FedCPF: An Efficient-Communication Federated Learning Approach for Vehicular Edge Computing in 6G Communication Networks

Published in IEEE Transactions on Intelligent Transportation Systems · August 2021

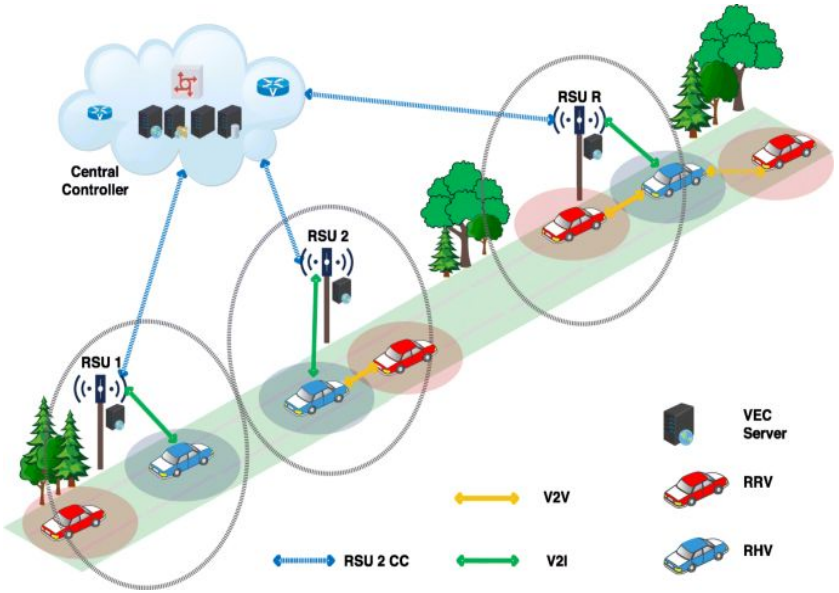
Authors:
Su Liu
Jiong Yu
Xiaoheng Deng
Shaohua Wan

Presented By:
Ittehad Saleh Chowdhury (11)
Reyadath Ullah Akib (33)

Vehicular Edge Computing(VEC)

Vehicular Edge Computing (VEC) refers to the implementation of edge computing technologies within vehicles or on the edge of vehicular networks.

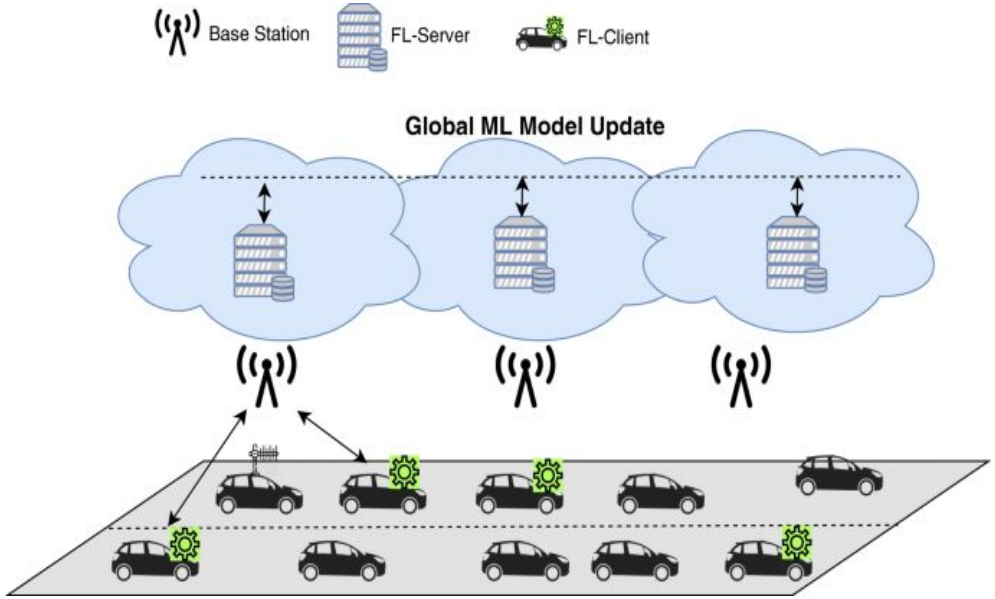
- Faster data processing
- Reduced latency and faster response time
- Real time decision making
- Enhanced security



Federated Learning(FL) in VEC

Federated learning in vehicular edge computing leverages the concept of collaborative learning to enhance data processing and model training within the vehicular network.

- Decentralized Machine Learning
- Privacy preserving approach
- Efficient knowledge sharing



Limitations of traditional FL

High Communication overhead due to primary three reasons:

- Slow convergence of the global model resulting in several communication rounds
- Slow aggregation procedure due to several participants
- Prolonged training time due to slow devices and straggler links

Reducing the Communication Rounds

Traditional method uses an algorithm called FedAVG with fixed epochs for all clients which results in slow communication rounds and slow convergence.

Fed**C**PF uses **CUSTOMIZED** local training strategy with a constraint item in the objective function on local clients to improve the convergence speed for reducing the total communication rounds.

Decreasing the Number of Upload Clients

Existing methods use client selection algorithms without considering the heterogeneity of data in vehicular clients which aren't suitable.

FedC**P**F uses **PARTIAL** client participation rule which allows a few clients with massive local data to upload their training results simultaneously, decreasing communication costs in each round.

Improving the Dynamic of the Aggregation Mechanism

The aggregation phase of the Federated Learning is prolonged due to inactive devices or slow connections.

FedCP**F** uses **FLEXIBLE** aggregation policy to drop the clients who exceed the time limitation to dynamically adjust the number of clients during the aggregation phase, thereby reducing each communication overheads.

FL setup in VEC

The general setting in FL is that various clients have the same neural network model in the initial. These distributed devices train local models with their private data for a few epochs.

A communication rounds consists of the following phases:

1. Downlink communication phase
2. Federated training phase
3. Uplink communication phase
4. Aggregation phase

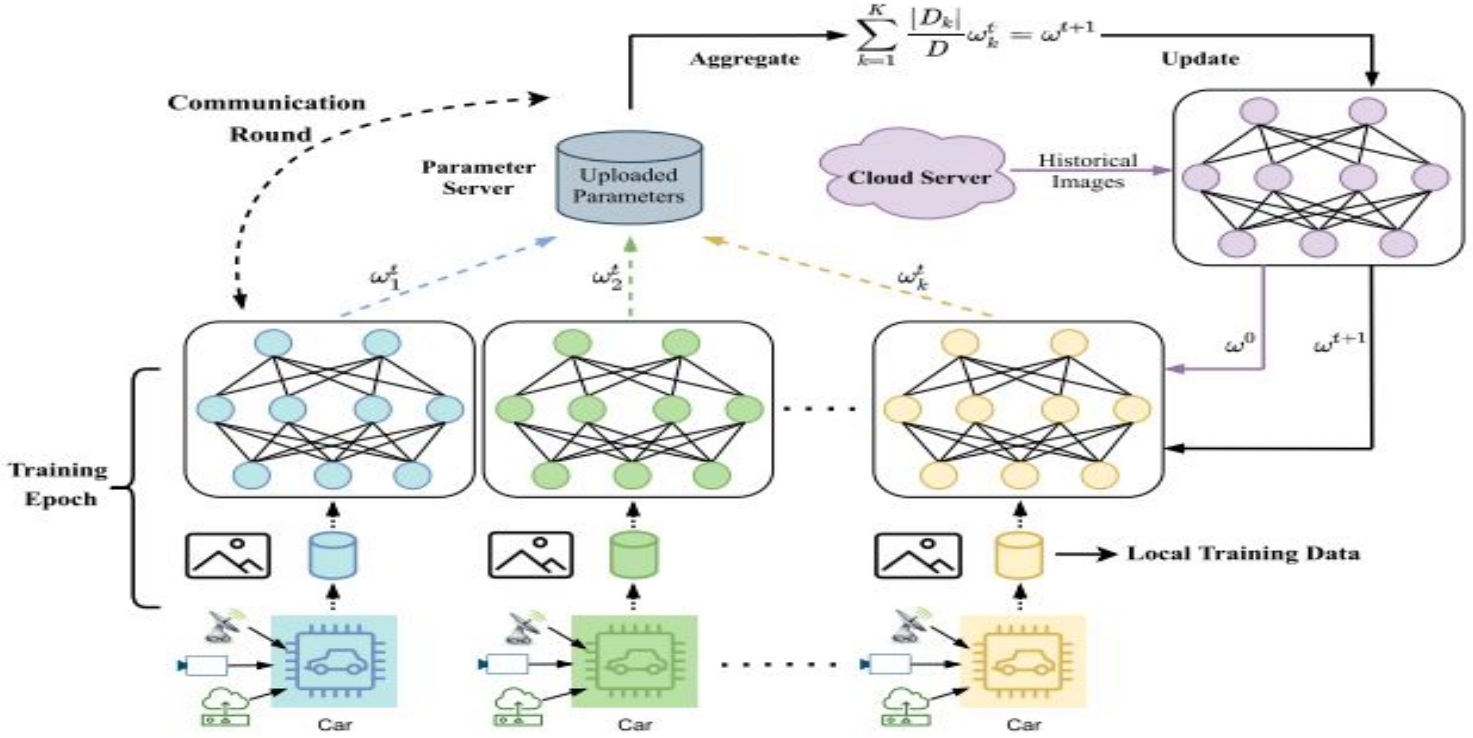


Fig. 1. Flowchart of original federated learning algorithm in VEC with traditional algorithm FedAvg.

System Framework for short driving vehicles(a)

- Vehicles collect data while driving and store it locally.
- When parked, vehicles use their local data to train their private model.
- Proximate RSU acts as the parameter server.
- Vehicles upload their private training results to the RSU, which aggregates the data to complete a federated training round.

System Framework for long driving vehicles(b)

- Long driving vehicles may not have immediate access to parking conditions for training tasks.
- Data from vehicles is offloaded to Roadside Units (RSUs).
- Each RSU acts as a client and collaboratively performs Federated Learning (FL) training
- The RSUs upload their local training results to the cloud server for aggregation.

2-Layer Network Architecture

FedCPF proposes a two-layers of network architecture, which contains a devices layer and a parameter server layer.

For System Framework (a):

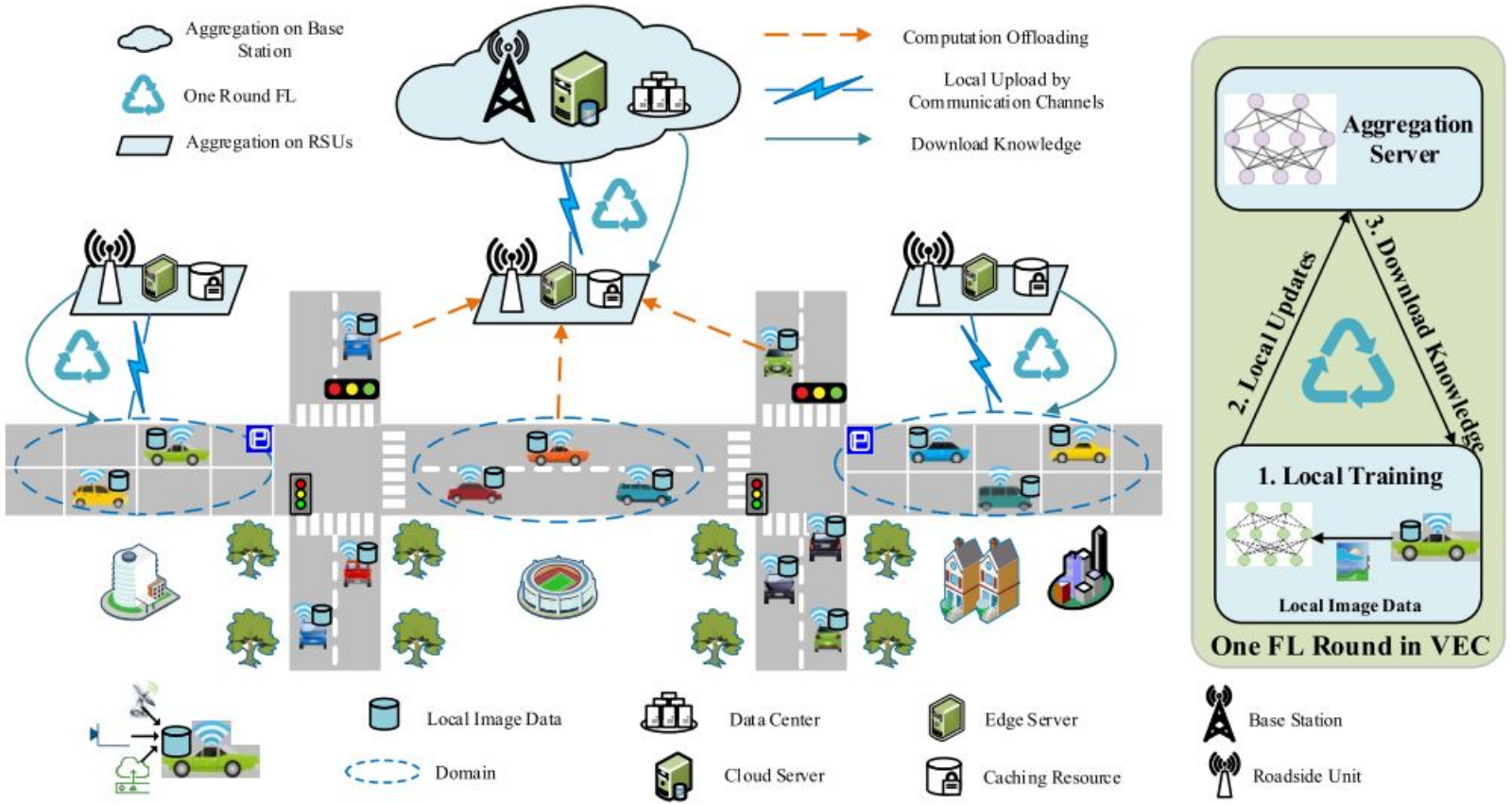
Device Layer : Vehicles

Parameter server Layer: RSUs

For System Framework (b):

Device Layer : RSUs

Parameter server Layer: Cloud server


Aggregation on Base Station
One Round FL
Aggregation on RSUs
Computation Offloading
Local Upload by Communication Channels
Download Knowledge
Aggregation Server
2. Local Updates
3. Download Knowledge
1. Local Training
Local Image Data
One FL Round in VEC
Local Image Data
Domain
Data Center
Cloud Server
Edge Server
Caching Resource
Base Station
Roadside Unit

Proposed Methods

System Model

Notations

K = Total number of clients

k = Index of the clients

D_k = Local Dataset of each client

$|.|$ Denotes the size of the set

$D_1, D_2, \ldots\ldots D_k$ = Whole dataset

$|D| = \sum_{k=1}^{K} |D_k|$

$\{x_k\}$ some local dataset with label set $\{y_k\}$ in D_k

ω^0 = Initial global parameter

ω_k^t = Local model parameters of the k_{th} client for the t_{th} communication round

$F(\omega)$ = Averaged global model parameters after the aggregate phase

$f_k(\omega_k)$ = Loss function of the k_{th} client

Objective

- for some input $\boldsymbol{x_i}$ in the neural network (local) $\boldsymbol{y_i}$ is the desired output with some loss function $\boldsymbol{f_i(\omega)}$.
- At the t-th communication round, the objective function is to minimize the training error as depicted for some client $\boldsymbol{k}$

$$\min_{\boldsymbol{\omega}_k^t} F_k^t\left(\boldsymbol{\omega}_k\right)=\frac{1}{\left|D_k\right|} \sum_{i \in D_k} f_i\left(\boldsymbol{\omega}_k^t\right) .$$

Generalized Loss Function

- **F(ω)** denotes the averaged global model parameters.
- Optimizing ***F(ω)*** is equivalent to minimizing the weighted average of local loss function $F_k(\omega_k)$
- Each client performs the training task locally, and shares their own local parameters.

$$\min_{\omega} F(\omega) = \sum_{k=1}^{K} \frac{|D_k|}{|D|} F_k(\omega_k). \quad (1)$$

Calculating The Communication Cost

- Consider n(t) devices participate on the t-th communication round. [n(t) <= K]
- The total communication cost can be expressed with the equation,

$$W = \sum_{t=1}^{T} \{n(t) + K\}.\omega^*$$

- Upper bound of the communication overhead,

$$\tilde{W} = 2T(K \cdot \omega^*) \quad \textbf{(2)}$$

FedCPF : Customized Local Training Strategy

- Gradient Diversity quantifies the degree to which individual gradients of the loss functions vary.
- This phenomenon is prevalent in FL due to its distributed nature
- Our goal is to reduce the weight divergence.

Weight Divergence

Weight Divergence in Federated Learning compared to Centralized Approach

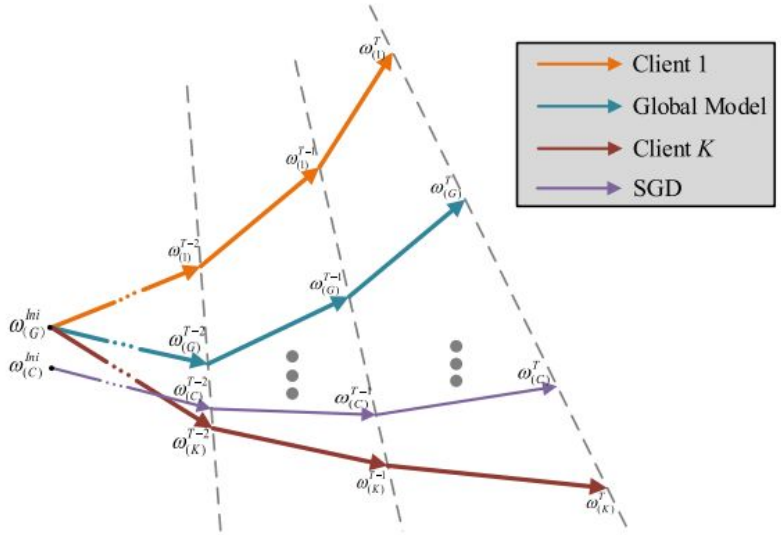


Fig. 3. Illustration of the weight divergence for FL.

Weight Divergence

- The fundamental reason for weight divergence is, $\sum_{i=1}^{C} \| p^{(k)}(y=i) - p(y=i) \|$ which is termed as Earth Mover's Distance (EMD)

- The weight divergence in FL is caused by two aspects,
  1. Accumulation of weight divergence
  2. Data distribution on each client may not be representative of the whole data distribution.

Reducing Gradient Diversity

- We introduce a constraint item to narrow down the gradients diversity among local functions effectively.
- Instead of minimizing the local loss function $F_k(.)$, client k minimizes the following objective equation,

$$\min_{\omega} g_k\left(\omega_k; \omega^t\right) = F_k\left(\omega_k\right) + \frac{\varepsilon}{2}\left\|\omega_k - \omega^t\right\|^2. \quad \textbf{(3)}$$

- The constraint **" ϵ /2 $||\omega_k - \omega^t||^2$ "** limits the number of local updates without manually setting the local epochs.
- Less local computation corresponds to higher accuracy

FedCPF : Partial Client Participation Rule

- There are often too many clients communicating with the parameter server
- FL Efficiency is affected by two things
  - Limitations of uplink communication
  - Some clients only have a small amount of data
    - This will become noise in the aggregation phase.
    - Which will eventually cause the global model to be more biased.

Solution ?

- At each global communication round, a subset of the clients, denoted as $\{S_t\}$, are selected and local models on only those clients are used to optimize the equation,

$$\min_{\omega} F(\omega) = \sum_{k=1}^{K} \frac{|D_k|}{|D|} F_k(\omega_k)$$

- Clients upload both ω_k and the size of the local data set
- Parameter server calculates the probability of each client being selected, denoted as p_k.
- Clients with larger data sets tend to have a larger p_k

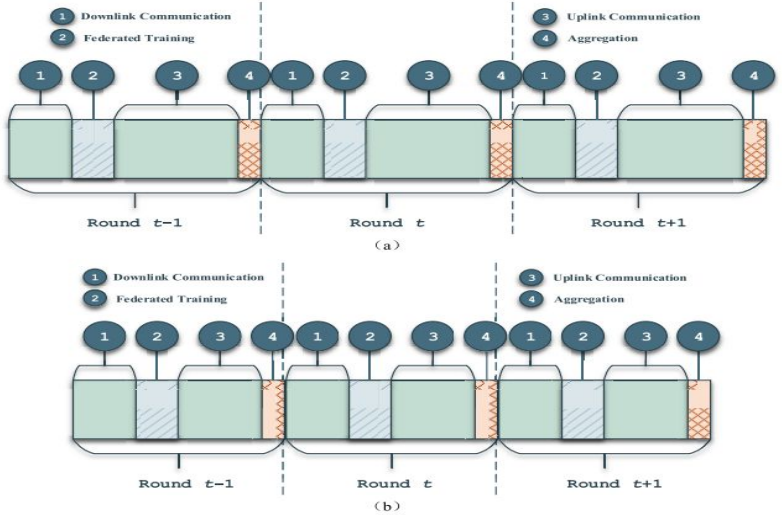
Modified Objective Function

- The subset of selected clients in each round is different
- This causes the global model contain as much information about the local clients as possible
- With all the new information we can change the loss function to,

$$\min_{\boldsymbol{\omega}} F(\boldsymbol{\omega}) = \sum_{k=1}^{|S_t|} p_k g_k \left(\boldsymbol{\omega}_k; \boldsymbol{\omega}^t\right), \sum_{k=1}^{|S_t|} p_k = 1$$

FedCPF : Flexible Aggregation Policy

- Uplink communication phase is the most time consuming
- The parameter server needs to wait for all the clients to upload local training results and aggregate them



Solution ?

- To resolve this, a flexible aggregation policy to limit the time of the uplink communication phase is required
- ***t***-th uplink communication phase begins at time r_s^t and ends at time r_e^t . r_k denotes the time when the ***k***-th client finishes the local training.
- Transmission time should be restricted as equation,

$$r_s^t \leq r_k \leq r_e^t.$$

Flexible Aggregation Policy (cont.)

- The running time of a complete FL round can be represented as,

$$T_g = T_{co} + \theta_k^t . T_{cp}$$

- T_{co} denotes the communication time which we reduce this by shortening the uplink communication time.
- Based on the flexible aggregation policy, the number of clients participating in each round of aggregation has shown a declining trend
- This results in the reduction of communication overhead.

FedCPF Algorithm Summarized (Pseudocode)

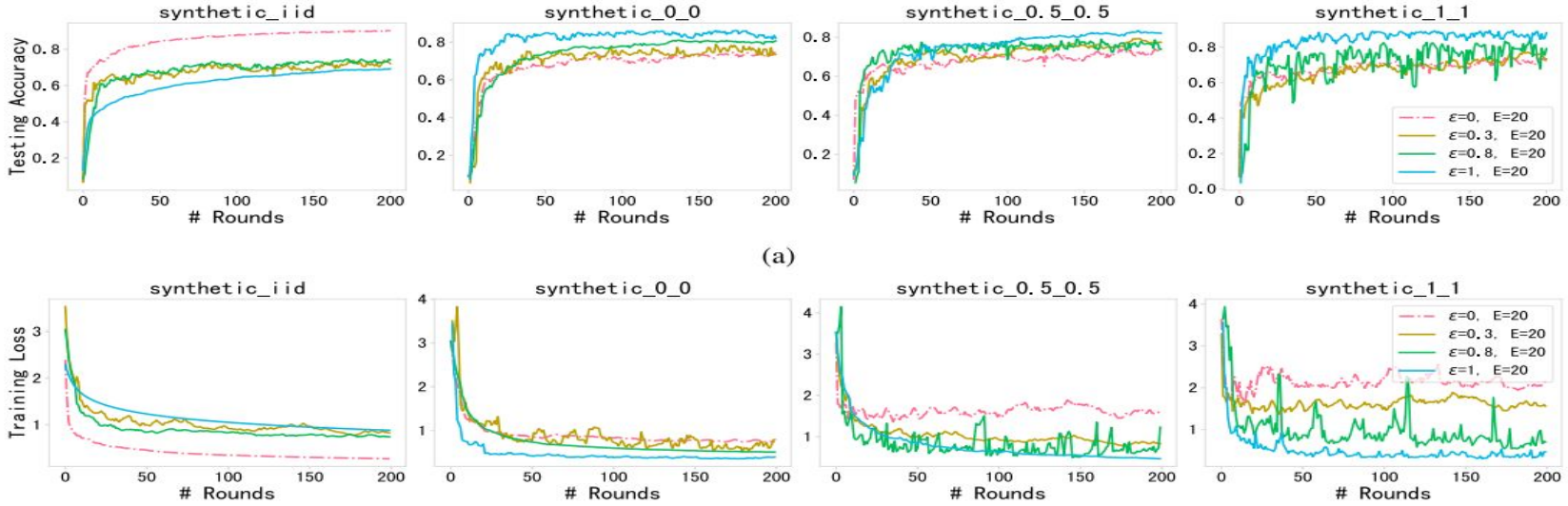
Input: The K clients are indexed by k, toal communication rounds of FL T, the coefficient of the constraint item ε, the total number of the clients K, private data set on client k is $D_k, k = 1, \ldots, K$, $n(t)$ denotes the number of chosen clients at the t-th communication round, and constitutes a subset S_t, the number of local training epochs is θ_k^t, the initialized parameters of the model ω^0.

Output: The updated global model parameters ω_k^{t+1}

Initialize all the local models parameters with ω^0
for $t \leftarrow 0$ to $T-1$ **do**
    Downlink Communication:
    Server picks a subset S_t of K clients with the probability p_k.
    Server sends ω^t to clients in the subset S_t.
    Federated Training:
    for each client $k \in S_t$ **in parallel do**
        Each client computes the θ_k^t local epochs with equation (8) to find a minimum loss function through the constraint item.
    end for
    Uplink Communication:
    Each client in the subset S_t sends the ω_k^{t+1} back to the server before the defined time considering the flexible aggregation policy.
    Aggregation:
    Server average the ω^{t+1} with the probability p_k.
    Server distribute the ω^{t+1} to all the clients.
end for

Experimental data for Customized Local Training

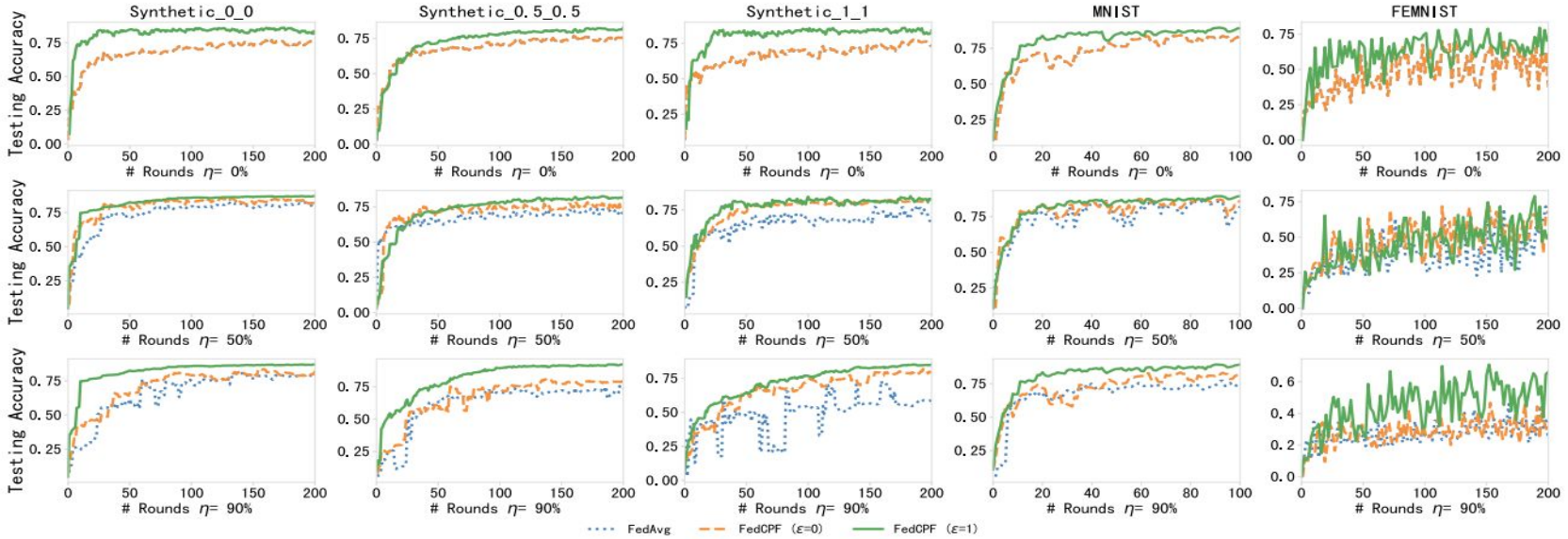
We compare the accuracy and loss function values of our proposed approach FedCPF (with varied values of ε) to the baseline approach FedCPF (with ε = 0)



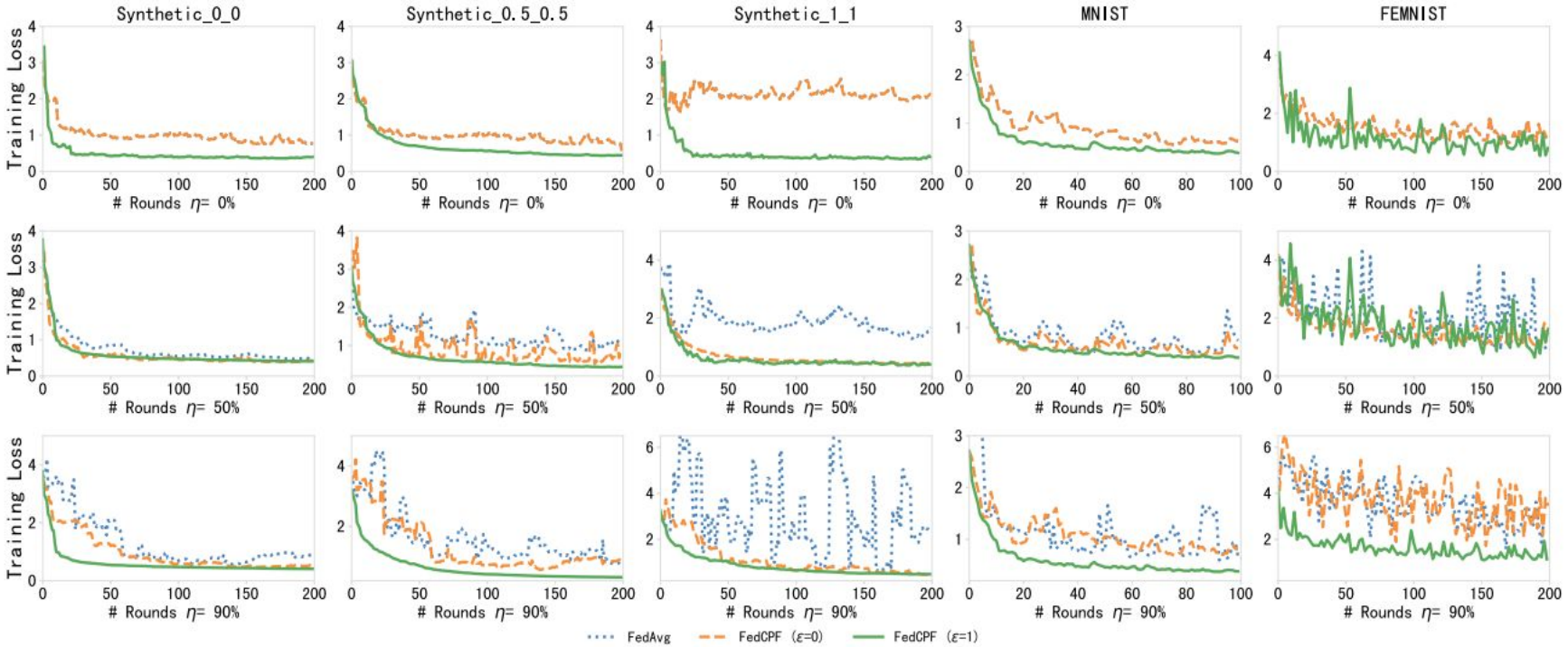
The accuracy in FedCPF increases by 4.72%, 3.69%, and 10.51%, compared with the FedAvg, respectively.

Experimental data for Flexible Aggregation

We incorporate the Flexible Aggregation policy by setting a parameter η as the number of epochs to 0%, 50%, and 90% of the selected clients.

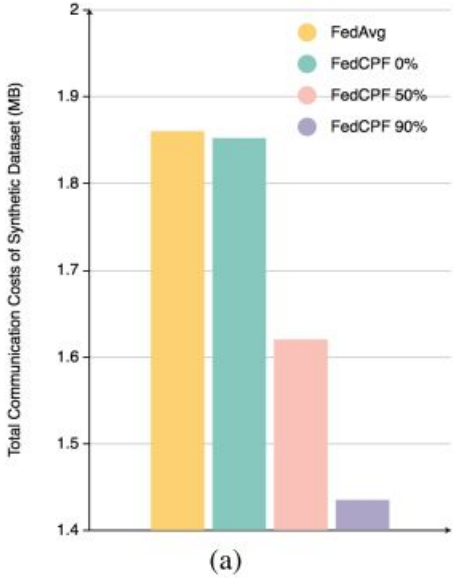


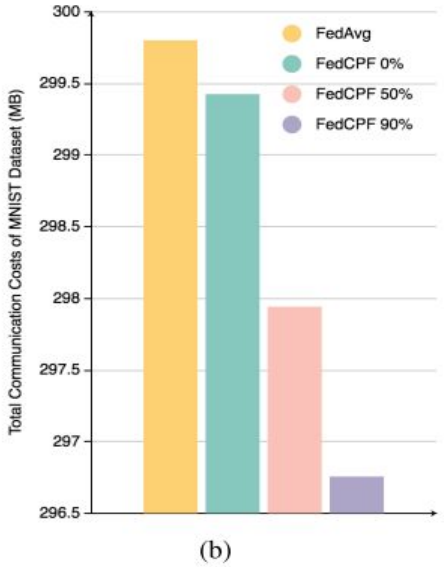
Experimental data for Flexible Aggregation(CONT.)

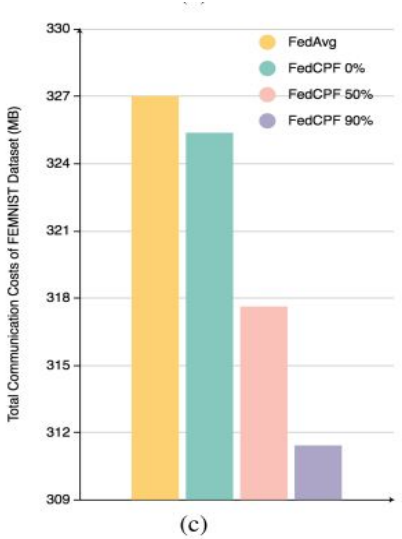


Communication Cost

Compared with FedCPF (ε = 0, η = 50%) and FedAvg (f = 0.1, η = 0%) algorithms, the average communication optimization rate is 1.74 times.







Thank You!