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 hierarchical federated learning (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 hierarchical federated learning 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.
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.

### 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.

<table>
<thead>
<tr>
<th>Notation</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>$W_t^{(d)}$</td>
<td>global deep model at round $t$</td>
</tr>
<tr>
<td>$W_t^{(s)}$</td>
<td>global shallow model at round $t$</td>
</tr>
<tr>
<td>$W_t^{(c)}$</td>
<td>shallow model kept in client $c$ at round $t$</td>
</tr>
<tr>
<td>$W_t^{(m)}$</td>
<td>deep model kept in mediator $m$ at round $t$</td>
</tr>
<tr>
<td>$\mathcal{U}$</td>
<td>all the clients</td>
</tr>
<tr>
<td>$\mathcal{P}$</td>
<td>sampling probability of each client</td>
</tr>
<tr>
<td>$\mathcal{S}$</td>
<td>sampling probability of each example</td>
</tr>
<tr>
<td>$\mathcal{C}$</td>
<td>global compression ratio</td>
</tr>
<tr>
<td>$\mathcal{I}$</td>
<td>iterations of deep training</td>
</tr>
<tr>
<td>$\ell_2$</td>
<td>$\ell_2$-norm of the clipped gradients</td>
</tr>
<tr>
<td>$\sigma$</td>
<td>noise level</td>
</tr>
</tbody>
</table>

Table 1: Notations and Definitions
of collaboratively building a shared model; 3) they do not col-
lude 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 compo-
nents: shallow model and deep model, then distributes the
former one to the Aggregation Mediator (AML) and the latter
to the other Mediators (M). AML 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 P, sampling probability of each ex-
ample S, global compression ratio C (C < 0.5), iterations of
deep training in mediators I, ℓ2-norm of the clipped gradi-
ents 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 individ-
ual clients is collected by the clients themselves on the ba-
sis of their local environments, the distribution of the local
datasets will considerably differ with each other. Consider-
ing this characteristic, we redefine the optimization objective
function of federated learning training on non-IID datasets as follows:

$$\min_{w, p^{(c)}} E_{(x, y) \sim p^{(c)}} [\ell(f(x; w^{(c)}), y)] + \sum_{c} D_{KL}(p||p^{(c)})$$ (1)

where \(w^{(c)}\), \(p^{(c)}\), \(p\), \(D_{KL}\) 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 dis-
tributions 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 distribu-
tion \(p\), enabling the latter term in Formula (1) to be 0. How-
ever, sharing local data raises serious privacy risks and causes
high communication overhead. Therefore, we introduce the
runtime distribution reconstruction strategy to mitigate dif-
fferences 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 infor-
mation entropy \(H^{(c)}\) of its local distribution \(p^{(c)}\) and KL
divergence \(D_{KL}(p^{(r)} || p^{(c)})\) between \(p^{(r)}\) and \(p^{(c)}\). Fur-
thermore, K-means algorithm is utilized to cluster the clients ac-
cording to the binary group \((H^{(c)}, D_{KL}(p^{(r)} || p^{(c)}))\). Then
H-FL randomly selects clients from each cluster, marks them
as a group, and assigns the group to one of mediators. The al-
location pattern loops until all the clients are assigned to the
corresponding mediator.

Algorithm 1 Runtime distribution reconstruction

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

1: Randomly initialize a distribution \(p^{(r)}\)
2: for each \(c \in U\) in parallel do
3: Compute \(H^{(c)}, D_{KL}(p^{(r)} || p^{(c)})\)
4: end for
5: Cluster according to \((H^{(c)}, D_{KL}(p^{(r)} || p^{(c)}))\)
6: for each \(m \in M\) do
7: Randomly select clients from each cluster according to
the same ratio \(1/|M|\) and assign them to \(m\)
8: \(B^{(m)} \leftarrow \emptyset\)
9: end for
10: \(M' \leftarrow\) (Randomly sampling mediators in \(M\))
11: for each \(m \in M'\) do
12: \(U' \leftarrow\) (Randomly sampling clients in \(U\) with \(P\))
13: for each \(c \in U'\) do
14: Randomly sampling a mini-batch \(X^{(c)}\) with \(S\)
15: \(O^{(c)} \leftarrow W^{r}X^{(c)}\)
16: \(k \leftarrow |O^{(c)}| * C\)
17: \(B^{(m)} \leftarrow B^{(m)} \cup LF(O^{(c)})\)
18: end for
19: end for
20: 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 mediat-
or concatenates the received features through a connector
(as shown in Figure 1) to obtain synthetic features. This pro-
cedure can be considered as sampling from a virtual recon-
structed distribution \(p^{(m)}\) and then conducting forward propa-
gation using the shallow model (see Algorithm 1). Intuitively,
\(p^{(m)}\) will be more approximate to the potential global distri-
bution \(p\) than \(p^{(c)}\). The optimization objective function will
be changed to the following form:

$$\min_{W, p^{(m)}} E_{(x, y) \sim p^{(m)}} [\ell\left(f\left(x; W^{r}(c), W^{l}(m)\right), y\right)] + \sum_{m} D_{KL}\left(p||p^{(m)}\right)$$ (2)

Assuming that there exists enough clients, \(p^{(m)}\s\) will in-
finitely approximate the potential global distribution \(p\) and
the latter term will be 0, which is translated to the optimiza-
tion problem under IID. When finishing the distribution re-
construction, 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 het-
erogeneity 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:

$$LF(O) = U_O \cdot \Sigma_O \cdot V_O^T \cdot k$$

where $O$ is feature matrix extracted by the shallow model, $U_O$, $\Sigma_O$ and $V_O^T$ are the results of singular value decomposition (SVD) respectively, $U_O \cdot \Sigma_O \cdot V_O^T$ represent the first $k$ columns of $U_O \Sigma_O$ and $V_O$ 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:

$$O = W^{(c)} X^{(c)}$$

$$B = LF(O)$$

$$A = W^{(m)} B$$

$$L = E[\ell(A, y)]$$

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

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

However, according to formula (3), we cannot compute $\partial B/\partial W^{(c)}$ directly since there is no direct differentiable mapping from $W^{(c)}$ to $B$. For convenience, $\partial O/\partial W^{(c)}$ can be used instead of $\partial B/\partial W^{(c)}$, 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 $O$ to $B$ so that we can better approximate $\partial B/\partial W^{(c)}$ and counter-weigh the reduction. According to the feature of SVD, we can get:

$$B = U_O D_k U^T_O O$$

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

$$\partial B/\partial W^{(c)} \approx U_O D_k U^T_O \cdot (\partial O/\partial W^{(c)})$$

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:

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

where $g^{(c)}$ is $dW^{(c)}$ itself, $n^{(c)}$ is the size of the sampled mini-batch in client $c$, $N$ is the Gaussian distribution with mean $\theta$ and standard deviation $\sigma L^2 / \sqrt{n}$.

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

---

**Algorithm 2** The workflow for H-FL

**Input:** $U, AM, M$

**Parameter:** $W_t^{(m)}, W_t^{(c)}, P, S, C, I, L, \sigma$

**Output:** $W_{t+1}^{(d)}, W_{t+1}^{(s)}$

**Mediators:**

1: $B^{(m)} \leftarrow$ Run-time data augmentation
2: for each $m \in M \setminus AM$ in parallel do
3: for each epoch $i$ from 1 to $L$ do
4: $W_t^{(m)} \leftarrow W_t^{(m)} - \eta \nabla W_t^{(m)} B(m, y)$
5: end for
6: $dB^{(m)} \leftarrow \nabla B^{(m)} (W_t^{(m)} B(m, y))$
7: for each $c \in m$ do
8: $dB^{(c)} \leftarrow dB^{(m)} \cdot n^{(c)}$
9: $dB^{(m)} \leftarrow dB^{(m)} \cdot n^{(c)}$
10: end for
11: end for

**Clients:**

1: for each $c \in U'$ in parallel do
2: $B^{(c)} \leftarrow U_O^{(c)D_k^{(c)}} U_O^{(c)T} O^{(c)}$
3: $W_t^{(c)} \leftarrow dB^{(c)} dB^{(c)} / dW^{(c)}$
4: $W_t^{(c)} \leftarrow W_t^{(c)} + N \left(0, \sigma^2 L^2 I / n^{(c)}\right)$
5: $W_t^{(c)} \leftarrow W_t^{(c)} - \eta dB^{(c)}$
6: end for

**FL Server:**

1: $W_{t+1}^{(d)} \leftarrow \sum_{m \in M \setminus AM} W_t^{(m)}$
2: $W_{t+1}^{(s)} \leftarrow \sum_{c \in U'} W_t^{(c)}$

**AM:**

1: $W_{t+1}^{(s)} \leftarrow \sum_{c \in U'} W_t^{(c)}$

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

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

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

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:

$$N \left(0, \frac{\sigma^2 L^2 I}{n^{(c)}}\right) = \sum_{i=1}^{n^{(c)}} N \left(0, \frac{\sigma^2 L^2 I}{n^{(c)}}\right)$$
differential privacy parallel principle.

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

(11)

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%.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Clients</th>
<th>Mediators</th>
<th>(\eta)</th>
<th>classes per client</th>
<th>(L)</th>
<th>(L')</th>
</tr>
</thead>
<tbody>
<tr>
<td>CIFAR10</td>
<td>100</td>
<td>3</td>
<td>0.015</td>
<td>3</td>
<td>10</td>
<td>1</td>
</tr>
<tr>
<td>FMNIST</td>
<td>100</td>
<td>3</td>
<td>0.015</td>
<td>2</td>
<td>10</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 2: Experiment Settings
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 $P$, $S$ and $C$ 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 $P$ and $S$ 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 $S$ 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 $C$ increases, the lossy compression becomes less and less effective and the behavior will get better. In addition, as the noise level $\sigma$ 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 Hierarchical Federated Learning 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 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


[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.


