# Study of MobileNets Model in Federated Learning

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 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 in 2018, Europe passed its data privacy law, General Data Protection Regulation (GDPR). Soon after that, California also created its legislature called California Consumer Privacy Act (CCPA, 2018). India is also creating its privacy law, which is still under much scrutiny. Federated Learning is one such method which can satisfy the rules of cryptography and privacy laws around the globe.

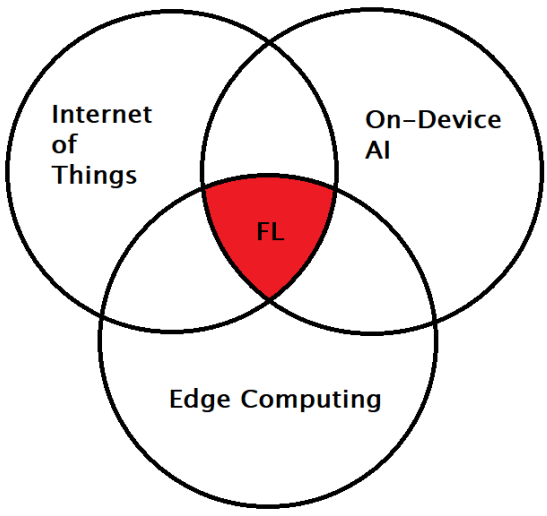


Fig. 1. Venn diagram of Federated Learning.

MolbileNets [5] are designed for mobile and embedded applications and in IoT-based products have done wonders due to their small size. The proposed work uses these models in a

Federated Learning environment which provided us with good results. Federated Learning has two models, local and global. The local model is trained on the edge device with actual user data; the global model is an aggregation of all the local models which is created using sharing the model weights. In Federated learning the data is never shared only the model weights are shared. In Fig. 2 a basic IoT Federated Learning Architecture is shown. The IoT devices will perform the local training and perform the global update by uploading the model weights to the FL Server. The FL server will train the global model by aggregating the global updates and will share the new weights using model update to the local models.

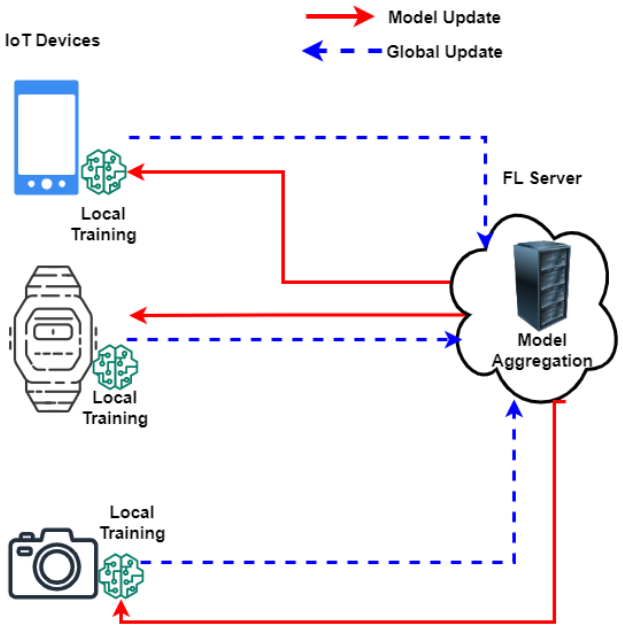


Fig. 2. Basic IoT Federated Learning Architecture.

**Workings of Federated Learning:**

1) The first phase is the initialization of the global parameter values. It can be pretrained or random.
2) The second phase is the selection of clients for participation in the rounds.
3) The third phase is to distribute those values to the participating clients.
4) The fourth phase is to update and upload the values from local to global models.
5) The fifth phase aggregates the model by averaging the data (FedAverage).
6) The sixth phase repeats phases second to first until we get the desired performance.

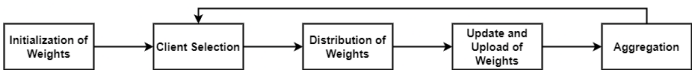


Fig. 3. Block Diagram of Federated Learning Architecture.

## 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. [6] and Nguyen et al. [7] 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. [8] 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. [9] 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 [8]. They are not choosing clients at random rather; they are using Stratified Sampling.

Saha et al. [10] 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. [11] 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 [5], [12], and [13] 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. [14] 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 [12] architecture.

Yang et al. [15] 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. [16] 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. [17] 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. Dataset and Model Architecture

Fig. 4 represent the dataset and NN architecture.

1) CIFAR10 dataset has been used, and the input size is $32 \times 32$ and ten classes.
2) MobileNet [5] is used; its size is 16MB and has 4.3M parameters with 28 layers of Convolutions.
3) MobileNetV2 [12] is used; its size is 14MB and has 3.5M parameters with 53 layers of Convolutions.
4) MobileNetV3Small [13] is used; its size is 9.83MB and has 3M parameters.
5) MobileNetV3Large [13] is used; its size is 21.11MB and has 5M parameters.

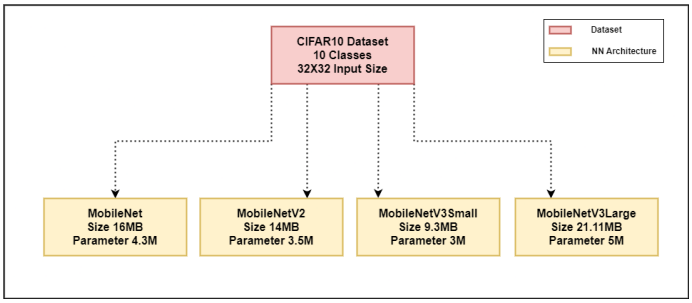


Fig. 4. Used Dataset and NN Architecture.

## IV. Framework Used

Flower FL [18] is a unified approach to federated learning, analytics, and evaluation. It can Federate any workload in any ML framework. The proposed methodology uses Flower for all the experiments. Fig. 5 represents the Flower FL framework.

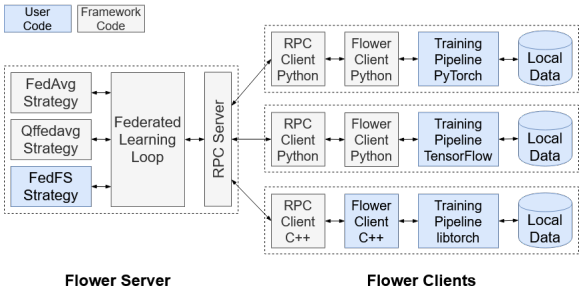


Fig. 5. Flower FL Framework.

The key aspects are we have two kinds of code: User and Framework. User codes are the models, and user-generated strategies like hyperparameter tuning, model architectures, etc. Framework codes are the critical parts of Flower-like the gRPC Server it uses. gRPC is based on two fast and efficient protocols: protocol buffers and HTTP/2. Protocol buffers are a data serialization protocol that is language-agnostic. It produces smaller binary payloads than JSON once it is serialized. The serialized data is transported using HTTP/2 which is fully multiplexed and can send data in parallel over a single TCP connection. The flower server takes care of the strategy and the number of communication rounds and features like setting up timeout, etc. The flower client takes care of the deep learning model in which we can use all major machine learning frameworks like PyTorch, Tensorflow, Mxnet, etc.

### A. FedAvg Algorithm

**Algorithm 1 FederatedAveraging.** 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

**Server Executes**
initialize $w_0$
**for** each round $t = 1, 2, \ldots$ **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**
    $w_{t+1}^k \leftarrow ClientUpdate(k, w_t)$
  **end for**
  $w_{t+1} \leftarrow \sum_{k=1}^{K} \frac{n_k}{n} w_{t+1}^k$
**end for**

**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**
    $w \leftarrow w - \eta \nabla l(w; b)$
  **end for**
**end for**
return $w$ to server

The algorithm is the primary protocol for federated learning. We initialize the global model randomly or by pre-trained data. The clients are chosen at random, and then the values are distributed. The data is updated and uploaded on the global models. These steps are iterated until we get the desired performance.

## V. Results and Discussion

### A. Running on Real-Time

Table 1. Shows the configuration of the devices used in the experiment.

TABLE I
Experimental Setup for Real-Time System.

| S No. | System Type | Processor | GPU | OS |
|---|---|---|---|---|
| 1 | Workstation | Intel Xeon Silver 4216, 32 cores | RTX A4000 | Windows |
| 2 | PC | Second generation Intel i5 2400, 4 cores | Intel | Manjaro OS |
| 3 | Laptop | AMD Ryzen 9 6900HS, 8 cores | 3060 Mobile | Windows |

Table 2. Shows the real-time result for the three devices in which the performance of the workstation with CPU was

similar to any ordinary PC. Ex. Local epoch times were 211s and 225s, respectively. If two clients run on a single machine

TABLE II
REAL-TIME WITH THREE DEVICES ON LOCAL NETWORK WITH 5 ROUNDS AND 5 EPOCHS.

| Device | Local Accuracy | Local Loss | Local Epoch | Server Accuracy | Server Loss | Server Epoch | Total Time |
|---|---|---|---|---|---|---|---|
| Workstation | 84.09 | 0.4610 | 211s | 78.77 | 0.6628 | 14s | 1.57Hrs |
| Laptop | 85.17 | 0.4283 | 33s | | | 3s | |
| PC | 85.15 | 0.4278 | 225s | | | 8s | |

(laptop) GPU and CPU, the laptop GPU takes 46s for an epoch and finishes quickly, and the CPU takes 126s for an epoch. However, the CPU epoch time changed to 99s after the GPU client finished training. Observed the 40s wait time after finishing each communication round. When the GPU was switched on the workstation, the epoch time was significantly reduced to 110sec. Half of what it was getting before. The server waits for 24hrs to receive clients. If none of the clients joins the federation, then the server sends an error message and goes back to waiting.

### *B. Running on Local System*

The PC is over a decade old, and its performance can be compared to the IoT and Edge devices of now. That is why we have chosen to run the experiments on this device. Technology has change a lot in a decade, but this PC is the closest device we could find to simulate our IoT systems.

TABLE III
TRAINING RESULTS IN PC WITH TWO CLIENTS (CPU).

| Architecture | Epoch and Round | Local Accuracy | Local Loss | Server Accuracy | Server Loss | Local Epoch Time | Server Epoch Time | Total Time |
|---|---|---|---|---|---|---|---|---|
| MobileNet | 20 and 20 (400) | 99.65% | 0.0114 | **80.81%** | 1.0538 | 614s | 17s | 67.1hr |
| MobileNetV2 | | 99.22% | 0.0246 | **80.74%** | 0.9376 | 314s | 12s | 43.8hr |
| MobileNetV3Small | | 64.24% | 1.0019 | 64.20% | 1.0017 | 206s | 7s | 22.7hr |
| MobileNetV3Small | | 66.20% | 0.9562 | | | | | |
| MobileNetV3Large | | 97.57% | 0.0735 | 77.04% | 1.1494 | 436s | 15s | 49.3hr |
| MobileNetV3Large | | 96.26% | 0.1055 | | | | | |

Table 3. Shows the training results in PC with Two Clients. MobileNetV2 is showing the best results. If we increase the no of clients in a single machine, the local time increases to 517s with 23s for the Server. That means one client finishes in 190 seconds. V3Small took half of what V2 took but didn't achieve the expected result. While using V3, we saw around a 2% of difference between local clients. This is the first time we have observed it. We got the best accuracy with V2 and the original MobileNet, but it took longer than V2. The codes and detailed results are available on Github [19].

### *C. Discussion*

The MobileNets model architectures are small, low-latency, low-powered models for resource-constrained devices. Federated Learning is famous for having big data overheads, and the smaller size of Mobilenet models reduces that burden and creates a robust system. The smaller size of these models comes at the cost of accuracy, but we have observed that with each new iteration, the model run time has reduced, but so is the accuracy. One of the reasons for low accuracy could be that the newer architecture doesn't suit our CIFAR-10 Dataset. The major application for these models is to solve computer vision problems in IoT and Mobile Devices.

## VI. CONCLUSION

Baseline benchmarks for the MobileNets family of models have been established for real-time and local systems. The performance of models is excellent for local models, but it is far behind in global accuracy. The V2 and the original MobileNets are the most optimised models for federated learning. The newer V3s are suitable for other applications, but they are not good with Federated Learning. This work shows the performance benchmarks of MobileNets in federated learning, which are ideal for computer vision applications.

## 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] 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.

[6] 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.

[7] 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.

[8] 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.

[9] 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.

[10] 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.

[11] 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.

[12] 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.

[13] 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.

[14] 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.

[15] 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.

[16] 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.

[17] 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.

[18] Daniel J Beutel, Taner Topal, Akhil Mathur, Xinchi Qiu, Titouan Parcollet, Pedro PB de Gusmão, and Nicholas D Lane. Flower: A friendly federated learning research framework. *arXiv preprint arXiv:2007.14390*, 2020.

[19] Vipul Singh Negi. Flower Codes for MobileNets model architecture, 2023. https://github.com/vipulhld001/FLWR [Accessed 15-Sep-2023].