcity and limitation of labeled data in a federated learning setting, which is federated semi-supervised learning. An efficient and robust framework is proposed named FedtriGAN, which achieves state-of-art performance in both iid and three Non-iid data distribution scenarios. With the consideration of the communication efficiency, we further developed a novel update schema that significantly reduces the uploaded data amount. A series of further experiments show our methods achieve state-of-art performance on several benchmark image datasets.

7.1 Suggestions for further work

The well-labeled and high quality of the data is still a key block for the further improvement of the FL model performance. Thus, more powerful and communication efficient FedSSL is a hot spot for the next direction of FL research. Here, our proposed method FedtriGAN has obtained the state-of-art performance compared with the recent FedSL and FedSSL work. While the communication cost is still in need of reducing, FedtriGAN-CE has dramatically reduced the cost caused by the transmission of parameters from Generator and Discriminator Networks introducing new cost via delegating generated images. Another possible direction for future research is to improve the training stability of the federated learning model, especially for the condition when the labeled data is scarce and limited categories. As for the application side, this combination method between FL and GANs has the potential to provide data supplement for solving the unbalance on data amount and quality among different clients.

References

- [1] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, pages 1273–1282. PMLR, 2017.
- [2] Qimai Li, Zhichao Han, and Xiao-Ming Wu. Deeper insights into graph convolutional networks for semi-supervised learning. *AAAI*, 2018.
- [3] Mingqing Chen, Rajiv Mathews, Tom Ouyang, and Françoise Beaufays. Federated learning of out-of-vocabulary words. *arXiv preprint arXiv:1903.10635*, 2019.
- [4] Abdullatif Albaseer, Bekir Sait Ciftler, Mohamed Abdallah, and Ala Al-Fuqaha. Exploiting unlabeled data in smart cities using federated learning. *2020 International Wireless Communications and Mobile Computing (IWCMC)*, 2020.
- [5] Chongxuan LI, Taufik Xu, Jun Zhu, and Bo Zhang. Triple generative adversarial nets. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30, pages 4088–4098. Curran Associates, Inc., 2017.
- [6] Yue Zhao, Meng Li, Liangzhen Lai, Naveen Suda, Damon Civin, and Vikas Chandra. Federated learning with non-iid data. *arXiv preprint arXiv:1806.00582*, 2018.
- [7] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *MLSys 2020*, 2018.
- [8] Li Huang, Yifeng Yin, Zeng Fu, Shifa Zhang, Hao Deng, and Dianbo Liu. Load-boost: Loss-based adaboost federated machine learning on medical data. *arXiv preprint: 1811.12629*, 2018.
- [9] Sebastian Caldas, Jakub Konečný, H Brendan McMahan, and Ameet Talwalkar. Expanding the reach of federated learning by reducing client resource requirements. *arXiv preprint arXiv:1812.07210*, 2018.
- [10] WANG Luping, WANG Wei, and LI Bo. Cmfl: Mitigating communication overhead for federated learning. In *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, pages 954–964. IEEE, 2019.
- [11] Fei Chen, Mi Luo, Zhenhua Dong, Zhenguo Li, and Xiuqiang He. Federated meta-learning with fast convergence and efficient communication. *arXiv preprint arXiv:1802.07876*, 2018.
- [12] Abhishek Bhowmick, John Duchi, Julien Freudiger, Gaurav Kapoor, and Ryan Rogers. Protection against reconstruction and its applications in private federated learning. *arXiv preprint arXiv:1812.00984*, 2018.
- [13] Robin C Geyer, Tassilo Klein, and Moin Nabi. Differentially private federated learning: A client level perspective. *arXiv preprint arXiv:1712.07557*, 2017.
- [14] Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. Practical secure aggregation for federated learning on user-held data. *arXiv preprint arXiv:1611.04482*, 2016.

- [15] Wonyong Jeong, Jaehong Yoon, Eunho Yang, and Sung Ju Hwang. Federated semi-supervised learning with inter-client consistency. *In ICML Workshop*, 2020.
- [16] T. C. Chiu, Y. Y. Shih, A. C. Pang, C. S. Wang, W. Weng, and C. T. Chou. Semisupervised distributed learning with non-iid data for aiot service platform. *IEEE Internet of Things Journal*, 7(10):9266–9277, 2020.
- [17] Olivier Chapelle, Bernhard Scholkopf, and Alexander Zien. Semi-supervised learning (chapelle, o. et al., eds.; 2006)[book reviews]. *IEEE Transactions on Neural Networks*, 20(3):542–542, 2009.
- [18] Dong-Hyun Lee. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML*, volume 3, 2013.
- [19] Antti Rasmus, Mathias Berglund, Mikko Honkala, Harri Valpola, and Tapani Raiko. Semi-supervised learning with ladder networks. In *Advances in neural information processing systems*, pages 3546–3554, 2015.
- [20] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Advances in neural information processing systems*, pages 1195–1204, 2017.
- [21] Samuli Laine and Timo Aila. Temporal ensembling for semi-supervised learning. *In ICLR, arXiv:1610.02242*, 2017.
- [22] Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE transactions on pattern analysis and machine intelligence*, 41(8):1979–1993, 2018.
- [23] Sungrae Park, Jun-Keon Park, Su-Jin Shin, and Il-Chul Moon. Adversarial dropout for supervised and semi-supervised learning. *AAAI*, 2018.
- [24] Augustus Odena. Semi-supervised learning with generative adversarial networks, 2016.
- [25] Zihang Dai, Zhilin Yang, Fan Yang, William W Cohen, and Russ R Salakhutdinov. Good semi-supervised learning that requires a bad gan. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30, pages 6510–6520. Curran Associates, Inc., 2017.
- [26] Ben Athiwaratkun, Marc Finzi, Pavel Izmailov, and Andrew Gordon Wilson. There are many consistent explanations of unlabeled data: Why you should average. *ICLR*, 2019.
- [27] Qizhe Xie, Zihang Dai, Eduard Hovy, Minh-Thang Luong, and Quoc V Le. Unsupervised data augmentation for consistency training. *arXiv preprint arXiv:1904.12848*, 2019.
- [28] David Berthelot, Nicholas Carlini, Ekin D Cubuk, Alex Kurakin, Kihyuk Sohn, Han Zhang, and Colin Raffel. Remixmatch: Semi-supervised learning with distribution matching and augmentation anchoring. In *ICLR*, 2019.
- [29] Kihyuk Sohn, David Berthelot, Chun-Liang Li, Zizhao Zhang, Nicholas Carlini, Ekin D Cubuk, Alex Kurakin, Han Zhang, and Colin Raffel. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *arXiv preprint: 2001.07685*, 2020.

- [30] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014.
- [31] Chongxuan Li, Kun Xu, Jun Zhu, and Bo Zhang. Triple generative adversarial nets, 2017.
- [32] Yikai Zhang, Hui Qu, Qi Chang, Huidong Liu, Dimitris Metaxas, and Chao Chen. Training federated gans with theoretical guarantees: A universal aggregation approach, 2021.
- [33] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2323, 1998. Copyright: Copyright 2012 Elsevier B.V., All rights reserved.
- [34] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017.
- [35] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*, 2011.
- [36] Ishaan Gulrajani, Faruk Ahmed, Martín Arjovsky, Vincent Dumoulin, and Aaron C. Courville. Improved training of wasserstein gans. *CoRR*, abs/1704.00028, 2017.