Selective Federated Learning: Trade-offs for Federated Learning for the Mobile Environment

Jiahan Liu, Seth Lee, and Christine Julien

The University of Texas at Austin
{jiahanliu, sethlee, c.julien}@utexas.edu

***Abstract*—Personal mobile devices and applications involving edge devices have become ubiquitous over the past few years and the amount data that these devices are collected have increased exponentially. This data has enormous potential for training machine learning models in many areas including healthcare, social media, and bioinformatics. At the same time, challenges have arisen from many directions such as data ownership, privacy, computational power, and data storage. Federated learning has been proposed to address these challenges, but needs more research in managing device availability, being robust against non independent and identically distributed (non-IID) data, and mobile device computational and data storage limitations.**

In this paper, we develop a novel selective federated learning paradigm to approach the engineering trade-offs of federated learning. First, we then deploy a federated learning onto heterogeneous devices, the Jetson Nano, Raspberry Pi 4, Macbook Pro to observe how more complex models are robust to non-IID data but face computational and memory limitations in these devices. We then set out to prove that applying federated learning over all the nodes does not necessarily improve model performance for an non-IID test set where a local loss maybe a better proxy than global loss for accuracy. Thus, we propose selective federated learning which performs federated learning over similar nodes to increase robustness to a non-IID dataset, reduces the engineering complexity of synchronizing device availability, isolates the information exchange in the federated learning process to fewer devices to increase privacy, and security.

***Index Terms*—Federated Learning, Edge Computing**

I. Introduction

The current paradigm for machine learning trains model(s) using data stored at a centralized location. The data, however, is often collected on personalized devices or edge devices. This introduces the challenges of data privacy, ownership, communication, and computation. Federated learning is a form of distributed machine learning first proposed by Konecny et al. in 2015[1] that seeks to tackle these challenges. Federated learning brings the training computation to the data instead of bringing the data to the computation and averages the model weights to create an aggregate model.

Federated learning works by optimizing weights locally on each node then aggregating the weights on a centralized server before updating all the local models with the new weights to start another round of training. We consider this one round of training between communications. Rounds of training continue until model convergence. Federated learning addresses of data ownership because the raw data never leave the device. While a backdoor to federated learning has been proposed by Bagdasaryan et al. [2], the paper exposes how the performance of the federated model can be attacked, and not how the privacy of the model can be compromised. There are currently no known methods to reverse engineer the data from the weights; hence federated learning is a potential paradigm that preserves data privacy. Federated learning is better than centralized learning in terms of communication bandwidth in the cases where the memory size of the weights is less than the memory size of the data. While the memory size of weights is fixed as a function of a model architecture, the amount of data collected by a node devices can vary greatly from application to application. While it's true that models can have millions or billions of parameters, raw data can be even larger; for example, lidar data for the city of Dublin is 0.5 terabytes [3]. Finally, as personalized and edge devices are getting more powerful and numerous, they can be leveraged to perform the computations required in federated learning.

An example application where federated learning has been successfully applied is Google's Gboard which uses federated learning to improve query suggestions by training on personal phone data without collecting the data itself in a centralized location [4]. Federated learning, however, still faces both systems and theoretical challenges. The systems challenges are currently bottlenecked by communication efficiency and has been researched by Konecny et al., 2017[5]. These communication challenges stem from the fact that the training time is dominated by the time between communication rounds rather than the computation time because participation of the devices in the aggregation step needs to be synchronized. The theoretical challenges involve the convergence of the federated model and robustness to non-IID data. While Xi et al. have proven the convergence of FedAvg on non-IID Data [6], Smith et al. has shown that federated models may underperform models trained using only local data [7]. However, in practice, nodes may not have enough data to train it's own local node; hence, the purpose of federated learning.

A. Contributions

We conduct experiments to observe trade-offs for federated learning. Our experiments explore the trade-off between model complexity and robustness of federated learning to non-IID data and lead to selective federated learning, a novel way to make the trade-offs for federated learning in the mobile environment. We also explore the trade-offs between applying federated learning over every nodes data vs federated learning over similar nodes. While applying federated learning to more nodes increases the amount of data for training, applying to nodes with non-IID datasets does not necessary improve the performance of the model since data quality is also important [8]. Applying federated learning over less nodes also impacts the engineering complexity in the real deployment of selective federated learning in terms of ameliorating the effects of device availability, and the communication overhead of lock-step federated averaging. Furthermore, by partitioning the nodes into groups, we isolate the data and communication to create robustness to federated learning attacks and increase privacy.

II. Background

We cover four key papers that lead up to the ideas present in this thesis.

A. Federated Learning of Deep Networks using Model Averaging [9]

McMahan's paper covers the FederatedAveraging algorithm which uses the weighted average of the model parameters as the aggregation step of federated learning. McMahan's paper shows empirically that FederatedAveraging is robust to non-iid data by showing in experiments which data was distributed non-IID and then tested on a single test set that spans examples from each node. This experiments in this thesis differs from the ones conducted in Federated Learning of Deep Neural Networks using Model Averaging because it creates test sets which are unbalanced as well and has greater variation in model complexity.

B. Federated Learning: Strategies for Improving Communication Efficiency [5]

Bonawitz's paper examines the engineering challenges associated with deploying a federated learning algorithm to a large number of devices. In practice, the network could be slow and the availability of the clients may be unreliable. The paper proposes two methods to improve communication costs, *structured updates* which systematically chooses a subset of devices to aggregate each round, and *sketched updates*, which uses quantization, random rotations, and subsampling of the local dataset before sending it to the server.

We implement a simpler version of the federated learning presented here. Our paper focuses on the performance aspect of federated learning in terms of model accuracy and we build a simple deployed model as a preliminary test for feasibility.

C. Federated Multi-Task Learning [7]

In the Federated Multi-Task Learning paper, Smith et al. proposes multi-task learning for federated learning on non-IID datasets. In multi-task federated learning, the data from the other nodes are incorporate indirectly as part of the loss function instead of directly by averaging the weights. Her team uses support vector machines to classify three unbalanced datasets, the Google Glass dataset, the Human Activity Recognition dataset, and the Vehicle Sensor dataset. In each dataset, the local model performs better than the global federated model and the multi-task learning model performs better than the local model.

Smith's paper led to the experiments for this thesis. It indirectly suggests that the test set should be distributed the same as the training set which the experiments form the FederatedAveraging did not do. This thesis builds upon Smith's research by exploring another method which train a model to perform better on non-IID test sets. This thesis also explores more model complexity by using multi-layer neural networks with convolutional networks rather than just support vector machines.

D. On the Converge of FedAvg on Non-IID Data [6]

The key concept of this paper shows that the the global loss converges under FederatedAveraging converges to a global minimum with the dominant variable being the number of iterations. Namely, FederatedAveraging converges at rate $O(\frac{1}{T}$ where T is the number of iterations.

This thesis builds upon the proof presented by Li et al. by showing that in the FederatedAveraging algorithm as even as the global loss converges to a minimum, the local loss at each node may not be the minimum.

III. Implementation

A. Dataset Division

In each experiment, we partition the MNIST training set into partitions to represent different degrees of non-IID (unbalanced). Then we assigned nodes to a unique partition and distributed the data in each partition to it's nodes. We performed two sets of experiments. For our unbalanced test experiments we partitioned the test set to create unbalanced test sets in the same fashion we created the training set. For our balanced tests, we used the global test set of 10,000 images provided by the original MNIST dataset (Balanced Tests)

Ours partitions can be distinguished by the percentage with which they contain balanced data. In table I the data partitions are described. We first remove 10,000 images from the original MNIST training set to create a validation set. The term random data refers to MNIST images which have been randomly assigned, this represents the balanced data.

TABLE I
Data Partition

	N% Balanced
Partition 1	N% Random Data (100-N)% Labels 0-3
Partition 2	N% Random Data (100-N)% Labels 4-6
Partition 3	N% Random Data (100-N)% Labels 7-9

For example, to create datasets for 9 nodes, we would divide partition 1 equally among nodes 1-3, divide partition 2 equally among nodes 4-6, and divide partition 3 equally among nodes 7-9. Likewise, the test set can be made non-IID.

B. Model Complexity

The amount of data needed to train a neural network depends on model complexity, and as neural networks get larger they need more data to train. Since we are varying the distribution of data, it is also important to take the effects of model complexity into consideration. We conduct our experiments on three models of different complexity. We would like to see the effects of this on federated learning with data distributed non-IID.

1) Single Layer ReLu Model: This is our simplest model. Fully Connected Neural Network with one hidden layer composed of thirty neurons. Each neuron uses the ReLu activation function. A output is generated using a log softmax.

2) Four Layer Convolutional and ReLu Model: This is our second most simplest model. A convoluted neural network with two 5x5 convolutional layers. Both convolutional layers have a max pooling layer. The second convolutional layer has a drop out of 0.5. The convolutional layers are followed by two fully connected layers. The first has 320 hidden neurons and the second has 50 hidden neurons and. The first hidden layer has a drop out of 0.5. The output is generated using log softmax.

3) Six Layer Extra Wide Convolutional and ReLu Model: This is our most complex model. The convlution block has three 3x3x2 convolutional layers of stride 1 and padding 1. Each convolutional layer is followed by a batch norm layer, a ReLu layer. The second and third layers have a max pooling layer. The convolutional layers are followed by three fully connect layers with 6272, 64, and 10 hidden neurons. Each fully connected layer is followed by a batch norm layer and has a drop out of 0.5. The output is generated using log softmax.

IV. Federated Convergence of Global vs Local Losses

For non-IID distributed dataset, it is reasonable to expect the test set should also be distributed non-IID with the same percentage of unbalanced data. For a non-IID test set, the local loss may be a better proxy for local accuracy than the global loss. In a federated model in which the global loss converges to a minimum, the local loss not be at the minimum for every node. We build upon the proof of the convergence of FederatedAveraging by Li et al. [6] to show this.

In proof of convergence of $FedAvg$ [6], Li assumes the following assumptions:

Assumption 1. All the subproblems, $F_1, ..., F_N$ are L-smooth.

Assumption 2. All subproblems, $F_1, ..., F_N$ are all μ-strongly convex.

Assumption 3. The variance of stocastical gradients in each device is bounded by σ_k^2.

Assumption 4. That the expected squared norm of stochastic gradients is uniformly bounded by G^2.

Let T be the total number of steps, F^* be the minimum value of F, p_k be the probablity of choosing partition k uniformaly sampled without replacement, E be the number of local iterations between two rounds of communication, $\kappa = \frac{L}{\mu}$, γ is $max\{8\kappa, E\}$ for learning rate chosen to be $n_t = \frac{2}{\mu(\gamma+t)}$. Li proved that:

$$\mathbf{E}[F(w_T)] - F^* \leqslant \frac{2\kappa}{\gamma + T}\left(\frac{B+C}{\mu} + 2L\|w_0 - w^*\|^2\right)$$

$$where$$

$$B = \sum_{k=1}^{N} p_k^2 \sigma_k^2 + 6L\Gamma + 8(E-1)^2 G^2$$

$$and$$

$$C = \frac{4}{K} E^2 G^2$$

For an arbitrary fixed model and dataset, the only variable in the root convergence rate is T, and as T approaches to infinity we have $E[F(w_T)] - F^* \leqslant 0$. Now, let's use Li's proof on the convergence of $FedAvg$ to prove theorem 1.

Theorem 1: **Worse Case Local Loss for Convergent Federated Model**

Arbitrarily pick any federated learning objective $\min_{w \in \mathbb{R}^d} F(w)$ satisfyingly assumptions 1-4 that uses $FedAvg$ as the aggregation step, any unacceptable loss constant multiplier C, and any number of training data for some node s. We want to show there exist a dataset P distributed into partitions P_k such that as $\mathbf{E}[F(w_T)]$ converges to F^*, $F_s(w_T)$ converges to $C \cdot F_s^*$. Let N be the number of partitions greater than 1.

Proof is provided in Appendix A.

V. Selective Federated Averaging

Here we proposed selective federated averaging to boost the performance of simple federated learning non-IID data. In extremely non-IID data, complex models do well in the federated setting. Local models also do well cannot be trained if there is insufficient data on each node. Selective federated averaging partitions the set of nodes N into k groups and then create k different models for each of the groups. The algorithm is described in figure 1. In selective federated averaging, each of the local nodes first trains their own local model. In the selective grouping process, the local models uses the validation set $V = \{v_1, v_2, v_3 ...\}$ to generate a selection vector by the selection function given in figure 1. For every pair of nodes, a, b are in the same group if the percentage of overlap in their selection vector meets a similarity threshold. The similarity threshold is calculated as a hyperparameter using a separate validation set. In our experiments, we split the validation set in half. Finally, a separate federated model is trained using federated average on all the nodes in each group. The end result is that there is a model for every group.

Selection Function S(validation set V, model M)

$$s(v_k) = \begin{cases} s_k = 0 & \text{M's prediction for } v_k \text{ is incorrect} \\ s_k = 1 & \text{M's prediction for } v_k \text{ is correct} \end{cases}$$

Fig. 1. Selection Function

Algorithm 1: Selective Federated Averaging

```
Train local models for all nodes
for each node $i \in |N|$ do
    request central server for validation set
      $V = \{v_1, v_2, v_3...\}$
    form selection vector $S(V, m_i)$
end
initialize set of ungrouped nodes U = N
initialize set of groups G = $\varnothing$
initialized added = False
for each node $n_i \in |U|$ do
    added = False
    if $G == \varnothing$ then
        create group g = $\{n_i\}$
        insert g into G
        continue
    end
    for each group $g_j \in G$ do
        if $(S_i \cdot S_k)/|V| <$
          similarity threshold for all nodes $k$ in $g_j$ then
            insert node i into $g_j$
            added = True
        end
    end
    if False == added then
        create group g = {node i}
        insert g into G
    end
end
Train federated model for all group
```

VI. Time Complexity Analysis

Time complexity in Federated Learning is dominated by communication time [5]. For federated learning model that requires N rounds of communication, selective federated averaging requires N+1 rounds of communication. The extra round is required communicate the validation vector to the central server for group selection.

VII. Experimental Results

We perform two sets of experiments. The first calculates test accuracy with the 10,000 image test set from MNIST to represent the performance of a model exposed to all a test set of classes during inference time. The second calculates test accuracy with unbalanced partitioned test set as described in section 3.2.1. In each set of experiments, we test all three models from section 3.4. We then display the accuracy on the graph represents in which the x-axis is the percentage of balanced data in the training set. We also provide the centralized accuracy for each model architecture as a benchmark.

A key finding in our experiments that was not found in previous research is that robustness of federated learning models requires model of certain depth or width. In the Federated Learning of Deep Networks using Model Averaging paper [9], the smallest model tested was a 2 layer hidden neural network that was 200 neurons wide. We found that a 1 layer network with 30 neurons wide can achieve 96.40% accuracy in the centralized setting but only 33.21% accuracy in the federated setting. There is, however, a tradeoff between model complexity vs computation, storage and power constraints of an edge device. The Six Layer Extra Wide Convolutional and ReLu Model in the federated setting, which is robust to a completely unbalanced data distribution, cannot run on the Raspberry Pi 4. There are many embedded systems which run slow CPU's and have less memory than a Raspberry Pi 4. Thus we found edge device capabilities to be a constraint.

A. Experiments: Test Set of Original 10,000 MNIST Images

In this set experiments, ordinary federated averaging had the best performance. Wider or deeper models performed better in the federated setting when given unbalanced data. Six Layer Extra Wide Convolutional and ReLu Model Accuracy for IID Test Set is almost completely robust to non-balanced data. In our experiments, models of all complexity perform poorly if trained locally with 0% balanced data. Models of all complexity cannot learn to recognize images that it does not see. Selective federated averaging performs poorly if it's tested on a global dataset because it only creates federated models for similar nodes whose collective dataset is unbalanced. For federated, selective federated and local models, performance increases rapidly as the amount of balanced data increases. For federated learning models trained on 0% balanced data, the performance increases from 32.66% accuracy to 84.32% accuracy to 97.31% accuracy as the model's width and depth increases. Figures 5, 6, 7 show the test accuracy vs percentage of unbalanced data in the training set.

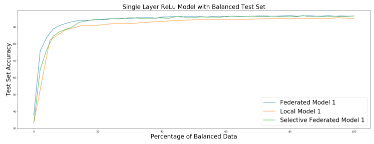

Fig. 2. Single Layer ReLu Model Accuracy for IID Test Set

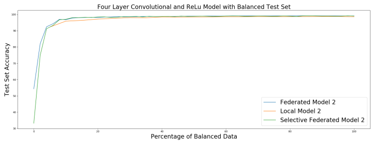

Fig. 3. Four Layer Convolutional and ReLu Model Accuracy for IID Test Set

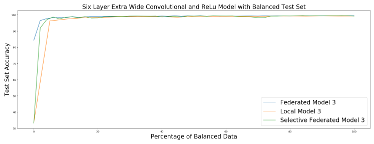

Fig. 4. Six Layer Extra Wide Convolutional and ReLu Model Accuracy for IID Test Set

B. Experiments: Test Set from N% Balanced Paritioned

In these experiments, we distribute the test set data so that each test set contained the same percentage of unbalanced data as the training set. This is an important test not covered in the Federated Learning of Deep Networks using Model Averaging paper [9] because if an edge device encounters unbalanced data during training time, it is reasonable to expect the edge device to encounter unbalanced data during testing time. Simpler models using selective federated averaging a model perform better than simple models using federated averaging. Selective federated averaging has the performance of local models but since it aggregates multiple node's datasets together, it is more likely to have sufficient for training.

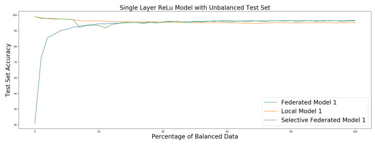

Fig. 5. Single Layer ReLu Model Accuracy for non-IID Test Set

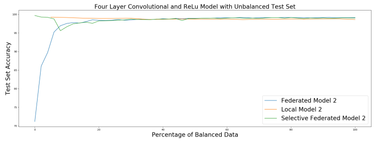

Fig. 6. Four Layer Convolutional and ReLu Model Accuracy for non-IID Test Set

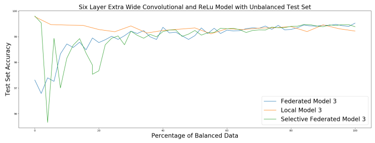

Fig. 7. Six Layer Extra Wide Convolutional and ReLu Model Accuracy for non-IID Test Set

C. Centralized Learning Baseline

These are the baselines for each of the models trained the entire training set.

TABLE II
CENTRALIZED TRAINING TEST ACCURACIES

Model	Accuracy (percent)
1	96.40
2	99.12
3	99.54

VIII. TRADE-OFFS

The ability to aggregate over a smaller set of nodes introduces a set of trade-offs for federated learning.

A. Model Complexity vs Robustness to Non-IID

Our results show that robustness of federated learning models requires model of certain depth or width. In the Federated Learning of Deep Networks using Model Averaging paper [9], the smallest model tested was a 2 layer hidden neural network that was 200 neurons wide. We found that a 1 layer network with 30 neurons wide can achieve 96.40% accuracy in the centralized setting but only 33.21% accuracy in the federated setting. Selective federated learning groups similar nodes together so that simpler models can be used.

B. Amount of Data Available vs Engineering Complexity

Not all data is equally useful [8]. Some nodes may be located such that it never encounters inputs that other nodes receive or even worse the outputs may conflict. Thus by grouping similar nodes together we reduce the total amount of data available and only select groups which are similar. We choose the groups of enough size to have sufficient data for a model and possibly improve performance for non-IID datasets. In doing so we also reduce the complexity of the distributed networking in federated learning. The challenges of device availability, unreliable device connectivity, and lock step execution first stated in Konecny's paper [5] can now be ameliorated by grouping the nodes using the selective federated learning grouping process.

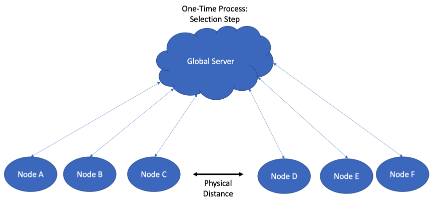

Fig. 8. One Time Step

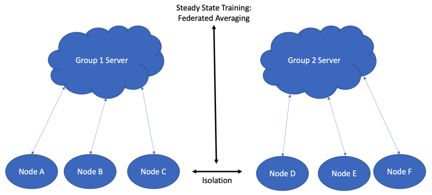

Fig. 9. Steady State Training

In figure 8, the selection step communication is only performed one time. The steady state training steps are isolated so that communication only occurs within the group. By using selective federated learning, we can choose smaller group sizes we can fit the constraints of distributed systems engineering without compromising model performance in both IID and non-IID data.

C. Size of Groups vs Security and Privacy

By performing federated learning over a smaller group, we can the number of edges in the communication graph of each federated model. An attack using the backdoor method proposed by Bagdasaryan et al. [2] would only effect the model which was the malicious node is grouped into. Any reverse engineering of the weights can also only discover data associated with the group of nodes.

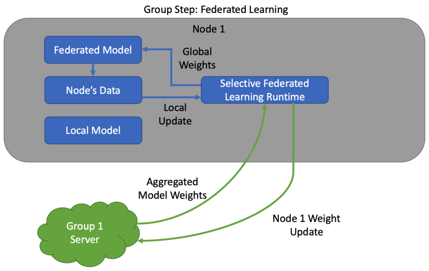

Fig. 10. Global Step of Selective Federated Averaging

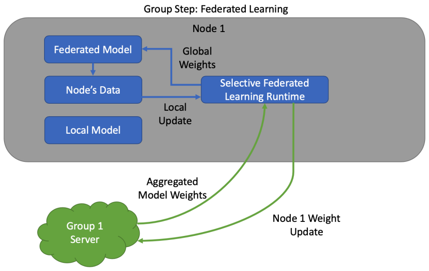

Fig. 11. Local Step of Selective Federated Averaging

Show in figure 10, the global step of federated learning does not use the Node's Data so that it is the data is never exposed the all the nodes. Meanwhile, in figure 11, the local step of federated learning uses the Node's data to produce local updates but this information is only communicated through the group server. Thus, we can control the trade-off between the number of nodes we choose to federated over to achieve a model accuracy and the exposure the data to attacks. This heighten control of ownership can be useful in healthcare and corporations where companies may choose to share data with some companies but not others.

IX. Conclusion and Future Work

In this paper we examine theoretically and experimentally the effects non-IID training data on Federated Learning with both IID and non-IID test sets. We found that model complexity is an significant constraint in federated learning performance. In the federated setting, however, there is a tradeoff between model complexity and hardware constraints. In our experiments, we find that the model that performs the best in the federated setting cannot be deployed to the Raspberry Pi 4, a common edge device. Additionally for non-IID test sets, we propose selective federated averaging, which is more robust to non-IID data than federated averaging.

This paper calls for an more indepth examination of the relationship between complexity of a model and the model's robustness to non-IID data in the federated setting.

References

1 Konecný, J., McMahan, B., and Ramage, D., "Federated optimization: Distributed optimization beyond the datacenter," *CoRR*, vol. abs/1511.03575, 2015. [Online]. Available: http://arxiv.org/abs/1511.03575

2 Bagdasaryan, E., Veit, A., Hua, Y., Estrin, D., and Shmatikov, V., "How to backdoor federated learning," *CoRR*, vol. abs/1807.00459, 2018. [Online]. Available: http://arxiv.org/abs/1807.00459

3 Cao, V., Chu, K., Le-Khac, N., Kechadi, M., Laefer, D., and Truong-Hong, L., "Toward a new approach for massive lidar data processing," in *2015 2nd IEEE International Conference on Spatial Data Mining and Geographical Knowledge Services (ICSDM)*, July 2015, pp. 135–140.

4 Yang, T., Andrew, G., Eichner, H., Sun, H., Li, W., Kong, N., Ramage, D., and Beaufays, F., "Applied federated learning: Improving google keyboard query suggestions," *CoRR*, vol. abs/1812.02903, 2018. [Online]. Available: http://arxiv.org/abs/1812.02903

5 Konecný, J., McMahan, H. B., Yu, F. X., Richtárik, P., Suresh, A. T., and Bacon, D., "Federated learning: Strategies for improving communication efficiency," *CoRR*, vol. abs/1610.05492, 2016. [Online]. Available: http://arxiv.org/abs/1610.05492

6 Li, X., Huang, K., Yang, W., Wang, S., and Zhang, Z., "On the convergence of fedavg on non-iid data," 2019.

7 Smith, V., Chiang, C., Sanjabi, M., and Talwalkar, A., "Federated multi-task learning," *CoRR*, vol. abs/1705.10467, 2017. [Online]. Available: http://arxiv.org/abs/1705.10467

8 Sessions, V. and Valtorta, M., "The effects of data quality on machine learning algorithms," in *ICIQ*, 2006.

9 McMahan, H. B., Moore, E., Ramage, D., and y Arcas, B. A., "Federated learning of deep networks using model averaging," *CoRR*, vol. abs/1602.05629, 2016. [Online]. Available: http://arxiv.org/abs/1602.05629

Appendix

A. *Proof of Worse Case Loss for Convergent Federated Model*

Proof: Construct dataset P with the following properties. Without loss of generality, let node s to be the last of the nodes, node N. Populate nodes 1 to $N-1$ such that $F_k(w_T)$ converges to F_k^* for $1 \leqslant k \leqslant (N-1)$ or in other words $F(w)$ is a good model for nodes 1 to $N-1$. Choose the degree of non-iid, Γ, and and the probability of choosing from node N, p_N, such that $\frac{\Gamma}{p_N}$ = $(C-1) \cdot F_N^*$. In other words, we can increase the amount of unacceptable loss on node N by increasing the degree of non-iid or increasing the number of data points on the other nodes to decrease p_N.

To show that the loss on node N converges to $C \cdot F_s^*$, we must be able to find T for any $\epsilon > 0$ that satisfies $|F_N(w_T) - C \cdot F^*| \leqslant \epsilon$. The proof of the convergence of $FedAvg$ by Li et al. shows $\mathbf{E}[F(w_T)] - F^*$ converges at rate $O(\frac{1}{T})$, so we choose T such that $|\mathbf{E}[F(w_T)] - F^*| \leqslant \frac{\epsilon}{2}$ and $|F_k(w_T) - F_k^*| \leqslant \frac{\epsilon \cdot p_N}{2(N-1)}$ for $1 \leqslant k \leqslant (N-1)$.

$$\begin{aligned}
\mathbf{E}[F(w_T)] - F^* &\leqslant \frac{\epsilon}{2} \\
\mathbf{E}[F(w_T)] &\leqslant F^* + \frac{\epsilon}{2} \\
\mathbf{E}[F(w_T)] &\leqslant \Gamma + \sum_{k=1}^{N} p_k F_k^* + \frac{\epsilon}{2} \\
\sum_{k=1}^{N} p_k F_k(w_T) &\leqslant \Gamma + \sum_{k=1}^{N} p_k F m_k^* + \frac{\epsilon}{2} \\
\sum_{k=1}^{N} p_k F_k(w_T) - \sum_{k=1}^{N} p_k F_k^* &\leqslant \Gamma + \frac{\epsilon}{2} \\
\sum_{k=1}^{N} (p_k(F_k(w_T) - F_k^*)) &\leqslant \Gamma + \frac{\epsilon}{2} \\
\sum_{k=1}^{N-1} (p_k(F_k(w_T) - F_k^*)) + p_N(F_N(w_T) - F_N^*) &\leqslant \Gamma + \frac{\epsilon}{2} \\
\sum_{k=1}^{N-1} \left(p_k \left(\frac{-\epsilon \cdot p_N}{2(N-1)} \right) \right) + p_N(F_N(w_T) - F_N^*) &\leqslant \Gamma + \frac{\epsilon}{2} \\
p_N(F_N(w_T) - F_N^*) &\leqslant \Gamma + \frac{\epsilon}{2} + \sum_{k=1}^{N-1} \left(p_k \left(\frac{\epsilon \cdot p_N}{2(N-1)} \right) \right) \\
F_N(w_T) - F_N^* &\leqslant \frac{\Gamma}{p_N} + \frac{\epsilon}{2} + \sum_{k=1}^{N-1} \left(p_k \left(\frac{\epsilon}{2(N-1)} \right) \right) \\
F_N(w_T) - F_N^* &\leqslant (C-1) F_N^* + \frac{\epsilon}{2} + \sum_{k=1}^{N-1} \left(p_k \left(\frac{\epsilon}{2(N-1)} \right) \right) \\
F_N(w_T) - C \cdot F^* &\leqslant \frac{\epsilon}{2} + \sum_{k=1}^{N-1} \left(p_k \left(\frac{\epsilon}{2(N-1)} \right) \right) \\
F_N(w_T) - C \cdot F^* &\leqslant \epsilon & p_k \leqslant 1
\end{aligned}$$

The same can be done to show that $F_N(w_T) - C \cdot F^* \geqslant -\epsilon$, hence we have $|F_N(w_T) - C \cdot F^*| \leqslant \epsilon$. In other words the loss at Node N converges to $C \cdot F_s*^*$, a loss that is unacceptable magnitude times larger than the acceptable loss observed at the central server. ■