Critique:
ScaleFL: Resource-Adaptive Federated Learning with Heterogeneous Clients

Archit Sengupta: 904013900

October 2024

1 Problems Addressed

This paper addresses the challenges caused because of heterogeneous client resources in Federated Learning (FL) systems. Devices can have different levels of computational capabilities, power and resources. Normal FL systems assume that all the clients can train deep neural networks (DNNs) of the same size. However in real-world deployments this is not true. Nowadays a lot of low power IOT devices are used for FL. All clients should be treated differently. Many clients may have very limited resources, making it difficult for them to participate in training large models. There have been several techniques used to deal with this. The key problems tackled in this paper are:

- **Resource Differences between clients**: As discussed before clients with differing computational, storage, and network capabilities struggle to participate in FL tasks involving large models. This is the biggest problem addressed.
- **Efficient model scaling**: Existing methods, such as gradient compression or model pruning, can't handle resource limitations properly. These techniques are not flexible and don't adapt well across different levels of constraints (computation, memory, bandwidth).
- **Knowledge transfer among clients**: It is important to make sure that models which are trained on low-resource devices still contribute well. This is a big challenge. The smaller models should transfer knowledge without losing too much information during aggregation.

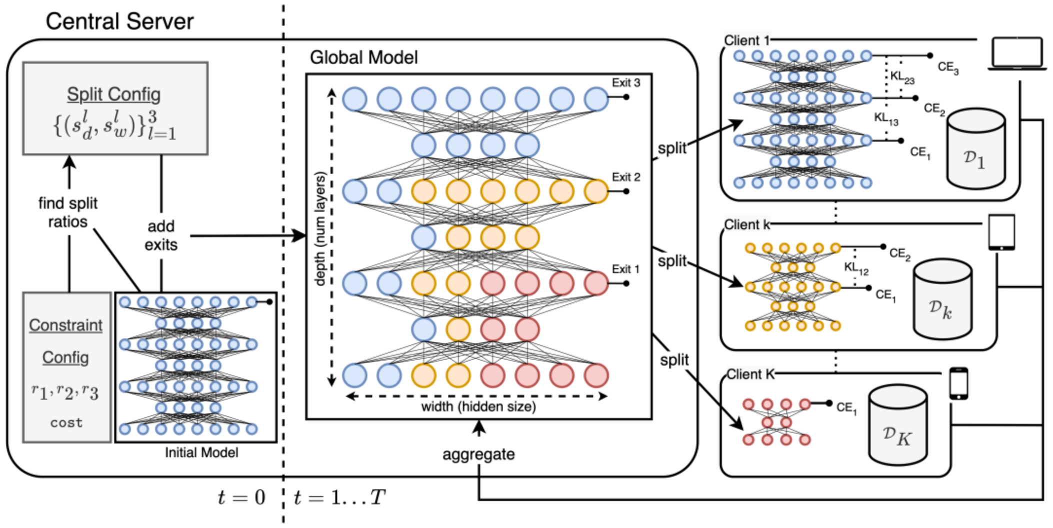


Figure 1: Architecture of Scale FL

2 Strengths of This Paper

This paper introduces ScaleFL, a novel adaptive FL framework which has been designed to deal with the problems created because of resource heterogeneity in federated learning. The methodology involves balancing model complexity with resource limitations in the heterogeneous clients.

2.1 Adaptive Two-Dimensional Model Scaling

The biggest strength of ScaleFL is to scale the global DNN model adaptively. This is done across both the width (number of neurons per layer) and depth (number of layers). The available computational resources of each client are studied and then accordingly the model is scaled. While some other approaches like HeteroFL, which scale models by adjusting only the width, ScaleFL improves on this. Both low-level features (handled by wider models) and high-level features (handled by deeper models) are preserved in smaller models. Early exit points DNN architecture have also been used. This is a very effective method which allows early termination of training and inference. So it is much more efficient and can be used to improve performance low-resource devices. This means minimal computational load, but the key information is preserved.

The method achieves a balance between feature extraction and model efficiency. It has considerable advantages over one-dimensional scaling methods

2.2 Self-Distillation and Aggregation Mechanism

An important innovation in ScaleFL is self-distillation to improve knowledge transfer among models of different complexities. Self-distillation is used in the training process by having earlier exits (smaller models) learn from the final exit (larger model). This has been done through KL divergence minimization. The knowledge is distilled from the larger model to smaller subnetworks. So there is effective learning at every complexity level. This means that additional external datasets during the distillation process are not needed. This is more efficient than other approaches like FedDF, which rely on ensemble distillation and have a lot of computational overhead.

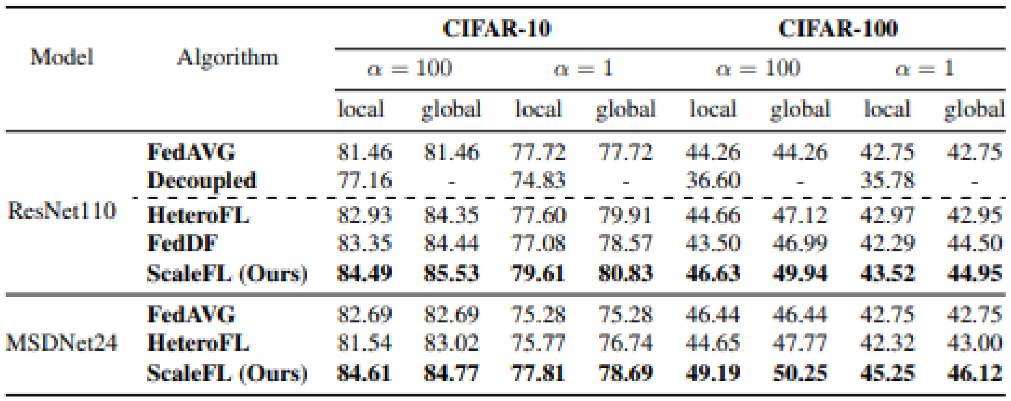
The aggregation of local updates is also improved by using overlapping model weights based on split indices. Clients operating on different submodels update different parts of the global model according to their resources and capability. So the gradient aggregation is more effective. There is not too much loss of information across different model complexities.

2.3 Experimental Validation for Improvements

A large number of experiments were conducted to prove the improvements through ScaleFL. Some of the results include:

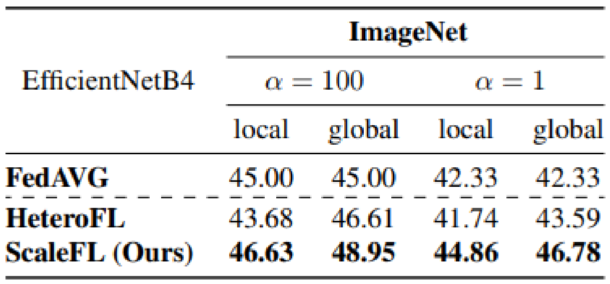
- Image classification: ScaleFL was tested on CIFAR-10, CIFAR-100, and ImageNet datasets. It was consistently better than baseline FL methods like FedAVG, HeteroFL, and FedDF.
- Natural language processing: The framework was tested using SST-2 and AgNews datasets. The test results as shown below proves application to non-vision tasks also.
- Higher accuracy: ScaleFL gets up to 3.19% higher global model accuracy compared to SOTA methods. This was tested under different distribution settings.

Several test results have been shown below.



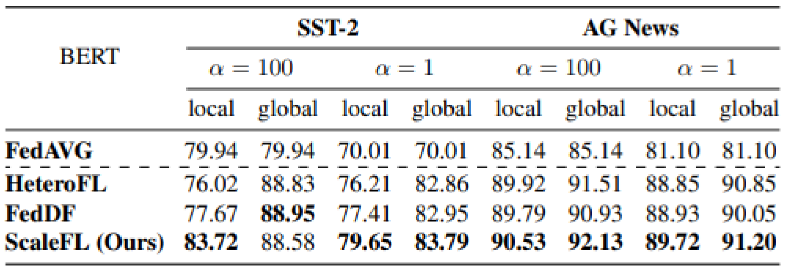
Model	Algorithm	CIFAR-10				CIFAR-100			
		$\alpha = 100$		$\alpha = 1$		$\alpha = 100$		$\alpha = 1$	
		local	global	local	global	local	global	local	global
ResNet110	**FedAVG**	81.46	81.46	77.72	77.72	44.26	44.26	42.75	42.75
	Decoupled	77.16	-	74.83	-	36.60	-	35.78	-
	HeteroFL	82.93	84.35	77.60	79.91	44.66	47.12	42.97	42.95
	FedDF	83.35	84.44	77.08	78.57	43.50	46.99	42.29	44.50
	ScaleFL (Ours)	**84.49**	**85.53**	**79.61**	**80.83**	**46.63**	**49.94**	**43.52**	**44.95**
MSDNet24	**FedAVG**	82.69	82.69	75.28	75.28	46.44	46.44	42.75	42.75
	HeteroFL	81.54	83.02	75.77	76.74	44.65	47.77	42.32	43.00
	ScaleFL (Ours)	**84.61**	**84.77**	**77.81**	**78.69**	**49.19**	**50.25**	**45.25**	**46.12**

Figure 2: Local and global accuracy tests on CIFAR



EfficientNetB4	ImageNet			
	$\alpha = 100$		$\alpha = 1$	
	local	global	local	global
FedAVG	45.00	45.00	42.33	42.33
HeteroFL	43.68	46.61	41.74	43.59
ScaleFL (Ours)	**46.63**	**48.95**	**44.86**	**46.78**

Figure 3: Local and global accuracy tests on ImageNet



BERT	SST-2				AG News			
	$\alpha = 100$		$\alpha = 1$		$\alpha = 100$		$\alpha = 1$	
	local	global	local	global	local	global	local	global
FedAVG	79.94	79.94	70.01	70.01	85.14	85.14	81.10	81.10
HeteroFL	76.02	88.83	76.21	82.86	89.92	91.51	88.85	90.85
FedDF	77.67	**88.95**	77.41	82.95	89.79	90.93	88.93	90.05
ScaleFL (Ours)	**83.72**	88.58	**79.65**	**83.79**	**90.53**	**92.13**	**89.72**	**91.20**

Figure 4: Local and global accuracy tests on SST-2 and AG News

2.4 Resource Efficiency and Scalability

ScaleFL shows very high resource efficiency. Clients with very different computational resources can contribute properly to federated learning through it. Improvements include:

- Reduction in Inference Times: There is 2x faster inference compared to baseline FL methods. This is through the use of early exits and model scaling.
- Reduction in model size: There are 4x smaller model sizes. So it's ideal for clients with low memory or processing power.
- Scalability: These improvements make ScaleFL very scalable. It is easy to deploy in real-world systems with heterogeneous client resources.

2.5 Conclusion

In conclusion, the paper offers several novel technical advancements to Federated Learning.

- Splitting model in 2 dimensions: A novel method for splitting DNN models along both width and depth dimensions. This maintains balance between basic and complex features.

- Self-distillation during training: Adding self-distillation among subnetworks improves the performance of smaller models. Additional overhead is also minimized.
- Equitable participation: ScaleFL allows equal participation, from clients, even from those with limited resources. So even clients with the most constrained and low resources can contribute to the global model.

3 Weaknesses

There are certain limitations which could have been improved to make the paper more complete.

- Early Exits for Inference Only: The early exit for neural networks is used for improving inference efficiency. This could also have been used in training itself, to provide improvements in that area.
- The framework assumes that the server has an accurate understanding of each client's computational resources and constraints. In the real-world, this information may be difficult and challenging to collect, leading to suboptimal model allocation.
- Overhead from Subnetwork Aggregation: The self-distillation method improves the learning process among subnetworks but, it also adds more aggregation overhead. The extra processing could slow down the federated learning rounds in large-scale implementations with many clients. Although the improvements are massive this needs to be taken into consideration.
- Independent effect of different techniques: It might be better to understand the effect of each technique in ScaleFL on its own. (e.g., the effect of self-distillation, the importance of two-dimensional splitting could be studied separately). Tests could have been conducted accordingly

4 References

1. ScaleFL: Resource-Adaptive Federated Learning With Heterogeneous Clients
   `https://paperswithcode.com/paper/scalefl-resource-adaptive-federated-learning`

Critique:
Communication-Efficient Learning of Deep Networks from Decentralized Data

Archit Sengupta: 904013900

October 2024

1 Introduction

This papper introduces the concept of Federated Learning (FL). As discussed in the previous critique it is a novel framework. In federated learning we train machine learning models on decentralized data. This data is mostly stored on mobile devices (including IOT devices) . The main idea is to avoid transferring raw data from devices to a central server for training. This can solve several issues, including those related to privacy and communication costs.

2 Problems Addressed

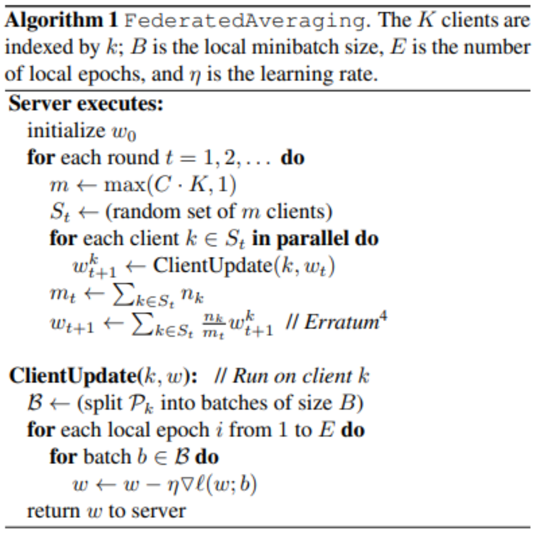
The researchers in the paper point out many interesting challenges which occur when someone trains models using data distributed and split between many mobile devices.

- **Communication Costs**: In decentralized setups with mobile devices, bandwidth is often limited. So the cost of communication with the server is a lot more than the cost of a centralized data center environment. That is why reducing the number of communication rounds needed to train the models is important
- **Non-IID Data**: The data across the devices in the federated learning is non-IID (non-independent and identically distributed). This is because different users generate data in different ways. It depends on the personal usage patterns. So a challenge is created. The convergence and performance of the models using distributed optimization techniques.
- **Unbalanced Datasets**: Due to various levels of device usage, the amount of data available on different devices is often unbalanced. Some devices having significantly more data than others. In the traditional distributed learning techniques we assume that there are balanced datasets. So these techniques are not good for this setting.

3 Proposed Solution

The paper has introduced a practical solution to these problems through the FederatedAveraging(FedAvg) algorithm. This algorithm involves local stochastic gradient descent (SGD) updates and then a model-averaging procedure which is carried out on a central server. The main contributions of the paper are:

- **Decentralized Model Training**: All the raw data is not sent to a central server. Each device performs local updates to a global model. These local updates are sent to the server, the server averages these and then updates the global model.
- **Reduction in Communication Rounds**: One of the main advantages of this technique is that there is a big reduction in the number of communication rounds needed for model convergence. There is more local computation on each device. So the algorithm reduces the number of communication rounds by 10-100x as compared to the conventional approach of synchronized stochastic gradient descent.
- **Handling Non-IID Data**: This algorithm is supposed to be less affected by the Non-IID nature of the data in federated learning. Several experiments are conducted to show the performance of the approach in scenarios where data distributions vary greatly between devices.
- **Unbalanced Data**: FedAvg is shown to be effective even when data is unbalanced across devices. It dynamically adjusts to different sizes of datasets different clients.



Algorithm 1 `FederatedAveraging`. The K clients are indexed by k; B is the local minibatch size, E is the number of local epochs, and η is the learning rate.

```
Server executes:
  initialize $w_0$
  for each round $t = 1, 2, \dots$ do
    $m \leftarrow \max(C \cdot K, 1)$
    $S_t \leftarrow$ (random set of $m$ clients)
    for each client $k \in S_t$ in parallel do
      $w_{t+1}^k \leftarrow$ ClientUpdate$(k, w_t)$
    $m_t \leftarrow \sum_{k \in S_t} n_k$
    $w_{t+1} \leftarrow \sum_{k \in S_t} \frac{n_k}{m_t} w_{t+1}^k$   // Erratum[4]

ClientUpdate$(k, w)$:   // Run on client $k$
  $\mathcal{B} \leftarrow$ (split $\mathcal{P}_k$ into batches of size $B$)
  for each local epoch $i$ from 1 to $E$ do
    for batch $b \in \mathcal{B}$ do
      $w \leftarrow w - \eta \nabla \ell(w; b)$
  return $w$ to server
```

Figure 1: Fedavg algorithm

4 Strengths of the Paper

The paper makes several important contributions and demonstrates a range of strengths:

- **Novelty and Impact**: The concept of Federated Learning is a novel approach and it can address privacy concerns in machine learning. Decentralizing the model training and focusing on communication efficiency is important and it has a lot of scope. The paper opens up new possibilities for privacy-preservation in machine learning on edge devices.

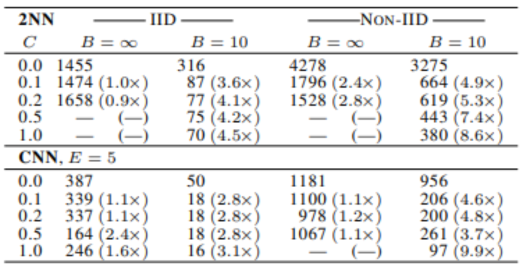
- **Empirical Evaluation**: There has been thorough empirical experimentation. The algorithm was tested on various datasets such as MNIST, CIFAR-10. Even a language modeling task using Shakespeare's works was tested. Five different model architectures were also included in the experiments to properly test the algorithm performance in different scenarios with non-IID. The experiments evaluate different neural network architectures, including convolutional networks and LSTMs.

- **Communication Efficiency**: Since federated learning in very reliant on communication(more so than compute), the number of rounds needs to be reduced. The aggregation in FedAvg allows more local computation between on rounds. Instead of gradients clients send updated model parameters only. This allows slower connections also because all clients don't need to be involved in every round. So the high cost of communication between devices and the server in federated learning is minimized. The paper demonstrates that adding local computation in each communication round can lead to dramatic reductions in communication costs. This is especially clear when smaller batch sizes are used or by increasing local computation epochs.

- **Privacy Considerations**: Although it is not the primary focus of the paper, there are considerable privacy advantages.

4.1 Results

Increased Parallelism and Computation: The algorithm scales well to a large number of devices. The researchers conducted an experiment to investigate how increasing parallelism affects performance. This is done by manipulating the client fraction C. C controls multi-client parallelism. They created learning curves for various combinations of parameters. The learning rate was optimized to make sure that results were accurate. Each curve showed monotonic improvement by selecting the highest test-set accuracy from the previous rounds. The number of communication rounds required was checked by finding where the curve intersected with the target accuracy using linear interpolation. Its also shown that by fixing C=0.1 and adding more local SGD updates per round can save on communication costs.



2NN	IID		NON-IID	
C	$B = \infty$	$B = 10$	$B = \infty$	$B = 10$
0.0	1455	316	4278	3275
0.1	1474 (1.0×)	87 (3.6×)	1796 (2.4×)	664 (4.9×)
0.2	1658 (0.9×)	77 (4.1×)	1528 (2.8×)	619 (5.3×)
0.5	— (—)	75 (4.2×)	— (—)	443 (7.4×)
1.0	— (—)	70 (4.5×)	— (—)	380 (8.6×)
CNN, $E = 5$				
0.0	387	50	1181	956
0.1	339 (1.1×)	18 (2.8×)	1100 (1.1×)	206 (4.6×)
0.2	337 (1.1×)	18 (2.8×)	978 (1.2×)	200 (4.8×)
0.5	164 (2.4×)	18 (2.8×)	1067 (1.1×)	261 (3.7×)
1.0	246 (1.6×)	16 (3.1×)	— (—)	97 (9.9×)

Figure 2: Results of the MNIST test

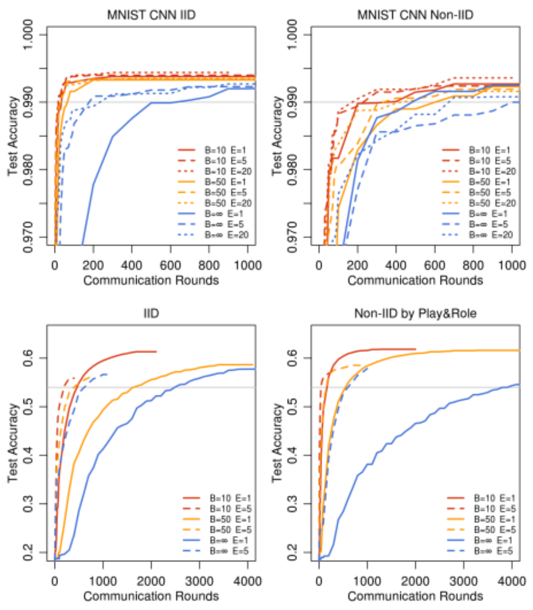
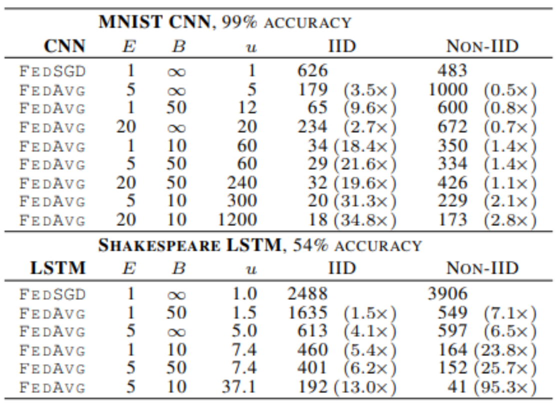


Figure 3: Accuracy as compared to communication rounds for MNIST CNN- IID, non-IID — Shakespeare LSTM- IID, Play&Role — C =0.1

- **MNIST**: On both IID and non-IID MNIST dataset, FedAvg gets a lot of reduction in communication rounds compared to traditional FedSGD. For example, FedAvg reduced the communication rounds needed to achieve 99% accuracy by up to 35x on IID data. In non-IID settings also, the algorithm performed well, but there was a smaller speedup.
- **CIFAR-10**: The CIFAR-10 dataset is more complex as compared to MNIST and involves larger models and more difficult classification task. In these experiments, FedAvg outperformed FedSGD. There was a speedup of up to 64.3x to reach 80% accuracy.
- **Language Modeling**: In the Shakespeare dataset for character-level language modeling, FedAvg performed really well on both IID and non-IID settings. The non-IID partitioning showed an even greater speedup (up to 95x) than the IID case.
- **Large-Scale LSTM**: There was a large-scale LSTM model for next-word prediction. FedAvg reduced the number of communication rounds by 23x compared to FedSGD but the acuracy was similar. This shows that the algorithm is scalable to the real world where there are millions of data points and thousands of clients.



MNIST CNN, 99% ACCURACY

CNN	E	B	u	IID	NON-IID
FEDSGD	1	∞	1	626	483
FEDAVG	5	∞	5	179 (3.5×)	1000 (0.5×)
FEDAVG	1	50	12	65 (9.6×)	600 (0.8×)
FEDAVG	20	∞	20	234 (2.7×)	672 (0.7×)
FEDAVG	1	10	60	34 (18.4×)	350 (1.4×)
FEDAVG	5	50	60	29 (21.6×)	334 (1.4×)
FEDAVG	20	50	240	32 (19.6×)	426 (1.1×)
FEDAVG	5	10	300	20 (31.3×)	229 (2.1×)
FEDAVG	20	10	1200	18 (34.8×)	173 (2.8×)

SHAKESPEARE LSTM, 54% ACCURACY

LSTM	E	B	u	IID	NON-IID
FEDSGD	1	∞	1.0	2488	3906
FEDAVG	1	50	1.5	1635 (1.5×)	549 (7.1×)
FEDAVG	5	∞	5.0	613 (4.1×)	597 (6.5×)
FEDAVG	1	10	7.4	460 (5.4×)	164 (23.8×)
FEDAVG	5	50	7.4	401 (6.2×)	152 (25.7×)
FEDAVG	5	10	37.1	192 (13.0×)	41 (95.3×)

Figure 4: Improvements using FedAvg on MNIST and LSTM

5 Weaknesses and Areas for Improvement

The paper offers many contributions, but it also has some limitations and areas where improvements could be made:

- **Privacy Guarantees**: Although the paper discusses the privacy benefits of federated learning, it does not provide formal guarantees. Including a better analysis of privacy mechanisms like secure multiparty computation or differential privacy can improve the security aspect of the framework.

- **Overfitting on Local Data**: Allowing too much local computation per client can lead to overfitting on local datasets. Because these datasets do not represent the global data , this is an issue. While the paper addresses this issue, it does not provide actual solutions to tackle it. Different techniques like regularization, early stopping, or gradient clipping can be used to avoid this overfitting.

- **Impact of Communication Constraints**: The paper assumes clients participate when they are on unmetered, stable connections. But in the real world, network conditions clients don't have reliable links all the time. The framework can be improved by handling intermittent connectivity and bottlenecks better.

- **Client Selection Strategy**: The methodology involves random client selection in each communication round. Better strategies can be used for this including data diversity or client availability. This can improve model convergence rates.

- **Security Considerations**: The paper does not deal with security issues like poisoning attacks. Malicious clients could intentionally send corrupted updates to the server.

6 References

1. Communication-Efficient Learning of Deep Networks from Decentralized Data
   https://arxiv.org/abs/1602.05629