# FedLR: A learning rate based approach towards Efficient communication

Vipul Singh Negi
Department of Computer Science and Engineering
National Institute of Technology
Rourkela, India 769008
vipulhld001@gmail.com

Suchismita Chinara
Department of Computer Science and Engineering
National Institute of Technology
Rourkela, India 769008
suchi.nitrkl@gmail.com

*Abstract*—The challenges of handling decentralised data lead to the demand for research on secure gathering, efficient processing, and analysing of the data. In decentralised systems, each node (device) can make independent decisions, reducing the complexity and challenges of dealing with extensive data. Privacy has become a significant concern for our society due to the rise in the number of Edge/IoT devices, the lack of presence of a centralised system, etc. To solve this conundrum, federated learning was proposed. Federated learning works on the sharing of parameter values rather than the data. Worldwide, 10.2 Billion non-IoT and 19.8 billion IoT devices will be active in 2023. These devices lack security when it comes to using traditional machine learning. However, federated learning models solve this problem using techniques such as Secure Aggregation and Differential Privacy, which provide security for the devices and efficient communication between them. The challenges arise from heterogeneous devices, leading to the client selection problem, unbalanced data, and many more problems. The Proposed work focuses on using the MobileNets series of model architecture for federated learning using the FedAvg Strategy. MobileNets architecture has always been robust and reliable when it come to devices with resource constraints. An older generation system is used to show that federated learning is a viable technique for decentralized machine learning.

*Index Terms*—Federated learning, IoT, Deep Learning, MobileNets

## I. INTRODUCTION

Federated Learning is born at the intersection of Edge computing/IoT, on-device AI, and blockchain. A Federation refers to a group of independent entities yet united under a central organization. In federated learning, multiple client or organisations share their training data (weights or compute) to remote servers, and all the clients participating in the process train a single neural network. This process is repeated by the clients, downloading the newer weights from the servers multiple times to provide better results. The training is done on the device's private data, then it is encrypted and communicated to the server, and on the server, they are decrypted, averaged, and integrated into the centralized model. The main objective of federated learning is to converge the client's weights so that it could yield meaningful results. For Example. WeBank (Banking), NVIDIA Clara (Healthcare), and Google Keyboard.

WeBank is a private Chinese bank they have created its own federated learning framework, known as WeBankAI (based on FATE) [1]. Nvidia Clara [2] is a platform to improve healthcare that focuses on [1] Medical Imaging and Medical Devices (Nvidia Clara Holoscan), Healthcare IoT (Nvidia Clara Guardian Collection), Biopharma (BioNeMo), and Genomics (Nividia Clara Parabricks). Google Keyboard (Gboard) [3] has been using federated learning for creating word prediction models.

Google introduced the term federated learning in 2016 (coined in 2017 by McMahan et al. [4]), about the same time the Cambridge Analytica scandal awakened users of the dangers of sharing personal information online. It started a revolution in the technology world about the three rules of Cryptography confidentiality, integrity, and availability. After a deeper review of our current laws, it was clear that we had none. So laws like GDPR and CCPA came into regulation. Federated Learning (Fig 1.) is one such method which can satisfy the rules of cryptography and privacy laws around the globe.
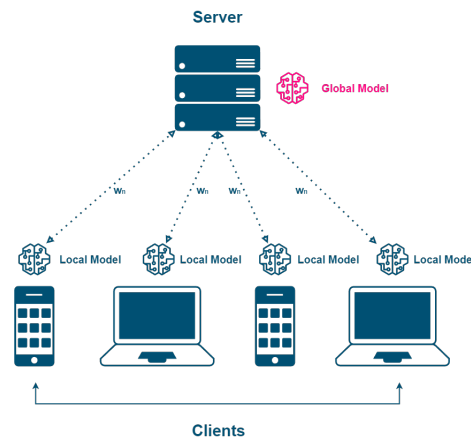


Fig. 1. Federated Learning

In this paper we present a new strategy which significantly addresses the problem with the old one keeping the effectiveness of federated learning alive. The main problem with the vanilla FedAvg is the client selection methodology which is taken at random and the devices participation have no threshold value to meet in order to participate in the federated

learning process. Another problem with FedAvg is to deal with device heterogeneity as each devices usually tend to have different CPU, RAM, and GPU which are the essential blocks of any neural network. The last major problem with the existing strategy is fairness. Fairness means all clients should be treated as fair and their contributions to the global model should carry equal weight. The strategy has been names as FedLR which is a dynamic learning rate and epoch based client strategy.

The proposed work focuses on reducing the total communication time taken by federated learning. Chen et al. [5] used RAM and CPU utilization data from clients in order to boost client selection. But getting such data from clients could lead to device fingerprinting [6] [7] which puts federated learning at risk. In this paper, by considering heterogeneous clients, we are employing FedLR, a learning rate based towards efficient communication. In the proposed methodology a time variable has been calculated and according to that time variable qualitative decisions towards learning rate selection has been made. FedLR guarantee faster communication time with fair selection of clients as it always consider slacker devices which makes this methodology Fair. The main contribution of the proposed work are summarized as follows:

1) By conducting extensive sets of experiments we came to a quantitative conclusion about the effects of learning rate (lr) in federated learning which is highly effective parameter in federated learning.

2) The next contribution is regarding individual clients epochs. The time variable which represents each clients runtime is used to find out the slacker devices and adjust the epoch values for reducing the number of slacker devices.

3) Pairing the time variable with lr and epoch values of client to perform better client selection which focused on fairness and heterogeneity. A client selection strategy FedLR has been proposed.

The remainder of the paper is organized as follows: Section II discusses Related Works. Section III explains the problem statement of the literature. Section IV introduces FedLR and explains its key features and how does it perform them. Section V talks about the implementation of FedLR and the datasets and model architectures used for the implementation. Section VI does result and analysis of the work with Section VII finally concluding this paper.

## II. RELATED WORKS

McMahan et al. [4] proposed the Federate Averaging algorithm and tested it using the MNIST Digit Dataset. The data was partitioned into IID and Non-IID. In Non-IID data, only two-digit data were given to the clients. They have also used CIFAR-10 in the balanced and IID settings. Khan et al. [8] and Nguyen et al. [9] have talked about advancements in federated learning for IoT, taxonomy and the open challenges. A state-of-the-art survey on the use of Federated Learning in smart healthcare. Advances in Federated Learning design

for healthcare addressing resource-aware federated learning, security and privacy federated learning.

Nishio et al. [10] proposed a new strategy for client selection in federated learning. The strategy is coined as FedCS (Federated Learning with Client Selection). They have added an extra step in the original FedAvg called Resource Request which gathers the client's resource information and groups them according to their resource capacity. They have also used schedule updates and upload and compared their results in both IID and Non-IID datasets.

Abdulrahaman et al. [11] proposed a multicriteria-based client selection (The server analyzes the client's responses to select the best set able to participate in the coming learning rounds). They have also added client filtering similar to [10]. They are not choosing clients at random rather; they are using Stratified Sampling.

Saha et al. [12] proposed fog-assisted federated learning for resource-constrained IoT devices. They have created a fog fl framework and formulated a greedy heuristic strategy to select the optimal global aggregator fog nodes at the end of an epoch to increase the reliability of the system. They have compared their findings with FedAvg and HFL. Shokri et al. [13] is the first paper to introduce privacy-preserving deep learning. They used distributed and selective SGD to make deep learning models privacy-preserving.

The MobileNets family of architecture [14], [15], and [16] are the best architectures for IoT devices because they are smaller in size and have yielded better results in IoT scenarios which makes them perfect for our use case. Mathur et al. [17] has implemented federated learning using the Flower framework. They have implemented federated learning in 5 mobile devices (three phones and two tablets). They used CIFAR-10 and Office-31 datasets in their experiments. They have evaluated their finding in terms of the local epoch, accuracy, convergence time (mins), and energy consumed by (kJ) the device. They have used the ever-popular MobileNetV2 [15] architecture.

Yang et al. [18] has written a book regarding the various keywords, features, and techniques of federated learning. This book is a good way to get acquainted with the concepts of federated learning. The book contains concepts for privacy-preserving, horizontal federated learning, vertical federated learning, and federated transfer learning.

Li et al. [19] have proposed Federated Domain Generalization, which is to add the concepts of Domain Generalization to Federated Learning. They have reviewed methods in Domain Generalization and Federated Learning and given their review on Federated Domain Generalization. Wang et al. [20] have talked about Statistical heterogeneity, communication cost, system heterogeneity, real-time etc, in the mHealth setting showing Federated Learning is also suitable for mobile health applications.

## III. PROBLEM STATEMENT

The FedAvg proposed by [4] open the doors for communication-efficient deep learning networks as they pro-

posed the Federated Averaging methods to train models collaboratively without sharing the underlying training data to preserve the privacy and ever since then it have become the state of the art method for Privacy preserving Deep Learning Models.

## A. Problem Setup

The vanilla FedAvg algorithm has some problems when it comes to client selection, heterogeneous computational resources, and fairness. As it focuses primarily on the privacy preserving aspect of the learning. The algorithm begins with initializing the global model at random or for pre-trained data. The beings the client selection process which selects clients at random. The global weights are distributed to the clients and the training begins and only stops when we get our desired outcome. To remedy the heterogeneity of FedAvg a new stategy FedProx [21] was proposed which try to contain the dropping slacker devices due to the heterogeneous nature of the devices.

## B. Client Selection

In the vanilla FedAvg the clients are selected at random using

$$m \leftarrow max(C \cdot K, 1) \qquad (1)$$

where C is the fraction of selected clients and K being the actual clients which gives us the total number of m clients for the federated learning. This method of selection client is good but it can easily be improved upon by proposing a new strategy for client selection. The proposed strategy should improve over the existing one by making the process more robust and highly communication efficient. A new parameter is required which can perform better client selection the current Equation 1 used by the vanilla FedAvg.

## C. Heterogeneous Computational Resources

The devices participating in the federated learning are not always homogeneous the chances of it being heterogeneous are more. A survey done by Markets and Markets [22] states that 35 key players exist in the edge devices manufacturing phase with some providing devices while other provides the processors. This many key players means that major amount of devices are heterogeneous which makes client selection much more complicated because as the diversity increases the number of slacker devices also increases. Slacker devices are the devices which are slowest performer in the system. The motto of federated learning is the faster the slacker device the model will be equally efficient. So that means in order to propose a better strategy one must figure out a way to improve upon the slacker devices.

## D. Fairness

Fairness is an unique thing which removes the discrimination between clients and making sure all the devices contribution are taken into consideration. The best way to understand this problem is to imagine two hospitals H1 and H2 with H1 being a bigger hospital with various specialization and H2 being a small clinic. Communicable Diseases are the most popular ones and easy to get and if H1 is getting patient for one such communicable disease that means is the same for H2 are well. But due to the size of H1 the data gathered from H2 will be minuscule but that doesn't mean that its irrelevant. So one must also consider fairness as one of the major components for creating a new strategy. Ezzeldin et al. [23] has proposed a Fairfed strategy to solve this problem using debiasing method across clients.

## IV. PROPOSED ALGORITHM: FEDLR

The proposed FedLR is broken into three segments: Learning Rate Selection, Epoch Selection for individual clients, and Client Selection based on the Client Runtime. The objective it to reduce the communication cost for efficient communication all the while preserving the accuracy of the model. A slacker device-centric algorithm has been proposed, focusing on the slowest devices and considering the fairness of client selection. A Federated Learning System is as fast as its slowest device. This is the basic principle of federated learning, so the proposed algorithm for client selection primarily focuses on optimizing the slowest devices in the federation. The algorithm is based on turning the learning rate and epochs of Individual clients. A time quantum has been proposed; the clients are classified as quicker or slacker based on the help of this time quantum. To validate if the learning rate based assumption (Section IV A) is correct (the bigger and smaller lr are assigned for quicker and slacker devices, respectively). However, this will only reduce the total communication time by a few seconds, so a new epoch for the slacker devices is proposed below in equation 2. to make it effective (Section IV B).

## A. Learning Rate Selection

The algorithm 1 below explains the learning rate (lr) selection by the clients. The learning rates are taken with an increment of ten. After running different lr it was observed that the accuracy were similar but the total communication times were being reduced with smaller lr values. All three lr cases were tested and they showed a good amount of change in communication cost. The dataset used for these experiments was the CIFAR10 dataset.

---

**Algorithm 1 Learning Rate Selection** The $K$ clients are indexed by $k$; $B$ is the local minibatch size, $E$ is the number of local epochs, and $\eta$ is the learning rate

---
1: **ClientUpdate**($k, W$): ▷ Run on client $k$
2: $\mathcal{B} \leftarrow$ (split $\mathcal{P}_k$ into batches of size $B$)
3: **for** each local epoch $i$ from 1 to $E$ **do**
4:     Fit $\eta$ as [0.01, 0.001, 0.0001] from the Server.
5:     **for** batch $b \in \mathcal{B}$ **do**
6:         $w \leftarrow w - \eta \nabla l(w; b)$
7:     **end for**
8: **end for**
9: return $w$ to server

---

*1) Case 1: FedLr (lr =0.01) with 10 Rounds and 5 Epochs:* The best accuracy we achieved in this scenario is 56.6%. This works similarly to the FedAvg Strategy and claims almost similar runtime the gap is 2s. The losses also perform similarly

to the base FedAvg Strategy. Table I shows the result of the experiment.

| FedLR (lr = 0.01) with 10 Round and 5 Epochs | | | | | |
|---|---|---|---|---|---|
| Clients | Accuracy | Loss | Local Acc | Local Loss | Total Time |
| 1 | 56.2% | 0.063 | 95.3% | 0.0050 | |
| 2 | 52.4% | 0.065 | 95.7% | 0.0047 | |
| 3 | 51.2% | 0.069 | 94.7% | 0.0053 | 835.6s |
| 4 | **56.6%** | 0.068 | 94.6% | 0.0056 | |
| 5 | 52.4% | 0.061 | 94.9% | 0.0054 | |

*2) Case 2: FedLr (lr =0.001) with 10 Rounds and 5 Epochs:* The best accuracy is 56.2%. The runtime is reduced by 6 seconds, which makes it half a second per round. However, the losses are greater compared to Case 1. In Case 1, the lowest loss was 0.0047; in Case 2, it was 0.0066. Table II shows the result of this experiment.

| FedLR (lr = 0.001) with 10 Round and 5 Epochs | | | | | |
|---|---|---|---|---|---|
| Clients | Accuracy | Loss | Local Acc | Local Loss | Total Time |
| 1 | 55.8% | 0.065 | 93.4% | 0.0066 | |
| 2 | 56% | 0.059 | 93.7% | 0.0064 | |
| 3 | 52.4% | 0.069 | 93.5% | 0.0067 | 830.9s |
| 4 | 52.4% | 0.069 | 93.7% | 0.0065 | |
| 5 | **56.2%** | 0.059 | 93.4% | 0.0066 | |

*3) Case 3: FedLr (lr =0.0001) with 10 Rounds and 5 Epochs:* The best accuracy is 57.6% with runtime reduced by 9 sec. While having similar losses as Case 1 and 2. The full results are in table III below.

| FedLR (lr = 0.0001) with 10 Round and 5 Epochs | | | | | |
|---|---|---|---|---|---|
| Clients | Accuracy | Loss | Local Acc | Local Loss | Total Time |
| 1 | 54.8% | 0.063 | 93% | 0.0060 | |
| 2 | **57.6%** | 0.062 | 95% | 0.0053 | |
| 3 | 56.8% | 0.063 | 95.2% | 0.0050 | 826.0s |
| 4 | 54.4% | 0.065 | 95.1% | 0.0052 | |
| 5 | 53.4% | 0.062 | 94.6% | 0.0056 | |

*B. Epoch Selection for Individual Clients*

The first step was to figure out suitable lr values which provides us with significant changes but just changing the lr values in not sufficient for us to make the strategy robust. The next step is to provide individual epoch values for each client based on a balanced parameter. Chen et al. [5] used CPU and RAM metric of clients in order to boost their strategy. This methodology is effective but is not fully privacy preserving. Device fingerprinting is a big risk to the Federated Learning scenario. Li et al. [21] proposed a Proximal term to improve the slacker devices which were dropping because of delays in computing $E$. The proposed equation 2 calculate local epochs for individual clients.

$$E_n = \left\lfloor \frac{mode[T_q] - count\_of\_value}{mode[T_q]} \right\rfloor * E \qquad (2)$$

In Equation 1. $mode[T_q]$ is the mode value from the time quantum, and *count_of_value* is the time the mode value is repeated. The key values for the Algorithm 2 are as follows:

1) The K clients are indexed by k.
2) C is the fraction of the client selected.
3) B is the local minibatch size.
4) E is the number of local epochs.
5) $\eta$ is the learning rate.
6) $T_q$ is time quantum.
7) $E_n$ is the new epoch.

---

**Algorithm 2 FedLR** The $K$ clients are indexed by $k$; $B$ is the local minibatch size, $E$ is the number of local epochs and $E_n$ is the new epoch, $T_q$ is the time quantum, and $\eta$ is the learning rate

---

```
1: Server Executes
2: initialize w_0
3: for each round t = 1,2,.. do
4:     m ← max(C · K, 1)
5:     S_t ←(random set of m clients)
6:     for each client k ∈ S_t in parallel do
7:         w_{t+1}^k ← ClientUpdate(k, w_t)
8:     end for
9:     w_{t+1} ← ∑_{k=1}^K (n_k/n) w_{t+1}^k
10: end for
11: ClientUpdate(k, W):                        ▷ Run on client k
12: B ← (split P_k into batches of size B)
13: for each local epoch i from 1 to E | E_n  do
14:     Find T_q (T_q is the epoch runtime of individual clients).
15:     Find mode[t_q]
16:     if: mode[t_q] is smallest
17:         i = E and η= Bigger Learning Rate
18:     else:
19:         Find E_n using equation 1.
20:         i = E_n and η = Smaller Learning Rate
21:     for batch b ∈ B  do
22:         w ← w − η∇l(w; b)
23:     end for
24: end for
25: return w to server
```

---

*C. Working of FedLR*

1) Initialize the global parameter value $\omega_0$.
2) Selection of all the eligible clients for training.
3) Finding each client's run-time as they finish training.
4) Finding the mode time value by comparing the individual times.
5) Selecting the learning rate and epoch values based on the mode value.
6) The bigger learning rate ($\eta$) will be assigned to quicker devices with the default epoch (E) value.
7) The slacker devices will be assigned a lower learning rate with a lower epoch value $E_n$ using equation 1. and a smaller $\eta$.
8) The time of these devices will be compared next, and the slacker devices will receive a low amount of work but will still be a pivotal element in the federated learning process.

## V. Implementation of FedLR

The model architecture used in the experiments is a 2 layered Convolution Neural Network. The list of datasets used for the experiments are as follows:

1) CIFAR10 [24] dataset containing ten classes.
2) MNIST [25] dataset of Handwritten digits containing ten classes.
3) FMNIST [26] dataset of fashion items containing ten classes.
4) CIFAR100 [27] dataset containing hundred classes.

All clients use the full datasets and run at 10 Epochs and 10 Communication Rounds. Using .25 and 1 GPU and CPU, respectively.

## VI. Result and Analysis

The Results of our Experiments are as follows: 10 Clients using FedAvg, FedProx, and FedLR on the all Datasets.

## VII. Conclusion

The proposed algorithm has reduced the total communication time by a significant margin and has maintained fairness in the system while maintaining the accuracy of the models. The proposed algorithm has also been tested on two different devices to prove the improvements in communication rounds for a federated learning system with many slacker devices. Both FedAvg and FedProx are compared with the proposed work and proposed work has significantly improved over both of them. Using proposed methodology in IoT and Edge devices will make the Federated Learning process more efficient and robust.

## References

[1] Yang Liu, Tao Fan, Tianjian Chen, Qian Xu, and Qiang Yang. Fate: An industrial grade platform for collaborative learning with data protection. *J. Mach. Learn. Res.*, 22(226):1–6, 2021.

[2] Nvidia. NVIDIA CLARA, 2020. https://www.nvidia.com/en-in/clara/ [Accessed 15-Sep-2023].

[3] Timothy Yang, Galen Andrew, Hubert Eichner, Haicheng Sun, Wei Li, Nicholas Kong, Daniel Ramage, and Françoise Beaufays. Applied federated learning: Improving google keyboard query suggestions. *arXiv preprint arXiv:1812.02903*, 2018.

[4] 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.

[5] Shuaijun Chen, Omid Tavallaie, Michael Henri Hambali, Seid Miad Zandavi, Hamed Haddadi, Song Guo, and Albert Y Zomaya. Boosting client selection of federated learning under device and data heterogeneity. *arXiv preprint arXiv:2310.08147*, 2023.

[6] Sakthi Vignesh Radhakrishnan, A Selcuk Uluagac, and Raheem Beyah. Gtid: A technique for physical device and device type fingerprinting. *IEEE Transactions on Dependable and Secure Computing*, 12(5):519–532, 2014.

[7] Kartik Patwari, Syed Mahbub Hafiz, Han Wang, Houman Homayoun, Zubair Shafiq, and Chen-Nee Chuah. Dnn model architecture fingerprinting attack on cpu-gpu edge devices. In *2022 IEEE 7th European Symposium on Security and Privacy (EuroS&P)*, pages 337–355. IEEE, 2022.

[8] Latif U Khan, Walid Saad, Zhu Han, Ekram Hossain, and Choong Seon Hong. Federated learning for internet of things: Recent advances, taxonomy, and open challenges. *IEEE Communications Surveys & Tutorials*, 2021.

[9] Dinh C Nguyen, Ming Ding, Pubudu N Pathirana, Aruna Seneviratne, Jun Li, and H Vincent Poor. Federated learning for internet of things: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, 23(3):1622–1658, 2021.

[10] Takayuki Nishio and Ryo Yonetani. Client selection for federated learning with heterogeneous resources in mobile edge. In *ICC 2019-2019 IEEE international conference on communications (ICC)*, pages 1–7. IEEE, 2019.

[11] Sawsan AbdulRahman, Hanine Tout, Azzam Mourad, and Chamseddine Talhi. Fedmccs: Multicriteria client selection model for optimal iot federated learning. *IEEE Internet of Things Journal*, 8(6):4723–4735, 2020.

[12] Rituparna Saha, Sudip Misra, and Pallav Kumar Deb. Fogfl: Fog-assisted federated learning for resource-constrained iot devices. *IEEE Internet of Things Journal*, 8(10):8456–8463, 2020.

[13] Reza Shokri and Vitaly Shmatikov. Privacy-preserving deep learning. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, pages 1310–1321, 2015.

[14] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[15] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.

[16] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for mobilenetv3. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1314–1324, 2019.

[17] Akhil Mathur, Daniel J Beutel, Pedro Porto Buarque de Gusmão, Javier Fernandez-Marques, Taner Topal, Xinchi Qiu, Titouan Parcollet, Yan Gao, and Nicholas D Lane. On-device federated learning with flower. *arXiv preprint arXiv:2104.03042*, 2021.

[18] Qiang Yang, Yang Liu, Yong Cheng, Yan Kang, Tianjian Chen, and Han Yu. Federated learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 13(3):1–207, 2019.

[19] Ying Li, Xingwei Wang, Rongfei Zeng, Praveen Kumar Donta, Ilir Murturi, Min Huang, and Schahram Dustdar. Federated domain generalization: A survey. *arXiv preprint arXiv:2306.01334*, 2023.

[20] Tongnian Wang, Yan Du, Yanmin Gong, Kim-Kwang Raymond Choo, and Yuanxiong Guo. Applications of federated learning in mobile health: Scoping review. *Journal of Medical Internet Research*, 25:e43006, 2023.

[21] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *Proceedings of Machine learning and systems*, 2:429–450, 2020.

[22] Markets and Markets. Edge AI Hardware Market, 2023. https://www.marketsandmarkets.com/Market-Reports/edge-ai-hardware-market-158498281.html [Accessed 22-Jul-2024].

[23] Yahya H Ezzeldin, Shen Yan, Chaoyang He, Emilio Ferrara, and A Salman Avestimehr. Fairfed: Enabling group fairness in federated learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 7494–7502, 2023.

[24] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 (canadian institute for advanced research).

[25] Yann LeCun. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998.

[26] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017.

[27] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009.