

FedTriNet: A Pseudo Labeling Method with Three Players for Federated Semi-supervised Learning

Liwei Che
College of IST
Pennsylvania State University
State College, USA
lwche@psu.edu

Zewei Long
Department of Computer Science
University of Illinois Urbana-Champaign
Champaign, USA
zeweil2@illinois.edu

Jiaqi Wang
College of IST
Pennsylvania State University
State College, USA
jqwang@psu.edu

Yaqing Wang
School of Electrical and Computer Engineering
Purdue University
West Lafayette, USA
wang5075@purdue.edu

Houping Xiao
Institute for Insight
Georgia State University
Atlanta, USA
hxiao@gsu.edu

Fenglong Ma
College of IST
Pennsylvania State University
State College, USA
fenglong@psu.edu

Abstract—Federated Learning has shown great potentials for the distributed data utilization and privacy protection. Most existing federated learning approaches focus on the supervised setting, which means all the data stored in each client has labels. However, in real-world applications, the client data are impossible to be fully labeled. Thus, how to exploit the unlabeled data should be a new challenge for federated learning. Although a few studies are attempting to overcome this challenge, they may suffer from information leakage or misleading information usage problems. To tackle these issues, in this paper, we propose a novel federated semi-supervised learning method named FedTriNet, which consists of two learning phases. In the first phase, we pre-train FedTriNet using labeled data with FedAvg. In the second phase, we aim to make most of the unlabeled data to help model learning. In particular, we propose to use three networks and a dynamic quality control mechanism to generate high-quality pseudo labels for unlabeled data, which are added to the training set. Finally, FedTriNet uses the new training set to retrain the model. Experimental results on three publicly available datasets show that the proposed FedTriNet outperforms state-of-the-art baselines under both IID and Non-IID settings.

Index Terms—federated learning, semi-supervised learning, pseudo labeling

I. INTRODUCTION

Federated learning [1]–[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 collaborate on learning a global model, which outperforms any local models. This distributed framework contributes a series of advantages to the protection of data privacy, access rights, and security.

However, several practical issues still shackle federated learning aggregation and affect its performance. For instance, the clients tend to generate a large amount of data, but they lack labels or only contain a few labels. While existing federated methods such as FedAvg [1] mainly focus on the supervised scenario where client data are fully labeled. It is

crucial to get full access to the information included inside the unlabeled data to improve the global model performance.

Only a few studies are considering the unlabeled data, such as FedMatch [4] and FedSem [5]. FedMatch [4] introduces the inter-client consistency loss and additive parameter decomposition to disjointly learn on both labeled and unlabeled data. However, as this approach needs to collect information from neighboring clients, it may leak sensitive information. FedSem [5] uses a simple two-phase pseudo-labeling based method for semi-supervised learning applications. If the performance of the first phase, i.e., pretraining the model with labeled data, is poor, it would introduce error messages in the subsequent marking process of pseudo labels, which seriously affects the learning effect.

To address those problems, in this paper, we propose a novel two-phase learning framework named FedTriNet to guarantee information privacy and automatically generate high-quality pseudo labels via three networks. In the first phase, we pretrain the framework using labeled data on each client with FedAvg [1] like FedSem [5]. In the second phase, we aim to generate high-quality pseudo labels for unlabeled data and further use them for retraining each local model. Towards this end, we design a new approach by considering three client networks and automatically generating a threshold as the criteria to filter out low-quality pseudo labels in each client.

Designing of Three Players. In particular, the first network is the client model trained with labeled data in each client, which has good classification ability. In a deep neural network, such ability is usually determined by the last few layers. The second one is the global model aggregated by all the client models, which usually has a strong ability to extract features using the first few layers. The third model combines the client model and the global model, which tries to unify both models' advantages by borrowing the first few layers from the global model and the last few layers from the client model. Then, the combined model conducts finetuning with the labeled client

data.

Pseudo Label Generation. In each client, FedTriNet runs three networks on unlabeled data to output three prediction probability vectors, which are further used to generate the pseudo labels for unlabeled data. To guarantee the quality of the pseudo labels, we design a dynamical control mechanism to generate a global-level threshold θ . Remarkably, each client will identify the maximum probability value and then upload it to the server. The server will average the uploaded client-level maximum probability values and distribute the maximum value, i.e., θ , to each client. To carefully add the pseudo labeled data to the training set, a dynamic control mechanism is designed to make that θ decreases with the increase of the number of global training rounds. If the maximum probability value of the three prediction probability vectors is larger than θ , then the corresponding unlabeled data will be added to the training data.

Finally, FedTriNet will retrain the client model using both the real labeled data and the pseudo-labeled data. Note that since there are three models in each client, we choose to retrain the finetuned combined model, which is significantly different from FedAvg [1] and FedSem [5]. We evaluate the proposed FedTriNet on three benchmark image datasets under both IID and Non-IID data distribution settings compared with state-of-art baselines. Experimental results show the effectiveness of the proposed FedTriNet framework.

The remainder of this paper is organized as follows. Section II systematically reviews the recent related work. Section III introduces the details of the proposed FedTriNet. Section IV presents experimental setups, results and analysis compared with baselines. Section V concludes.

II. RELATED WORK

This section systematically reviews the studies on federated learning, federated semi-supervised learning, and semi-supervised learning.

A. Federated Supervised Learning

Federated learning provides an efficient and privacy-preserved collaboration strategy for mutually training between different data owners, such as distributed data centers, customers and diverse institutions. The majority of federated learning works focus more on supervised learning scenarios and solving three challenges: statistical heterogeneity [6]–[8], system constraints [9]–[11], and trustworthiness [12]–[14]. In particular, [6] uses a shared server-stored dataset to help the clients achieve higher performance in Non-IID settings; [8] applies adjustment on the SGD convergence of federated learning; and [7] adds 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.

B. Federated Semi-supervised Learning

A more realistic setting in federated learning is federated semi-supervised learning, i.e., simultaneously considering both labeled and unlabeled data. However, the introduction of

the unlabeled data will significantly increase the difficulty of the problem. The studies on federated semi-supervised learning are still at the baby step, but more and more researchers are paying attention to this research topic. In [5], the authors propose a simple two-phase pseudo-labeling based method for semi-supervised learning application, and in [4], the authors introduce the inter-client consistency loss and additive parameter decomposition to disjointly learn on both the labeled and unlabeled data. However, the existing methods are efficient while may violate the clients' privacy or have poor performance with scarce labeled data, which are severe disadvantages for a federated semi-supervised learning problem.

C. Semi-supervised Learning

Semi-supervised learning SSL 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 [15]. The previous SSL work shows a series of diverse and coherent solutions. An intuitive approach is pseudo labeling, which uses the pretrained model to label the unlabeled data. [16] introduces a dynamic decision threshold to help the model labeling the data. Another effective and well-known strategy is to add consistency regularization on the training loss [17]–[21]. [22] presents an SSL method based on three neural networks, which characterize the conditional distributions between images and labels. In [23], the authors suggest that the flat platform of SGD leads to the convergence dilemma of consistency-based SSL. UDA [24], ReMixMatch [25], and Fixmatch [26] mix plenty of practical methods and do further exploration. [27] adapts curriculum learning idea into pseudo label method with self-training strategy, especially for the setting with a small set of labeled data and a large set of unlabeled data.

III. FEDTRINET FRAMEWORK

The goal of federated semi-supervised learning is to learn a global model G via collaboratively training K local client models $\mathcal{L} = \{l^k\}_{k=1}^K$. In this paper, we focus on the following setting: Each client stores both labeled data $\mathcal{D}_L^k = \{(\mathbf{x}_i^k, y_i^k)\}_{i=1}^{N_L^k}$ and unlabeled data $\mathcal{D}_U^k = \{\mathbf{x}_j^k\}_{j=1}^{N_U^k}$, where $y_i^k \in \{1, \dots, M\}$ is the corresponding label of the data instance \mathbf{x}_i^k , N_L^k denotes the number of labeled data of the k -th client, and N_U^k denotes the number of unlabeled data of the k -th client. Note that there are no data at the server side. To learn the global model G , we design a simple yet effective framework named FedTriNet. Next, we present the details of our framework.

A. Model Overview

Figure 1 shows the flow of the proposed framework FedTriNet. FedTriNet consists of two modules, i.e., *local training* and *server update*. In the local training module, each client k trains a local model l^k using both labeled and unlabeled data. The parameters of l^k , which is denoted as ω^k , will be uploaded to the server. In the server update module, the

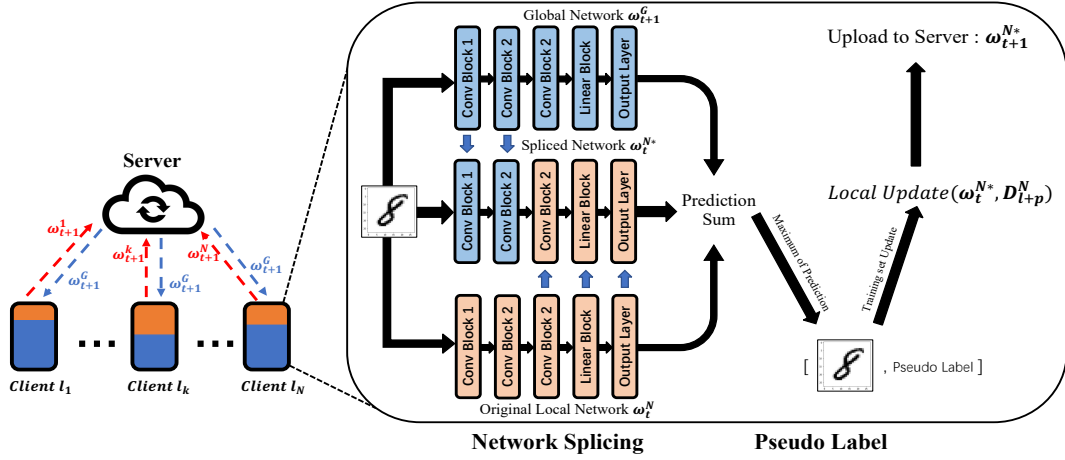


Fig. 1. The proposed FedTriNet Framework

server learns a global model G by aggregating K randomly uploaded local models, i.e.,

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

where ω^G is the parameters of G . ω^G will be then distributed to each local client. This procedure will be repeatedly executed until the global model G converges.

In particular, the local training module of the proposed FedTriNet has two stages, which are *pre-training* and *pseudo label learning*. The goal of the pre-training stage is to train each local model and global model T_1 rounds only using labeled data. Then in the pseudo label learning stage, FedTriNet generates a pseudo label for each unlabeled data using three networks, which are original local network, downloaded global network, and a spliced network separately.

The spliced network is a combination of the original local network and global network. Here, we assume that the global model's low-level feature extraction ability is better than that of the local model, which can be represented by the first n layers of G . However, the local model can capture the classification characteristics of local data, which can be described by the last m layers of ω^k . Thus, we can obtain a new network with $n + m$ layers to predict a pseudo label for each unlabeled sample.

By aggregating the outputs of the three networks, we can finally assign labels to unlabeled data. Using both labeled and pseudo labeled data, we can run a local training module to update the parameters, which will be uploaded to the server to update the global parameters. The new global parameters will also be distributed to each local client until they converge or the procedure runs T_2 rounds. The server update uses Eq. (1), and next, we will present the details of the local training in the proposed FedTriNet framework.

B. Pre-training Stage

The proposed FedTriNet framework aims to generate pseudo labels for unlabeled data and then to update the local

model using both labeled and pseudo labeled data. The critical issue of this approach is how to guarantee the quality of the generated pseudo labels. Towards this end, we propose to pre-train the local and global models only using labeled data by optimizing the following loss function as FedAvg [1]:

$$L_l(\mathcal{D}_L^k) = \min \left[\frac{1}{N_L^k} \sum_{i=1}^{N_L^k} CE(f(\mathbf{x}_i^k; \omega^k), y_i^k) \right], \quad (2)$$

where $L_l(\mathcal{D}_L^k)$ denotes the total loss, CE is the cross-entropy loss, $f(\cdot; \cdot)$ represents the neural network such as convolutional neural network (CNN), and ω^k is the parameter set. Then Eq. (1) is used to obtain the parameter set ω_t^G of the global model G .

In each communication round, the clients will download the global model's parameter for local training with predefined epochs from the server. After that, part of the clients will take part in the global aggregation that their local model parameters will be uploaded to the server. We repeatedly run this procedure T_1 times to pretrain both local and global models, and then FedTriNet starts to consider the unlabeled data..

C. Pseudo Label Learning Stage

To make fully use of unlabeled data, a straightforward approach is to generate pseudo labels based on the pre-trained model in the pre-training stage. However, there are two kinds of models for each client, i.e., a local model l^k and a global model G . The local model may perform better when the unlabeled data follow a similar distribution as the labeled data. However, real-world applications may not satisfy this constraint. The global model G is aggregated by several local models. Using G to generate the pseudo labels may not capture the characteristics of local models. Thus, either using local models or the global model may be prone to generate incorrect labels, further introducing incorrect information to model learning.

To guarantee the quality of pseudo labels as much as possible, in this paper, we introduce a combined model for each client, a combination of each local model and the global model. Intuitively, the shallow layers of deep neural networks focus more on low-dimensional feature learning, which can be shared even for different images. On the contrary, the class-related features of an image are abstracted into deeper layers, which are uniqueness. Based on this intuition, we can assemble a new network using the shallow layers' parameters of the global network that have better generalization ability and deep layers of the local network for capturing class-specific characteristics.

For instance, a convolutional neural network consists of three convolutional layers and two full connection layers. We usually select the parameters of the first two convolutional layers of the global network and the parameters of the full connection layers of the local network to form a new combined network. Note that the specific method of interception and the selection of layers are influenced by data type, network structures, and training parameter settings.

1) *Multi-view Pseudo-labeling*: In the pseudo-labeling process, the pseudo label of one unlabeled data is decided by a mutual output based on the sum of the prediction probabilities of three players. Different from the majority voting strategy, which uses one-hot coding to adapt the position with the highest vote identified as the category to which the input belongs, our method uses the outputs of the softmax layer, where the location of the output to which the input belongs is a probability value. This could avoid some statistical errors arisen in the decision process, such as three different votes, rounding errors.

Let $\mathbf{p}_{l^k}(\mathbf{x}_j^k)$ be the probability vector predicted by the local models l^k with parameters ω^k on the unlabeled data \mathbf{x}_j^k , and $\mathbf{p}_G(\mathbf{x}_j^k)$ be the probability vector outputted by the global model G . Let c^k denote the combined model and $\mathbf{p}_{c^k}(\mathbf{x}_j^k)$ be the outputted probability vector. Note that in our implementation, we use labeled data to fine-tune the model c^k first and then use it to make predictions. Thus, the pseudo label of the unlabeled data \mathbf{x}_j^k is

$$\begin{aligned} \mathbf{p}_j^k &= \frac{1}{3} [\mathbf{p}_{l^k}(\mathbf{x}_j^k) + \mathbf{p}_G(\mathbf{x}_j^k) + \mathbf{p}_{c^k}(\mathbf{x}_j^k)], \\ \hat{y}_j^k &= \arg \max \mathbf{p}_j^k. \end{aligned} \quad (3)$$

In order to use the pseudo labeled data to update the model, we must guarantee the quality of the pseudo labels. In other words, we cannot directly use all the pseudo labeled data and only use the data with high confidence. Thus, we design the following mechanism to control the quality of pseudo-labeled data dynamically. In particular, FedTriNet dynamically generates a global threshold θ . If the maximum probability of unlabeled data is greater than θ , then the corresponding data will be added to the training set. Next, we will how to estimate the value of θ .

2) *Dynamic Pseudo-labeled Data Selection*: Towards the goal of generating a global threshold θ , we first run the global model G on each unlabeled sample \mathbf{x}_j^k stored in each client

$k \in \{1, \dots, K\}$ to obtain the prediction $\mathbf{p}_G(\mathbf{x}_j^k)$. Then we can have the maximum probability of $\mathbf{p}_G(\mathbf{x}_j^k)$, i.e., $\max(\mathbf{p}_G(\mathbf{x}_j^k))$. Since there are N_U^k unlabeled data in client k , we can obtain N_U^k maximum probability values, i.e., $\{\max(\mathbf{p}_G(\mathbf{x}_j^k))\}_{j=1}^{N_U^k}$. Finally, the maximum predictive probability of all the unlabeled data is

$$\theta^k = \max\{\max(\mathbf{p}_G(\mathbf{x}_1^k)), \dots, \max(\mathbf{p}_G(\mathbf{x}_{N_U^k}^k))\}. \quad (4)$$

Since there are K clients, for each client, we can obtain a client-level threshold. These K thresholds are uploaded to the server to generate the global-level threshold θ as follows:

$$\theta(t) = \begin{cases} \alpha \bar{\theta} & t < 10, \\ \frac{(100-2t)}{100} \alpha \bar{\theta} & 10 \leq t < 35, \\ \frac{1}{2} \alpha \bar{\theta} & t \geq 35, \end{cases} \quad (5)$$

where t represents the number of communication rounds in the pseudo label learning stage, α is a predefined hyper-parameter to control the threshold, and $\bar{\theta}$ denotes the average of all the uploaded client-level thresholds, i.e., $\bar{\theta} = \frac{1}{K} \sum_{k=1}^K \theta^k$. The motivation behind Eq. (5) is that we want the local model to be more stable in the first few rounds of the pseudo label process. In order to avoid updating too many pseudo labeled data into the training set at one time, a larger threshold is used at the beginning of the pseudo label learning stage (i.e., $t < 10$) by setting $\alpha = 0.93$ (experimental result) in the experiment. In such a way, only a tiny amount of high-quality pseudo labeled data will be added to the training first. With the increase of the communication rounds, the threshold value will decrease. In other words, there will be more data to be added to the training set.

The global threshold $\theta(t)$ using Eq. (5) is then distributed to each client k . If $\max(\mathbf{p}_j^k)$ in Eq. (3) is greater than $\theta(t)$, then the corresponding unlabeled data will be added to the training set. Let \mathcal{D}_P^k denote the selected pseudo labeled data, which will be used to retrain the local model.

3) *Local Model Retraining & Server Aggregation*: FedTriNet is able to generate pseudo labels for unlabeled data and automatically add high-quality unlabeled data to the training set. Thus, based on the new training data $\{\mathcal{D}_L^k, \mathcal{D}_P^k\}$, we can retrain each local model by minimizing the following loss function:

$$L_{total} = L_l(\mathcal{D}_L^k) + \lambda L_p(\mathcal{D}_P^k), \quad (6)$$

where $L_l(\mathcal{D}_L^k)$ is the loss on the labeled data calculated by Eq. (2), λ is a hyperparameter to balance the loss obtained from the pseudo-labeled data, and $L_p(\mathcal{D}_P^k)$ is the loss of the pseudo-labeled data and defined as follows:

$$L_p(\mathcal{D}_P^k) = \min \left[\frac{1}{N_P^k} \sum_{j=1}^{N_P^k} CE(f(\mathbf{x}_j^k; \omega^k), \hat{y}_j^k) \right], \quad (7)$$

where N_P^k is the number of selected high-quality pseudo-labeled data, and \hat{y}_j^k is the pseudo label of the unlabeled data \mathbf{x}_j^k . We maintain the same uploading, model aggregation and downloading methods as in the pre-training stage to retrain

the model in the pseudo label learning stage until FedTriNet converges or runs T_2 times. However, the difference is that we train the combined network, i.e., c^k , at client side instead of the renewed global model G as FedAvg. The whole learning procedure is shown in Algorithm 1.

4) *Layers Selection for Model Splicing*: Obviously, the global model generally has the better generalization ability than the local models after aggregation. In contrast, the local models show better performance on their corresponding local datasets due to the difference among local trainsets. For a deep neural network, we can call the first few layers as shallow layers, and the counter-down few layers as deeper layers. The training of deep neural networks is often a process of extracting high-dimensional information from data. The deeper the network layer, the more abstract the information processed. Based on that we believe the shallow layers are tending to focus more on common features of a dataset, while deeper layers for more specific ones. Thus, the combination of the shallow layers of the global model and the deeper layers of local models could creat a stronger combined networks.

In our work, considering that the CNN network used has a relatively simple structure, we choose the first two convolutional networks as shallow layers, and the remaining network structure as deeper layers. Our method can replace the data set and network structure relatively easily. When faced with a complex network structure, in order to achieve the optimal training effect, further experiments are needed to find the optimal network splicing method. But in this work, our experiment results show that appropriate adjustments to shallow layers will not cause a huge difference in classification accuracy. Therefore, in the subsequent experimental sections, we will focus on the method itself instead of the selection of layers structure.

IV. EXPERIMENT

In this section, we first introduce the experimental settings, implementation, and then present the experimental results under both IID and Non-IID scenarios.

A. Experimental Settings

1) *Datasets*: In our experiments, we use three public datasets in our experiment: MNIST, Fashion-MNIST, and SVHN. For MNIST and Fashion-MNIST datasets, both of them are divided into a training set of 60,000 images and a test set of 10,000 images. For the SVHN dataset, 73,257 digits are used for training and 26,032 digits for testing. The three datasets are all used for the image classification task with 10 categories (i.e., $C = 10$).

2) *Data Distribution Setting*: Each of the three datasets is randomly shuffled and divided into 10 shares for N different clients. Given training data number D and labeled data proportion α , there is $D \times \alpha / N$ labeled data and $D \times (1 - \alpha) / N$ unlabeled data for each client. To estimate our model performance, we use labeled data amount and class categories in each client to control the data distribution. Furthermore, we consider non-IID and IID distribution settings respectively. For

Algorithm 1 FedTriNet

Require: D_L and D_U

```

1: procedure PHASE I (Pretrain)
2:   Initialization:  $\omega_0$  ▷ initialize weights
3:   for each communication round  $t = 1, 2, 3 \dots, T_1$  do
4:      $L_t = \{l^k\}_{k=1}^{N_t} \leftarrow \mathcal{L} = \{l^k\}_{k=1}^N$  ▷ random selection
     of clients for server aggregation
5:     for each client  $k \in L_t$  in parallel do
6:        $\Delta\omega_{t+1}^k, l_t^k \leftarrow$  Local Update I( $\omega_t^G, D_L^k$ ) ▷ local
       model training with labeled data
7:     end for
8:      $\omega_{t+1}^G \leftarrow \omega_t^G + \frac{1}{N_t} (\sum_{i=1}^{N_t} \Delta\omega_{t+1}^i)$  ▷ server
     aggregation by weights averaging
9:   end for
10: end procedure
11: procedure PHASE II (Pseudo Label Learning)
12:   for each communication round  $t = 1, 2, 3 \dots, T_2$  do
13:      $L_t = \{l^k\}_{k=1}^{N_t} \leftarrow \mathcal{L} = \{l^k\}_{k=1}^N$  ▷ random clients
     selection
14:     for each client  $k \in L_t$  in parallel do
15:        $\Delta\omega_{t+1}^k, l_t^k \leftarrow$  ←
       Local Update II( $\omega_t^G, \omega_{t-1}^G, D_L^k, D_U^k, t$ ) ▷ client k's local
       training and pseudo labeling
16:     end for
17:      $\omega_{t+1}^G \leftarrow \omega_t^G + \frac{1}{N_t} (\sum_{i=1}^{N_t} \Delta\omega_{t+1}^i)$  ▷ server
     aggregation by weights averaging
18:   end for
19: end procedure
20:
21: function LOCAL UPDATE I ( $\omega_t^G, D_L^k$ )
22:   for i in local epochs do
23:     for  $B_1$  in  $D_L^k$  do
24:        $l_t^k \leftarrow L_l(\omega_t^k, B_1)$  ▷ supervised loss
       computation
25:        $\omega_{t+1}^k \leftarrow \omega_t^k - \eta \nabla l_t^k$  ▷ mini batch gradient
       descent
26:     end for
27:   end for
28:   return  $\omega_{t+1}^k, l_t^k$  ▷ return updated weights and client
   loss
29: end function
30: function LOCAL UPDATE II ( $\omega_t^G, \omega_{t-1}^G, D_L^k, D_U^k, t$ )
31:    $\omega_t^k = \omega_t^G[0 : n] \cup \omega_{t-1}^k[n + 1 : m]$  ▷ model
   combination
32:    $\hat{Y}_j^k = \arg \max \mathbf{p}_j^k(D_U^k)$  ▷ joint prediction
33:   if  $p_j^k(D_U^k) > \theta(t)$  then ▷ compare joint prediction
   value with threshold
34:      $\hat{D}_U^k \leftarrow \hat{Y}_j^k$  ▷ pseudo labeling
35:   end if
36:   for i in local epochs do
37:     for  $B_1, B_2$  in  $D_L^k, \hat{D}_U^k$  do
38:        $l_t^k \leftarrow L_p(\omega_t^k, B_1, B_2)$  ▷ pseudo labeled loss
       computation
39:        $\omega_{t+1}^k \leftarrow \omega_t^k - \eta \nabla l_t^k$  ▷ mini batch gradient
       descent
40:     end for
41:   end for
42:   return  $\omega_{t+1}^k, l_t^k$  ▷ return updated weights and client
   loss
43: end function

```

the IID setting, both labeled and unlabeled data in the train set will be shuffled randomly and allocated to each client. For non-IID setting, every client owns all categories of unlabeled data and only two categories of labeled data.

3) *Baselines*: To fairly validate the proposed FedTriNet framework, we use one federated supervised learning model FedAvg [1], and two federated semi-supervised learning models, which are FedSem [5] and FedMatch [4].

- **FedAvg** [1], proposed by McMahan, et al., presents how to conduct federated learning of deep networks with decentralized data based on iterative model averaging under the communication cost constraints. Each client updates local models by stochastic gradient descent and the server performs model averaging. With empirical evaluation, this approach is robust to unbalanced and non-IID data distributions.
- **FedSem** [5], proposed by Abdullatif Albaseer, et al., combines pseudo labeling idea with federated semi-supervised learning problems in the smart city application. In this work, the training process is divided into two phases. In phase one, with the existing labeled data to supervise the training process, the local model obtains the certain classification ability. With the model, the local unlabeled data is labeled with the predicted value as a pseudo-label. In phase two, the whole federated framework will continue the same training process as in Phase I with data with real labels and pseudo-labels..
- **FedMatch** [4] 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 idea of consistency regularization is that the output of the predictor is expected to be as consistent as possible between an original sample and the processed version by data enhancement (the idea of consistency). In the process of server parameter delegation, in addition to the original model of the client, several models of other clients will be sent to the client as helper agents. The final purpose of local unsupervised training is to minimize the difference between the prediction results of the local model and the labels provided by each consensus model as small as possible.

In the following subsections, we will compare the performance of our model FedTriNet with the discussed baselines under two different data distribution settings.

B. Implementation

When implementing all baselines and FedTriNet, we use the same local model for each client. A Convolutional Neural Network (CNN) is used for the image classification tasks of three datasets. We adopt the weak data argumentation technique on the three datasets for all the baselines and FedTriNet, 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 5 and total communication rounds T as 100. For the non-IID setting,

the local training epoch is set to the same number as IID along with other parameters. For MNIST, the pre-training stage rounds T_1 is 40 and pseudo label learning stage rounds T_2 is 60; for Fashion-MNIST, T_1 is 30 and T_2 is 70; for SVHN, T_1 is 60 and T_2 is 40. Besides, the client number is fixed as 10. The local training batch size is set as 50 for both labeled data and unlabeled data.

C. Performance Evaluation for the IID Setting

Table I shows the performance of all the approaches under the IID scenario. From Table I, we can observe that FedTriNet shows the best performance with all the given settings on the three datasets. Besides, with the increase of the number of labeled data, the performance of all the approaches increases. Although Fedsem also uses two-phase training, it cannot even outperform the supervised method FedAvg. The reason is that after phase I training, Fedsem generates pseudo labels for all the unlabeled data, which are then used for phase II training. Since the quality of pseudo labels is pretty low when the number of labeled data is small, the misleading information further hurts the learning of Phase II. Thus, Fedsem performs worst compared with other baselines. FedMatch uses data augmentation, inter-client consistency, and disjoint learning techniques to achieve the second-best performance for all the settings. It has 46.75% accuracy on the MNIST dataset when the number of labeled data is set to 60 for all the clients, while FedAvg and Fedsem collapse. For the experiments on the SVHN dataset, we can also see that the FedMatch reaches 59.61% compared to the poor performance of both FedAvg and Fedsem.

D. Performance Evaluation for the Non-IID Setting

In Table II, with the Non-IID setting, our proposed approach FedTriNet still outperforms all the baselines. Especially for the experiment with 6000 labeled data, the accuracy of FedTriNet was 3.21% and 16.96% higher than that of FedAvg on MNIST and SVHN, respectively. Compared with the results listed in Table I, we find that all the accuracy drops. This observation is in accord with the fact, that is, the Non-IID setting is more challenging than the IID setting for federated learning due to the data and label imbalance.

It is worth mentioning that Fedsem shows extreme discomfort with Non-IID data, which has the greatest drop in performance among all the methods. This phenomenon further proves the fragility of the traditional self-training based pseudo-labeling method. FedTriNet with the three-player framework achieves the better ability for the heterogeneity challenge and even achieves higher classification accuracy for SVHN 3000 labeled data setting than the IID one.

E. Ablation Study

In this experiment, we aim to conduct the model insight analysis removing each of the following modules in FedTriNet, and the results are shown in Table III.

- **Threshold Guarantee Mechanism**. In the later stage of model training, the threshold of pseudo-labeling control

TABLE I
ACCURACY ON THE THREE DATASETS UNDER THE IID SETTING, WHERE ALL CLIENTS HAVE THE SAME DISTRIBUTION.

Dataset	MNIST			Fashion-MNIST			SVHN			
	# Labeled Data	60	600	6000	600	3000	6000	1000	3000	6000
FedAvg		29.26%	88.26%	96.46%	65.19%	74.54%	78.32%	27.82%	78.44%	87.77%
Fedsem		39.49%	84.54%	96.31%	45.33%	74.29%	78.78%	18.47%	76.22%	86.16%
FedMatch		46.75%	89.28%	97.14%	69.56%	77.28%	79.15%	59.61%	78.94%	88.26%
FedTriNet		77.25%	93.80%	97.56%	71.88%	78.01%	81.00%	63.99%	79.47%	89.48%

TABLE II
ACCURACY ON THE THREE DATASETS UNDER THE NON-IID SETTING.

Dataset	MNIST			Fashion-MNIST			SVHN			
	# Labeled Data	60	600	6000	600	3000	6000	1000	3000	6000
FedAvg		26.29%	77.67%	91.79%	63.86%	70.56%	74.28%	19.38%	46.70%	66.45%
Fedsem		32.06%	74.12%	84.11%	17.06%	56.67%	63.93%	18.95%	49.02%	53.94%
FedMatch		69.28%	79.15%	93.20%	65.44%	71.26%	74.81%	54.34%	74.27%	79.42%
FedTriNet		79.60%	82.55%	95.00%	69.50%	72.77%	75.05%	57.26%	82.78%	83.41%

is maintained at a relatively high value, which ensures that the pseudo-labeling data updated into the training set has consistently high quality and does not affect the model performance. The experiment results show that the shutdown of the Threshold Guarantee Mechanism will cause the performance drop by a few percent. The higher the original accuracy is, the less the drop is. This indicates that this mechanism can maintain the pseudo label quality to avoid introducing misleading information.

- *Fine-tuning.* In order to make the network parameters more suitable for local data, a new network constructed from the first several layers of the new global model and the several latter layers of the old local model will be labeled with fine-tuning operation with the labeled data. Another important reason for the fine-tuning process is that if the labeled data and unlabeled data are trained with the same model parameter respectively (that is, the model parameters are shared), the unlabeled training process may cause the model to forget the knowledge learned from the labeled data. To make a fair comparison, we compensate for additional local training epochs in the latter half of the communication round during the training process of the baseline models. From the results, we can see that the fine-tuning operation is essential for model learning.
- *Pseudo Labeling.* In the proposed FedTriNet, we first pre-train the model and then conduct the pseudo label learning. During the second phase, we remove the pseudo labeling operation and directly use the combined network c^k and the labeled data to train the model. We can observe that the performance of pseudo labeling significantly drops compared with that of FedTriNet. These results clearly demonstrate the importance of pseudo labeling for federated semi-supervised learning.

F. Phase Round Combination

For a constant total communication round setting, different phase I and phase II ratios may cause different model performances. Due to the different amounts of information in different images, a properly supervised learning training period would benefit the model performance more than an early entrance to the semi-supervised phase, a.k.a. pseudo label stage. Table IV demonstrates the performance changes with the different phase rounds under both IID and Non-IID settings. Here the first column represents the different phase round combinations. For example, 30 + 70 means the experiment is 30 phase I rounds and 70 phase II rounds.

Our IID experiments on MNIST, Fashion-MNIST, and SVHN show that under the parameter controlling condition, the model’s performance will increase as the supervised learning round increases till reaching a peak and then decrease. In our experiments, the most proper ratio for MNIST and Fashion-MNIST is 50 rounds in phase I and 50 rounds in phase II (50+50). For SVHN, the setting is 60+40 rounds. While for the non-IID setting, there is no apparent accuracy changing trends with the phase round combination for all three datasets. The best results of the three datasets MNIST, Fashion-MNIST, and SVHN are achieved with the settings of 50+50 rounds, 60+40 rounds, and 50+50 rounds, respectively. The overall results indicate the robustness of our algorithm for data distribution.

Fig 2 illustrates the loss and accuracy curves of phase round combination experiment for three datasets under IID setting. In loss curves, the start round of phase II usually causes a plummet, which is because the pseudo-labeled data is added into the training set. This results in the accuracy fluctuation within a narrow range, which usually happens to the curves whose finally performance is not satisfying, either. A proper phase round combination will allow the model to avoid introducing too many pseudo labeling errors into the training. For instance, in the accuracy curve of SVHN, the

TABLE III
ABLATION EXPERIMENT ON THE THREE DATASETS UNDER THE IID SETTING.

Dataset	MNIST	Fashion-MNIST	SVHN
# Labeled Data	600	1000	3000
FedTriNet	93.80%	78.07%	79.47%
-Threshold Guarantee Protection	90.22%	71.70%	74.93%
-Fine-tuning	87.20%	72.34%	71.30%
-Pseudo Labeling	86.50%	70.75%	72.73%

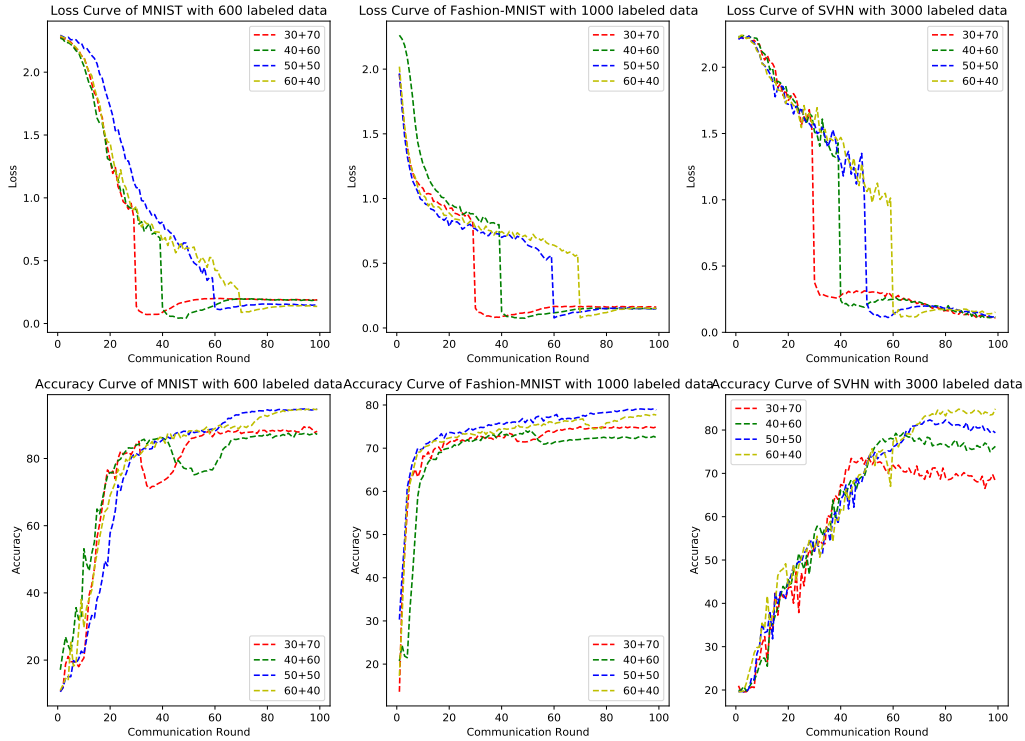


Fig. 2. Phase Round Combination Results under IID setting

60+40 case achieves best result, while the 30+70 one does not rise but fall. What noticeable else, is in phase II, the loss curves firstly increase then decrease, which reflects the correction function of our methods. Fig 3 shows the similar phenomenons under NonIID setting, with larger training curve fluctuation.

V. CONCLUSION

Federated learning is a new collaborative learning approach without sharing client data and can apply to many real-world applications. Although many federated learning approaches are proposed, they mainly focus on the supervised setting, which is not realistic due to the strict requirement that all the client data have corresponding labels. Only a few studies are trying to explore the power of unlabeled data, but they either need

to know the information of neighboring clients or introduce low-quality pseudo labels into the model training.

To address these problems, in this paper, we propose an effective pseudo labeling method with three players for federated semi-supervised learning called FedTriNet. FedTriNet consists of two learning phases. In the first phase, we use the labeled data to pre-train FedTriNet using FedAvg. In the second phase, we aim to use unlabeled data by generating high-quality pseudo labels. Towards this end, we propose to use three networks, including one local model, one global model, and one combined model from the previous two models. Besides, a quality control mechanism is proposed to generate a global-level threshold, which dynamically changes with the global training rounds. The corresponding unlabeled data can be added to the training set only when the max-

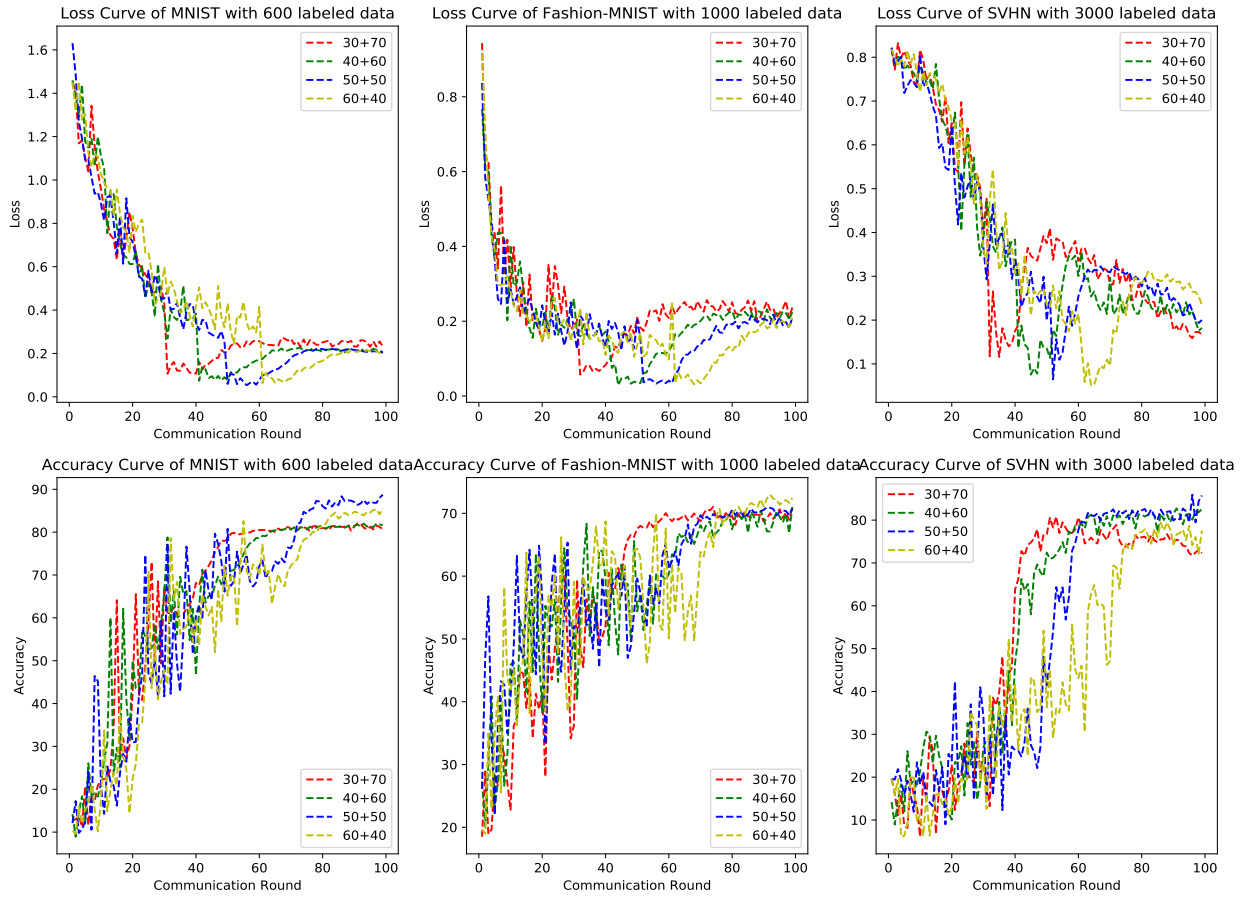


Fig. 3. Phase Round Combination Results under Non-IID setting

TABLE IV
ACCURACY ON THE THREE DATASETS UNDER DIFFERENT PHASE I AND PHASE II ROUNDS COMBINATION FOR IID AND NON-IID DATA.

Setting	IID			Non-IID		
Dataset	MNIST	Fashion-MNIST	SVHN	MNIST	Fashion-MNIST	SVHN
# Labeled Data	600	1000	3000	600	1000	3000
30+70	88.56%	74.79%	68.14%	80.94%	71.03%	72.31%
40+60	94.66%	75.84%	78.44%	80.65%	67.46%	82.36%
50+50	94.80%	79.05%	77.45%	87.06%	70.78%	84.57%
60+40	94.64%	77.67%	83.15%	85.51%	72.45%	77.57%

imum probability value is larger than this threshold. Finally, FedTriNet retrains the combined model with the new training data. We conduct experiments on three benchmark datasets to show the effectiveness of the proposed FedTriNet compared with state-of-the-art baselines.

REFERENCES

- [1] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial Intelligence and Statistics*. PMLR, 2017, pp. 1273–1282.
- [2] Q. Li, Z. Han, and X.-M. Wu, "Deeper insights into graph convolutional networks for semi-supervised learning," *AAAI*, 2018.
- [3] M. Chen, R. Mathews, T. Ouyang, and F. Beaufays, "Federated learning of out-of-vocabulary words," *arXiv preprint arXiv:1903.10635*, 2019.
- [4] W. Jeong, J. Yoon, E. Yang, and S. J. Hwang, "Federated semi-supervised learning with inter-client consistency," in *ICML Workshop*, 2020.
- [5] A. Albaseer, B. S. Ciftler, M. Abdallah, and A. Al-Fuqaha, "Exploiting unlabeled data in smart cities using federated learning," *2020 International Wireless Communications and Mobile Computing (IWCMC)*, 2020.
- [6] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated learning with non-iid data," *arXiv preprint arXiv:1806.00582*, 2018.
- [7] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," *MLSys 2020*, 2018.
- [8] L. Huang, Y. Yin, Z. Fu, S. Zhang, H. Deng, and D. Liu, "Loadaboost: Loss-based adaboost federated machine learning on medical data," *arXiv preprint: 1811.12629*, 2018.
- [9] S. Caldas, J. Konečný, H. B. McMahan, and A. Talwalkar, "Expanding the reach of federated learning by reducing client resource requirements," *arXiv preprint arXiv:1812.07210*, 2018.
- [10] W. Luping, W. Wei, and L. Bo, "Cmfl: Mitigating communication

- overhead for federated learning,” in *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2019, pp. 954–964.
- [11] F. Chen, M. Luo, Z. Dong, Z. Li, and X. He, “Federated meta-learning with fast convergence and efficient communication,” *arXiv preprint arXiv:1802.07876*, 2018.
- [12] A. Bhowmick, J. Duchi, J. Freudiger, G. Kapoor, and R. Rogers, “Protection against reconstruction and its applications in private federated learning,” *arXiv preprint arXiv:1812.00984*, 2018.
- [13] R. C. Geyer, T. Klein, and M. Nabi, “Differentially private federated learning: A client level perspective,” *arXiv preprint arXiv:1712.07557*, 2017.
- [14] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, “Practical secure aggregation for federated learning on user-held data,” *arXiv preprint arXiv:1611.04482*, 2016.
- [15] O. Chapelle, B. Scholkopf, and A. Zien, “Semi-supervised learning (chapelle, o. et al., eds.; 2006)[book reviews],” *IEEE Transactions on Neural Networks*, vol. 20, no. 3, pp. 542–542, 2009.
- [16] D.-H. Lee, “Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks,” in *Workshop on challenges in representation learning, ICML*, vol. 3, no. 2, 2013.
- [17] A. Rasmus, M. Berglund, M. Honkala, H. Valpola, and T. Raiko, “Semi-supervised learning with ladder networks,” in *Advances in neural information processing systems*, 2015, pp. 3546–3554.
- [18] A. Tarvainen and H. Valpola, “Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results,” in *Advances in neural information processing systems*, 2017, pp. 1195–1204.
- [19] S. Laine and T. Aila, “Temporal ensembling for semi-supervised learning,” in *ICLR*, *arXiv:1610.02242*, 2017.
- [20] T. Miyato, S.-i. Maeda, M. Koyama, and S. Ishii, “Virtual adversarial training: a regularization method for supervised and semi-supervised learning,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 8, pp. 1979–1993, 2018.
- [21] S. Park, J.-K. Park, S.-J. Shin, and I.-C. Moon, “Adversarial dropout for supervised and semi-supervised learning,” *AAAI*, 2018.
- [22] C. Li, T. Xu, J. Zhu, and B. Zhang, “Triple generative adversarial nets,” in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017, pp. 4088–4098. [Online]. Available: <https://proceedings.neurips.cc/paper/2017/file/86e78499eeb33fb9cac16b7555b50767-Paper.pdf>
- [23] B. Athiwaratkun, M. Finzi, P. Izmailov, and A. G. Wilson, “There are many consistent explanations of unlabeled data: Why you should average,” *ICLR*, 2019.
- [24] Q. Xie, Z. Dai, E. Hovy, M.-T. Luong, and Q. V. Le, “Unsupervised data augmentation for consistency training,” *arXiv preprint arXiv:1904.12848*, 2019.
- [25] D. Berthelot, N. Carlini, E. D. Cubuk, A. Kurakin, K. Sohn, H. Zhang, and C. Raffel, “Remixmatch: Semi-supervised learning with distribution matching and augmentation anchoring,” in *ICLR*, 2019.
- [26] K. Sohn, D. Berthelot, C.-L. Li, Z. Zhang, N. Carlini, E. D. Cubuk, A. Kurakin, H. Zhang, and C. Raffel, “Fixmatch: Simplifying semi-supervised learning with consistency and confidence,” *arXiv preprint: 2001.07685*, 2020.
- [27] P. Cascante-Bonilla, F. Tan, Y. Qi, and V. Ordonez, “Curriculum labeling: Revisiting pseudo-labeling for semi-supervised learning,” 2020.