H-FL: A Hierarchical Communication-Efficient and Privacy-Protected Architecture for Federated Learning

He Yang*
Xi'an Jiaotong University
sleepingcat@stu.xjtu.edu.cn

Abstract

The longstanding goals of federated learning (FL) require rigorous privacy guarantees and low communication overhead while holding a relatively high model accuracy. However, simultaneously achieving all the goals is extremely challenging. In this paper, we propose a novel framework called **h**ierarchical **f**ederated **l**earning (H-FL) to tackle this challenge. Considering the degradation of the model performance due to the statistic heterogeneity of the training data, we devise a runtime distribution reconstruction strategy, which reallocates the clients appropriately and utilizes mediators to rearrange the local training of the clients. In addition, we design a compression-correction mechanism incorporated into H-FL to reduce the communication overhead while not sacrificing the model performance. To further provide privacy guarantees, we introduce differential privacy while performing local training, which injects moderate amount of noise into only part of the complete model. Experimental results show that our H-FL framework achieves the state-of-art performance on different datasets for the real-world image recognition tasks.

1 Introduction

Federated learning (FL) is a promising distributed paradigm for training a shared model while keeping all the training data localized [Yang *et al.*, 2019; Kairouz *et al.*, 2019; Konečný *et al.*, 2016]. However, FL always involves expensive communication and privacy concerns in order to maintain a great model performance [Li *et al.*, 2020; Zhang *et al.*, 2021]. Therefore, how to find a great balance among model performance, communication overhead and privacy requirements is a long-term, challenging goal.

From a methodological standpoint, DGC [Lin *et al.*, 2017] and FetchSGD [Rothchild *et al.*, 2020] have given a good trade-off between communication overhead and model performance by compressing the gradients and giving some corrections. NbAFL [Wei *et al.*, 2020] and DP-FedAVG [McMahan *et al.*, 2017b] provide strong privacy guarantees via differential privacy without undue sacrifice on model performance. SplitNN [Vepakomma *et al.*, 2018] can achieve higher model performance in contrast to the aforementioned methods while protecting sensitive raw data. All these works try to make some trade-offs from different perspectives. However, when treating model performance, communication overhead and privacy requirements as a whole perspective, it will introduce a completely new contradiction: the contradiction between communication overhead and privacy requirements while maintaining model performance in a certain level. Since when utilizing some privacy protection methods such as differential privacy and secure multiparty computing to provide privacy guarantees, it will inevitably introduce additional communication overhead directly or slow down the convergence rate, leading to requiring extra communication rounds for FL algorithms to converge. Therefore, we cannot just do simple combinations from different perspectives.

In this paper, we develop a **h**ierarchical **f**ederated **l**earning architecture (H-FL) as shown in Figure 1. To counter-weigh the degradation of model performance due to statistic heterogeneity of the training data, H-FL introduces mediators to reconstruct the local distributions. We cluster the clients according to the KL divergence between local distributions of each client and a uniform distribution, as well as the information entropy of the local distributions, and then reallocate them to different mediators. When participating in federated tasks, H-FL selects mediators rather than clients and each mediator rearranges its clients to perform the training tasks in order to alleviate the statistic heterogeneity. In addition, we design a compression-correction mechanism to reduce the communication overhead without sacrificing the model performance, which significantly compresses the extracted features of the clients uploaded to mediators and corrects the corresponding gradients download from the mediators. To further provide privacy guarantees for clients, we introduce differential privacy when each client conducts its local training.

Our contributions can be summarized as following:

- To the best of our knowledge, H-FL is the first attempt to treat model performance, communication overhead and privacy requirements as a whole perspective to find a great balance among them.

*Contact Author

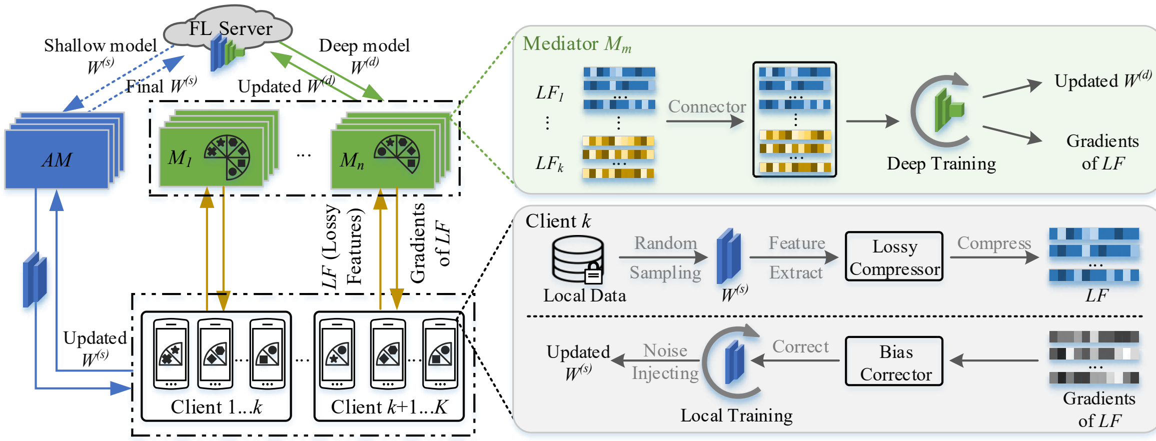


Figure 1: H-FL architecture. FL server splits the complete model into two pieces: shallow model and deep model, and then distributes the former one to the **A**ggregation **M**ediator ($\mathcal{AM}$) and the latter one to the other Mediators ($\mathcal{M}$). $\mathcal{AM}$ distributes the shallow model to all the clients and is responsible for aggregating shallow models. In addition, FL server is responsible for aggregating deep models. Particularly, $\mathcal{AM}$ sends the final global shallow model to FL server at the end of the collaborative training.

- We devise a runtime distribution reconstruction strategy to alleviate the statistic heterogeneity of the training data while not compromising user privacy. Moreover, we design a compression-correction mechanism to reduce the communication overhead without sacrificing the model performance.
- Extensive experiments on different datasets show that our H-FL architecture achieves state-of-the-art performance on federated image recognition tasks.

2 Related Research

Federated learning is a collaborative distributed learning paradigm which removes the necessity to pool the raw data out from local clients. Specifically, FedAVG algorithm proposed in [McMahan *et al.*, 2017a] aims to reduce the communication overhead while maintaining a good performance of the model on non-IID (Independent and identically distributed) training data, which is used as our baseline in Section 4. Furthermore, concurrent works such as [Lin *et al.*, 2018; Sattler *et al.*, 2019] have focused on further reducing communication overhead in FL via gradient sparsification, and propose solutions to counter-weigh the reduction in accuracy due to the statistic heterogeneity of the training data. Concretely, DGC [Lin *et al.*, 2018] employs momentum correction and local gradient clipping on top of the gradient sparsification to ensure no loss of accuracy. In addition, DGC also uses momentum factor masking and warmup training to overcome the staleness problem caused by reduced communication. STC [Sattler *et al.*, 2019] propose a sparse ternary compression (STC) framework to reduce the communication overhead in FL, which enables ternarization and optimal Golomb encoding of the weight updates and also behaves robust to non-IID training data. We conduct a comprehensive analysis and comparison with the aforementioned methods in Section 4 to illustrate the effectiveness of our H-FL framework. in Section 4 to illustrate the effectiveness of our H-FL framework.

Notation	Definition
$W_t^{(d)}$	global deep model at round t
$W_t^{(s)}$	global shallow model at round t
$W_t^{(c)}$	shallow model kept in client c at round t
$W_t^{(m)}$	deep model kept in meditor m at round t
$\mathcal{U}$	all the clients
$\mathcal{P}$	sampling probability of each client
$\mathcal{S}$	sampling probability of each example
$\mathcal{C}$	global compression ratio
$\mathcal{I}$	iterations of deep training
$\mathcal{L}$	ℓ_2-norm of the clipped gradients
σ	noise level

Table 1: Notations and Definitions

3 Our Approach

In this section, we propose a hierarchical FL architecture as shown in Figure 1 to find a great balance among model performance, communication overhead and privacy requirements.

3.1 Adversary Model

We first assume that all the components (FL Server, Mediators, Clients) in H-FL have following abilities: 1) they are ***honest-but-curious***, which means that they will honestly follow the designed protocol but are curious about the others' local data; 2) they have arbitrary auxiliary information to help infer a specific client's private information during the process of collaboratively building a shared model; 3) they do not collude with each other, which means that they will not provide any additional information to clients during the training.

3.2 Initialization

FL server first splits the complete model into two components: shallow model and deep model, then distributes the former one to the **A**ggregation **M**ediator ($\mathcal{AM}$) and the latter one to the other Mediators ($\mathcal{M}$). $\mathcal{AM}$ distributes the shallow model to all the clients. At the same time, FL server initializes the global hyper-parameters such as learning rate η, sampling probability of each client $\mathcal{P}$, sampling probability of each example $\mathcal{S}$, global compression ratio $\mathcal{C}$ ($\mathcal{C} < 0.5$), iterations of deep training in mediators $\mathcal{I}$, ℓ_2-norm of the clipped gradients $\mathcal{L}$ and noise level σ. Specifically, when sampling locally in practice, we randomly permute the local data and partition them into mini-batches of the appropriate sizes for efficiency.

3.3 Runtime Distribution Reconstruction

In FL settings, as the training data resident in the individual clients is collected by the clients themselves on the basis of their local environments, the distribution of the local datasets will considerably differ with each other. Considering this characteristic, we redefine the optimization objective function of federated learning training on non-IID datasets as follows:

$$\min_{w,p^{(c)}} \mathrm{E}_{(x,y)\sim p^{(c)}} \left[\ell(f\left(x; w^{(c)}\right), y)\right] + \sum_{c} \mathcal{D}_{KL}\left(p \| p^{(c)}\right) \quad (1)$$

where $w^{(c)}$, $p^{(c)}$, p, $\mathcal{D}_{KL}$ are the weights of client *c*, the local distribution of client *c*, the distribution of potential global training data, KL divergence between p and $p^{(c)}$, respectively. When the latter term is approximate to *0*, it will degrade to an optimization problem under IID. In general FL settings, $p^{(c)}s$ are a series of different fixed distributions such that the latter term is a fixed value and the optimization objective will be consistent.

Whereas we consider $p^{(c)}s$ as variable distributions rather than fixed distributions in H-FL, so we can change local distributions arbitrarily. An intuitive way is to gather the clients' local data and form a series of different new distributions, each of which is approximate to the potential global distribution p, enabling the latter term in Formula (1) to be *0*. However, sharing local data raises serious privacy risks and causes high communication overhead. Therefore, we introduce the runtime distribution reconstruction strategy to mitigate differences among local distributions while meeting the privacy requirements.

Specifically, a uniform distribution $p^{(r)}$ is initialized and broadcast among the clients. Each client calculates the information entropy $\mathcal{H}^{(c)}$ of its local distribution $p^{(c)}$ and KL divergence $\mathcal{D}_{KL}(p^{(r)} \| p^{(c)})$ between $p^{(r)}$ and $p^{(c)}$. Furthermore, K-means algorithm is utilized to cluster the clients according to the binary group $(\mathcal{H}^{(c)}, \mathcal{D}_{KL}(p^{(r)} \| p^{(c)}))$. Then H-FL randomly selects clients from each cluster, marks them as a group, and assigns the group to one of mediators. The allocation pattern loops until all the clients are assigned to the corresponding mediator.

Algorithm 1 Runtime distribution reconstruction

Input: $\mathcal{U}$, $\mathcal{M}$
Parameter: $W^{(c)}$,$\mathcal{P}$, $\mathcal{S}$, $\mathcal{C}$
Output: $\mathcal{B}^{(m)}$

```
Randomly initialize a distribution p^(r)
for each c ∈ U in parallel do
    Compute H^(c), D_KL(p^(r) || p^(c))
end for
Cluster according to (H^(c), D_KL(p^(r) || p^(c)))
for each m ∈ M do
    Randomly select clients from each cluster according to
    the same ratio 1/|M| and assign them to m
    B^(m) ← ∅
end for
M^t ← (Randomly sampling mediators in M)
for each m ∈ M^t do
    U^t ← (Randomly sampling clients in U with P)
    for each c ∈ U^t do
        Randomly sampling a mini-batch X^(c) with S
        O^(c) ← W^(c) X^(c)
        k ← ⌊|O^(c)| * C⌋
        B^(m) ← B^(m) ∪ LF(O^(c))
    end for
end for
return B^(m)
```

When performing local training, each client utilizes the shallow model to extract features, which will be compressed by the lossy compressor (introduced in subsection 3.4) and sent to the corresponding mediator. After that, each mediator concatenates the received features through a connector (as shown in Figure 1) to obtain synthetic features. This procedure can be considered as sampling from a virtual reconstructed distribution $p^{(m)}$ and then conducting forward propagation using the shallow model (see Algorithm 1). Intuitively, $p^{(m)}$ will be more approximate to the potential global distribution p than $p^{(c)}$. The optimization objective function will be changed to the following form:

$$\begin{aligned} \min_{W,p^{(m)}} \mathrm{E}_{(x,y)\sim p^{(m)}} &\left[\ell\left(f\left(x; W^{(c)}, W^{(m)}\right), y\right)\right] \\ &+ \sum_{m} \mathcal{D}_{KL}\left(p \| p^{(m)}\right) \end{aligned} \quad (2)$$

Assuming that there exists enough clients, $p^{(m)}$s will infinitely approximate the potential global distribution p and the latter term will be *0*, which is translated to the optimization problem under IID. When finishing the distribution reconstruction, each mediator leverages the synthetic features to train the deep model and then sends back the gradients of the synthesized features to the clients to assist training the shallow model. In this way, H-FL alleviates the statistic heterogeneity of the training data while not compromising user privacy.

3.4 Compression-Correction Mechanism

To reduce the communication overhead, each participating client compresses the extracted features through the lossy compressor in Figure 1 by:

$$LF(\mathcal{O}) = U_{\mathcal{O}}[:, : k]\Sigma_{\mathcal{O}}[: k]V_{\mathcal{O}}^T[: k] \tag{3}$$

where $\mathcal{O}$ is feature matrix extracted by the shallow model, $U_{\mathcal{O}}$, $\Sigma_{\mathcal{O}}$ and $V_{\mathcal{O}}^T$ are the results of singular value decomposition (SVD) respectively, $U_{\mathcal{O}}[:, : k]$, $\Sigma_{\mathcal{O}}[: k]$ and $V_{\mathcal{O}}[: k]$ represent the first k columns of $U_{\mathcal{O}}$, $\Sigma_{\mathcal{O}}$ and $V_{\mathcal{O}}$ respectively. In this way, the feature matrix can be changed to a low-rank matrix that can be expressed by as the product of two relatively small matrices, thus reducing the communication overhead.

For the sake of clarification, let us introduce some new representations:

$$\begin{aligned} \mathcal{O} &= W^{(c)}X^{(c)} \\ \mathcal{B} &= LF(\mathcal{O}) \\ \mathcal{A} &= W^{(m)}\mathcal{B} \\ \mathcal{L} &= \mathrm{E}[\ell(\mathcal{A}, y)] \end{aligned} \tag{4}$$

When updating $W^{(c)}$, we should compute $dW^{(c)}$ as follows according to the chain rule:

$$dW^{(c)} = \frac{\partial \mathcal{L}}{\partial \mathcal{A}} \cdot \frac{\partial \mathcal{A}}{\partial \mathcal{B}} \cdot \frac{\partial \mathcal{B}}{\partial W^{(c)}} \tag{5}$$

However, according to formula (3), we cannot compute $\partial\mathcal{B}/\partial W^{(c)}$ directly since there is no direct differentiable mapping from $W^{(c)}$ to $\mathcal{B}$. For convenience, $\partial\mathcal{O}/\partial W^{(c)}$ can be used instead of $\partial\mathcal{B}/\partial W^{(c)}$, which may still work but it leads to a reduction in model accuracy.

Therefore, we design a bias corrector on clients to correct the gradients of lossy features, which could build the mapping from $\mathcal{O}$ to $\mathcal{B}$ so that we can better approximate $\partial\mathcal{B}/\partial W^{(c)}$ and counter-weigh the reduction. According to the feature of SVD, we can get:

$$\mathcal{B} = U_{\mathcal{O}}\mathcal{D}_k U_{\mathcal{O}}^T \mathcal{O} \tag{6}$$

where $U_{\mathcal{O}}$ here is just the same thing as the $U_{\mathcal{O}}$ in formula (3), $\mathcal{D}_k$ represents a diagonal matrix where its first k elements on the diagonal are *1* and the rest are *0*. Therefore, $\partial\mathcal{B}/\partial W^{(c)}$ can be rewritten as:

$$\partial\mathcal{B}/\partial W^{(c)} \approx U_{\mathcal{O}}\mathcal{D}_k U_{\mathcal{O}}^T \cdot (\partial\mathcal{O}/\partial W^{(c)}) \tag{7}$$

Thus, the bias corrector can be considered as consisting of many fully connected layers stacked on top of each other, and the parameters depend on the SVD results of the features extracted from the shallow model. In other words, the parameters of the bias corrector will be updated during the procedure of forward propagation. We also compare the results for the presence or absence of the bias corrector through appropriate experiments in Section 4.

After we obtain rectified $dW^{(c)}$, we conduct gradient clipping so that the ℓ_2 norm of $dW^{(c)}$ is limited to L and then add noise for it in order to protect privacy:

$$g^{(c)} \leftarrow \frac{g^{(c)}}{\max\left(1, \left\|g^{(c)}\right\|_2 / L\right)} + \mathcal{N}\left(0, \frac{\sigma^2 L^2 I}{n^{(c)}}\right) \tag{8}$$

where $g^{(c)}$ is $dW^{(c)}$ itself, $n^{(c)}$ is the size of the sampled mini-batch in client *c*, $\mathcal{N}$ is the Gaussian distribution with mean *0* and standard deviation $\sigma LI/\sqrt{n}$.

In summary, the workflow of H-FL mainly includes run-time distribution reconstruction, training and aggregation, the pseudo-code of which is given as Algorithm 2.

Algorithm 2 The workflow for H-FL

Input: $\mathcal{U}$, $\mathcal{AM}$, $\mathcal{M}$
Parameter: $W_t^{(m)}$, $W_t^{(c)}$, $\mathcal{P}$, $\mathcal{S}$, $\mathcal{C}$, $\mathcal{I}$, $\mathcal{L}$, σ
Output: $W_{t+1}^{(d)}$, $W_{t+1}^{(s)}$
Mediators:
$\mathcal{B}^{(m)} \leftarrow$ Run-time data augmentation
for each $m \in \mathcal{M} \setminus \mathcal{AM}$ **in parallel do**
  for each epoch i from 1 to $\mathcal{I}$ **do**
    $W_t^{(m)} \leftarrow W_t^{(m)} - \eta\nabla_{W_t^{(m)}}\ell(W_t^{(m)}\mathcal{B}^{(m)}, y)$
  end for
  $d\mathcal{B}^{(m)} \leftarrow \nabla_{\mathcal{B}^{(m)}}\ell(W_t^{(m)}\mathcal{B}^{(m)}, y)$
  for each $c \in m$ **do**
    $d\mathcal{B}^{(c)} \leftarrow d\mathcal{B}^{(m)}[: n^{(c)}]$
    $d\mathcal{B}^{(m)} \leftarrow d\mathcal{B}^{(m)}[n^{(c)} :]$
  end for
end for
Clients:
for each $c \in \mathcal{U}^t$ **in parallel do**
  $\mathcal{B}^{(c)} \leftarrow U_{\mathcal{O}}^{(c)} D_k^{(c)} U_{\mathcal{O}}^{(c)T} \mathcal{O}^{(c)}$
  $dW_t^{(c)} \leftarrow d\mathcal{B}^{(c)}\partial\mathcal{B}^{(c)}/\partial W_t^{(c)}$
  $dW_t^{(c)} \leftarrow dW_t^{(c)} + \mathcal{N}\left(0, \sigma^2\mathcal{L}^2 I/n^{(c)}\right)$
  $W_t^{(c)} \leftarrow W_t^{(c)} - \eta dW_t^{(c)}$
end for
FL Server:
$W_{t+1}^{(d)} \leftarrow \frac{\sum_{m\in\mathcal{M}\setminus AM} W_t^{(m)}}{|\mathcal{M}\setminus AM|}$
AM:
$W_{t+1}^{(s)} \leftarrow \frac{\sum_{c\in\mathcal{U}^t} W_t^{(c)}}{|\mathcal{U}^t|}$

Theorem 1. *Formula (8) satisfies differential privacy in distribute environment and the privacy loss can be tracked via moments accountant.*

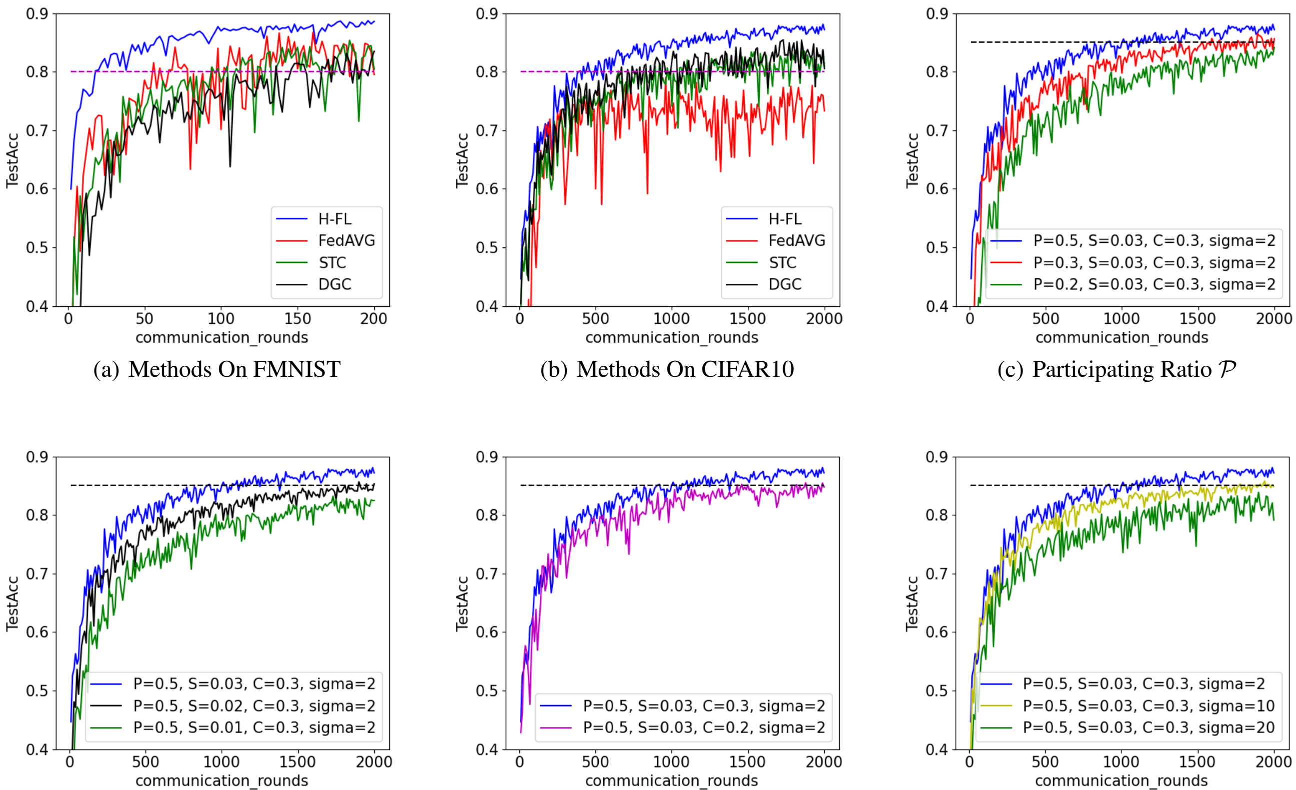
Proof. We can consider the first term of formula (8) as follows approximately:

$$g^{(c)} = \frac{\sum_{i=1}^{n^{(c)}} g\left(x_i^{(c)}\right) / \max\left(1, \left\|g\left(x_i^{(c)}\right)\right\|_2 / L\right)}{n^{(c)}} \tag{9}$$

where g is the gradient of backward propagation, $x_i^{(c)}$ is the *i*-th example of client *c* and $n^{(c)}$ is the size of sampled mini-batch of client *c*. In addition, we can also consider the latter term of formula (8) as follows according to central limit theorem:

$$\mathcal{N}\left(0, \frac{\sigma^2 L^2 I}{n^{(c)}}\right) = \frac{\sum_{i=1}^{n^{(c)}} \mathcal{N}\left(0, \sigma^2 L^2 I\right)}{n^{(c)}} \tag{10}$$

Therefore, we can rewrite formula (8) as formula (11), which satisfies example-level differential privacy for each client according to Theorem 2 [Abadi *et al.*, 2016]. In addition, since L and σ are the same for all clients, the privacy loss accumulated via moment accountant for each client in the distributed environment is the same. It also satisfies differential privacy in the distributed environment according to



(a) Methods On FMNIST (b) Methods On CIFAR10 (c) Participating Ratio $\mathcal{P}$

(d) Sampling Probability $\mathcal{S}$ (e) Compression Ratio $\mathcal{C}$ (f) Noise Level σ

Figure 2: Behavior of the Model Performance and Influence of Different Parameters for H-FL.

differential privacy parallel principle.

$$g^{(c)} = \frac{\sum_{i=1}^{n^{(c)}} \frac{g\left(x_i^{(c)}\right)}{\max\left(1, \left\|g\left(x_i^{(c)}\right)\right\|_2 / L\right)} + \mathcal{N}\left(0, \sigma^2 L^2 I\right)}{n^{(c)}} \tag{11}$$

Dataset	Clients	Mediators	η	classes per client	$\mathcal{I}$	$\mathcal{L}$
CIFAR10	100	3	0.015	3	10	1
FMNIST	100	3	0.015	2	10	1

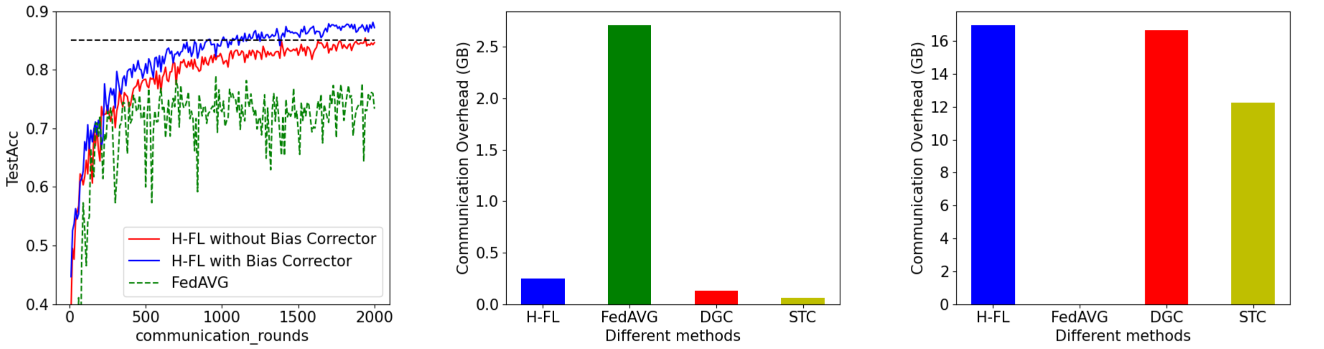
Table 2: Experiment Settings

4 Experimental Results

We evaluate H-FL on different datasets and compare the performance to FedAVG [McMahan *et al.*, 2017a], STC [Sattler *et al.*, 2019] and DGC [Lin *et al.*, 2018] in non-IID environments. Specifically, we have trained a modified version of *LeNet5* [LeCun *et al.*, 1998] network on FMNIST [Xiao *et al.*, 2017] and a modified *VGG16* [Simonyan and Zisserman, 2014] network network on *CIFAR10* [Krizhevsky *et al.*, 2009] respectively. In addition, the first two CNN blocks of VGG16 and the first one CNN block of modified LeNet5 are set to the shallow part in practice. All the batch-normalization layers are removed in the shallow model. The experiment settings are listed in Table 2.

4.1 Behavior Of The Model Performance

Figure 2(a) and Figure 2(b) show the top-1 accuracy of *LeNet-5* on FMNIST after 200 communication rounds and the accuracy of *VGG16* on *CIFAR10* after 2000 communication rounds respectively using H-FL and the aforementioned three methods. The magenta dotted line refers to an accuracy of 80%. The experiment results show that H-FL outperforms the other methods both on convergence rate and final accuracy. The results are quite reasonable since H-FL reconstructs a series of virtual distributions $p^{(m)}$s, each of which is more closer to the potential global distribution and the optimization problem under non-IID is almost turn into that under IID. Thus, H-FL have the better capability to handle the heterogeneous dataset. Specifically, we take the average of the last 10 rounds of the accuracy as the final accuracy after 200 rounds on FMNIST and 2000 rounds on CIFAR10 respectively. H-FL achieves an accuracy of 88.16% on FMNIST, whereas FedAVG, DGC and STC only achieve 82.28%, 82.00%, and 82.12% respectively. Moreover H-FL achieves an accuracy of 87.28% on CIFAR10, whereas FedAVG, DGC and STC only achieve 73.83%, 81.25% and 81.24%.



(a) Effectiveness of Bias Corrector (b) Communication Overhead On FMNIST (c) Communication Overhead On CIFAR10

Figure 3: Effectiveness of the Bias Corrector and Communication Overhead

4.2 Influence Of Different Parameters For H-FL

From Figure 2(c), Figure 2(d) and Figure 2(e), we can observe that as $\mathcal{P}$, $\mathcal{S}$ and $\mathcal{C}$ increase, the model performance and the convergence behavior are getting better. The phenomenon is quite reasonable because: 1) As aforementioned, the procedure of reconstructing distributions in H-FL is closely related to the training samples of each client. The larger $\mathcal{P}$ and $\mathcal{S}$ are, the more the training samples are, and the closer the reconstructed distribution is to the potential global distribution, thus reducing the impact of non-IID and obtaining a relatively ideal effect; 2) The number of training samples of each client has a great impact on noise injecting. As $\mathcal{S}$ increase, the number of training samples of each client becomes larger, so that the $n^{(c)}$ in Formula 8 will be larger and the injected noise is correspondingly small; 3) As $\mathcal{C}$ increases, the lossy compression becomes less and less effective and the behavior will get better. In addition, as the noise level σ increases in Figure 2(f), the oscillation amplitude of the accuracy curve becomes larger, the convergence speed becomes slower and the model performance becomes worse.

4.3 Effectiveness Of The Bias Corrector

Figure 3(a) shows the top-1 accuracy of *VGG16* on *CIFAR10* with and without the bias corrector. The black dotted line refers to an accuracy of 85%. As we can see, bias corrector has significant influence on the convergence behavior and the final accuracy of H-FL. When there exists the bias corrector, the accuracy of the global model converges to 85% around 1000 rounds. Whereas when we remove the bias corrector, the accuracy gradually approximates to 85% until 2000 rounds. Additionally, we take the average of the last 10 rounds of the accuracy as the final accuracy after 2000 rounds, and the bias corrector can achieve an improvement of 2.47 percentage points. The result of the experiment is in line with our expectation since the bias corrector gives a relatively precise approximation of $dW^{(c)}$ when it can't be calculated directly, and once we remove the bias corrector, it will obtain a biased $dW^{(c)}$, leading to the degradation in model performance and other metrics (convergence behavior).

4.4 Communication Overhead

Finally, we compare the different methods with respect to the communication overhead which are required to achieve a certain target accuracy. As we can see in the Figure 2(a) and Figure 2(b), the convergence behavior is much better than other methods, which considerably reduces the communication rounds. Notice that FedAVG does not converge on *CIFAR10*, thus we do not show that in Figure 3(c). To compare the communication overhead, we set a window of size 10, which is utilized to calculate an average of 10 rounds. The communication overhead accumulates while moving forward the window until the average accuracy is no less than the target accuracy (80% in our experiments). Figure 3(b) and Figure 3(c) show the communication overhead required to achieve the target accuracy for the different methods on *FMNIST* and *CIFAR10* respectively.

5 Conclusion

In this paper, we present a **H**ierarchical **F**ederated **L**earning architecture (H-FL) to find a great balance among model performance, communication overhead and privacy requirements. Firstly, we devise a runtime distribution reconstruction strategy to counter-weigh the degradation due to non-IID. Then we design a compression-correction mechanism to reduce the communication overhead without sacrificing the model performance. The experimental results have proved that H-FL achieves the state-of-the-art performance under different federated learning tasks.

Acknowledgements

We would like to thank anonymous reviewers for their helpful comments. This work was supported by the the National Key Research and Development Program of China (No. 2018AAA0100500), NSFC Grant No. 61872285, 62072367, 61772413, the Key Research and Development Program of Shaanxi Province (2020GY-033), Key Science and Technology Project of Henan Province (201300210400), Fundamental Research Funds for the Central Universities (xzy012020112).

References

[Abadi *et al.*, 2016] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 308–318, 2016.

[Kairouz *et al.*, 2019] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Keith Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *arXiv preprint arXiv:1912.04977*, 2019.

[Konečnỳ *et al.*, 2016] Jakub Konečnỳ, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016.

[Krizhevsky *et al.*, 2009] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. *Master's thesis, University of Tront*, 2009.

[LeCun *et al.*, 1998] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[Li *et al.*, 2020] Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3):50–60, 2020.

[Lin *et al.*, 2017] Yujun Lin, Song Han, Huizi Mao, Yu Wang, and William J Dally. Deep gradient compression: Reducing the communication bandwidth for distributed training. *arXiv preprint arXiv:1712.01887*, 2017.

[Lin *et al.*, 2018] Yujun Lin, Song Han, Huizi Mao, Yu Wang, and William J Dally. Deep Gradient Compression: Reducing the communication bandwidth for distributed training. In *The International Conference on Learning Representations*, 2018.

[McMahan *et al.*, 2017a] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, pages 1273–1282. PMLR, 2017.

[McMahan *et al.*, 2017b] H Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. Learning differentially private recurrent language models. *arXiv preprint arXiv:1710.06963*, 2017.

[Rothchild *et al.*, 2020] Daniel Rothchild, Ashwinee Panda, Enayat Ullah, Nikita Ivkin, Ion Stoica, Vladimir Braverman, Joseph Gonzalez, and Raman Arora. Fetchsgd: Communication-efficient federated learning with sketching. In *International Conference on Machine Learning*, pages 8253–8265. PMLR, 2020.

[Sattler *et al.*, 2019] Felix Sattler, Simon Wiedemann, Klaus-Robert Müller, and Wojciech Samek. Robust and communication-efficient federated learning from non-iid data. *IEEE transactions on neural networks and learning systems*, 2019.

[Simonyan and Zisserman, 2014] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[Vepakomma *et al.*, 2018] Praneeth Vepakomma, Otkrist Gupta, Tristan Swedish, and Ramesh Raskar. Split learning for health: Distributed deep learning without sharing raw patient data. *arXiv preprint arXiv:1812.00564*, 2018.

[Wei *et al.*, 2020] Kang Wei, Jun Li, Ming Ding, Chuan Ma, Howard H Yang, Farhad Farokhi, Shi Jin, Tony QS Quek, and H Vincent Poor. Federated learning with differential privacy: Algorithms and performance analysis. *IEEE Transactions on Information Forensics and Security*, 2020.

[Xiao *et al.*, 2017] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[Yang *et al.*, 2019] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(2):1–19, 2019.

[Zhang *et al.*, 2021] Qingsong Zhang, Bin Gu, Cheng Deng, and Heng Huang. Secure bilevel asynchronous vertical federated learning with backward updating. *arXiv preprint arXiv:2103.00958*, 2021.