# Fault-Tolerant Framework with Federated Learning for Reliable and Robust Distributed System

Lokendra Gour[1], Akhilesh A. Waoo[2]

{lokendra.gaur@gmail.com[1], akhileshwaoo@gmail.com[2]}
Dept. of CS and IT, AKSU Satna[1], HOD Dept. of CS and IT, AKSU Satna[2]

**Abstract.** Data intensive computing system like distributed systems suffer a potential problem of data storage, load balancing and privacy of confidential data. The distributed data needs to be analyzed and protected properly for ensuring data privacy and improving the fault tolerance. Data privacy and security have become prominent issues for data-oriented applications. Fault-tolerant must be deployed for smooth functioning of the distributed computing system. A Federated learning (FL) model can be deployed on multiple edge devices. The Federated learning model is deployed on both horizontal and vertical scopes. The feature of distributed computing is that it is growing tremendously towards horizontal and vertical directions. FL deployed with distributed deep learning could identify, recognize, and resolve the faults at a large scale. Federated learning is a multimodal deep learning system that captures training data across various distributed and decentralized edge devices.

**Keywords:** Big Data, Federated Learning, Edge Device, Fault Tolerance, Distributed Computing

## 1 Introduction

Fault is one of the major challenges in distributed environment. Fault tolerating framework is the most important element of a distributed system for reducing the chance of system failures. Federated learning [1, 2] is a kind of distributed and decentralized learning approach. The Federated learning (FL) method is most relevant approach applied in distributed system. Under this model a set of workers are deployed on local premises to train the local datasets. FL approach

THEETAS 2022, April 16-17, Jabalpur, India
Copyright © 2022 EAI
DOI 10.4108/eai.16-4-2022.2318146

also provides the service of data privacy and data security.[3, 4, 5]. FL is capable of detecting and diagnosing the anomalies and faults that occur frequently on edge devices and cloud.

Intelligent devices like smartphones, PCs or tablets and electronics devices achieved high scaling in present time. Devices equipped with multiple sensors and IoTs generates and consumes a tremendous quantity of information. In distributed computing end-user devices perform the training on local models and use localized datasets. The role parameter server is to aggregate the local updates and upgrading global models. Edge device operates as an intermediary system between the cloud and end-user devices. Edge behaves like the cloud. Each edge performs the work of obtaining the outcome of end devices, implements collections and classification, and finally delivers its intermediary output to the cloud environment for more processing.

The algorithm is designed on the principle of ensemble learning. Under this approach gradients are evaluated from the active nodes by a separate unit called aggregator. The variant of stochastic Gradient Descent (SGD) algorithm is implemented.[8, 9, 10].

## 2 System Architecture

Figure 1.1 depicts the model. The central server or parameter server performs the job of collecting local gradients. The fundamental function of the main controller is to receive requests from edge nodes. The basic purpose of the central server is to establish coordination among various working modules. The key objective of the proposed model is to enhance fault tolerance of the system by applying the machine learning prediction module [11, 12, 13, 14, 15]. Various computational units and methods of the system are framed and deployed by the algorithmic model.

Federated learning refers to the operations together with local data are executed at edge device. Mathematically federated learning is a graph G = (N, E), where the N = set of computational nodes and E = set of edges (bi/directional communication channel or links). The graph based framework for federated learning includes several edge devices like smart mobile phones [16, 17, 18], smart sensors, etc. Processes are executed on the edge devices. Every edge device has its own computational model together with associated data [19, 20].

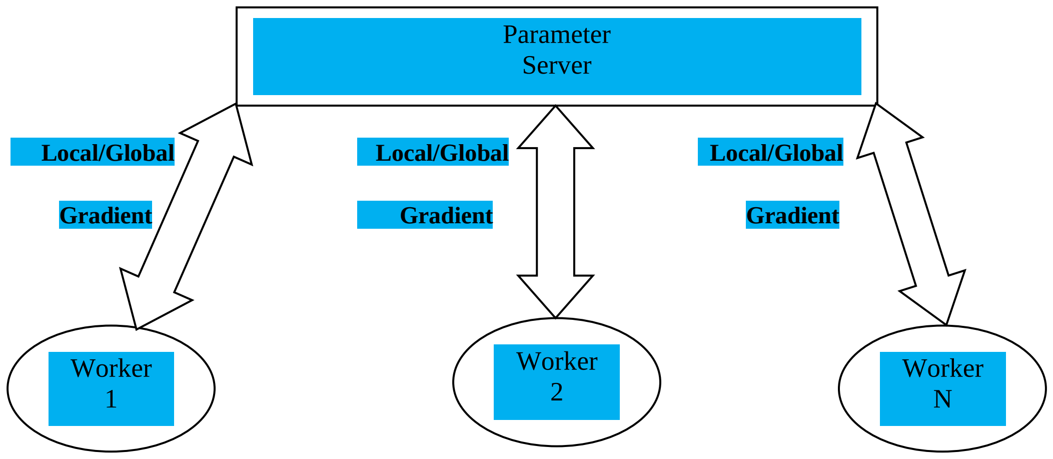


**Figure 1.1 Architecture of Federated Learning**

Figure 1.2 depicts the distribution of federated learning across the edge devices. The deep learning model operates among the edge devices. Each end device holds its dataset. Since data is stored on its local premises, data security and privacy is achieved. The central server merely orchestrates the edge devices. The updated information produced by the end devices is propagated to the central server for aggregation and in turn for final prediction. Privacy is implemented on both the central server and on end devices. Deep learning is the inter-nodal process for obtaining the hidden pattern or features of the datasets.

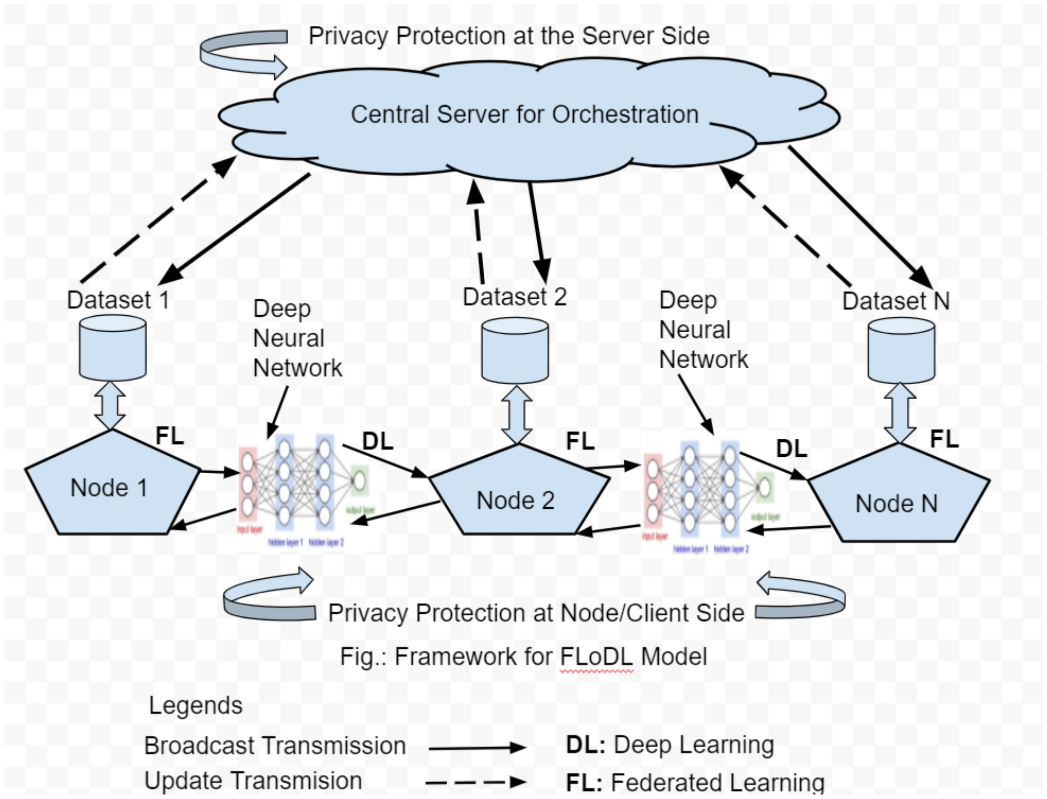


Figure 1.2 Federated Learning Distributed on Edge Devices

## 3 Proposed Methodologies

The Federated Learning algorithm incorporates an ensemble learning approach and establishes coordination among multiple distributed working machines. The system needs to communicate and aggregate the model's updated information in a secure, efficient, scalable, and fault-tolerant way. There is no need to share or exchange the local data for centralized updates. Data is updated at local domain on edge devices that preserves the privacy of confidential data.

**Federated Learning Algorithm**

```
Algorithm: FedLearning
Input: Dataset x
Output: Fault Resilient Model

 1. Set initial value for algorithm
 2. for r <=n do
 3.           Evaluate : $output_i$ = $fpp$ ($x_i$ , $\theta_i$);
 4.           calculate objective function: $c_i$ = loss ($\alpha$($x_i$), $output_i$);
 5.           If $d_{ir} < \varepsilon$
 6.           then
 7.                     Break;
 8.           else
 9.                     Evaluate: $grad_r$ = $bpp$ ($x_r$ , $\theta_i$, $c_r$);
10.           compute edge devices' gradients on their own local premises
                       and then aggregate updated gradients;
11.           Update: $\theta_{r+1} = \theta_r - k * grad_{new}$ ;
12.           end if
13.  end for
14.  return Fault Resilient system with parameters
```

**Table 1.1** Details of variates used in FedLearning

| S. No. | Variates | Description |
|---|---|---|
| 1 | $X_r$ | The data point at ith sample |
| 2 | $\theta$ | Model parameters |
| 3 | $fpp$ | Forward propagation procedure |
| 4 | $of$ | The objective function |
| 5 | $loss$ | The cost or loss function |
| 6 | $out$ | Output of iteration |
| 7 | $c_i$ | Cost calculation |
| 8 | $bpp$ | Back propagation procedure |
| 9 | $\varepsilon$ | Arbitrarily error |
| 10 | $grad_s$ | Gradient evaluated |
| 11 | $grad_{new}$ | Gradient evaluated by central computing server |
| 12 | k | Rate of learning/training |

Table 1.1 shows the parameters and hyper-parameters used in the computational model. These parameters are tuned to enhance the performance of the candidate system under consideration.

## 4 Experimental Setup

The proposed model is implemented by using Python language. Anaconda and Google Colab have been used as frameworks for proposed model. Keras and TensorFlow are used for model building. Functional API is best suited logic for proposed work. Dataset is bifurcated as training and testing the samples. To cover faults within an identified fault deep learning model is deployed to track the faults.

Various packages, libraries and dependencies are installed and configured to setup the workflow of the model. Some of the tools like R and Theano are used for visualization of the result/outcome.

Keras framework facilitates both sequential and functional models. It also support various kinds of libraries and modules.

## 4.1 Dataset Used

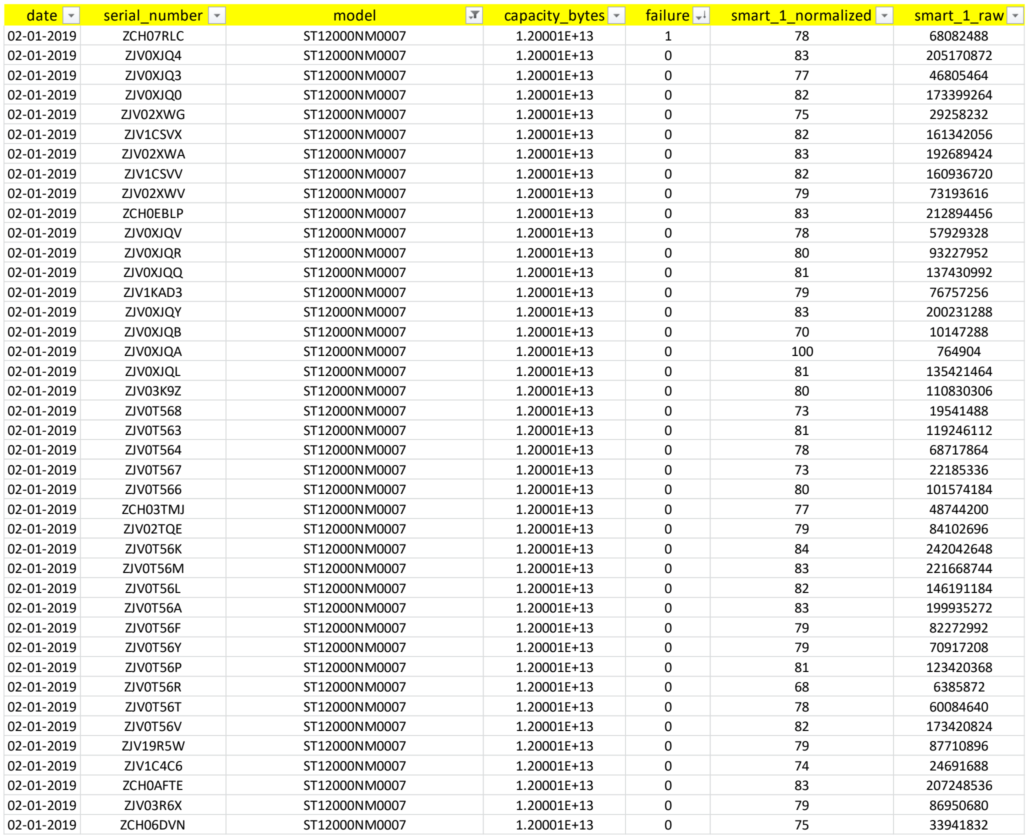
Datasets are the key ingredient to train, test, and validate the model for deep learning. Accurate and relevant datasets are hardly available or accessible for model training and testing in connection with fault tolerance. This research paper uses the datasets from Backblaze hard drive failure rates. The Backblaze dataset comprises various attributes like manufacturer, model name, drive size, drive count, Average age, etc. Drive size and drive failures have been taken as key attributes for predictions.

Table 1.2 shows the samples of data-points taken from Backblaze dataset. This dataset contains a large volume of data samples. Here only key attributes are considered for simplicity. Some of the key attributes are date, serial number, model, capacity bytes and failure etc.

Large dataset creates a major problem of loading in RAM as well as slow down the training, testing and inference process. One of the solutions for such kinds of problem is to create mini-batches or batches for the underlying dataset. Under the Keras functional API framework, inbuilt functions are implemented for ingesting the mini-batches to the proposed deep learning model.

Backblaze Hard Drive Failure Rates for 2020

(Reporting Period 01/01/2020 – 12/31/2020 inclusive)



| date | serial_number | model | capacity_bytes | failure | smart_1_normalized | smart_1_raw |
|---|---|---|---|---|---|---|
| 02-01-2019 | ZCH07RLC | ST12000NM0007 | 1.20001E+13 | 1 | 78 | 68082488 |
| 02-01-2019 | ZJV0XJQ4 | ST12000NM0007 | 1.20001E+13 | 0 | 83 | 205170872 |
| 02-01-2019 | ZJV0XJQ3 | ST12000NM0007 | 1.20001E+13 | 0 | 77 | 46805464 |
| 02-01-2019 | ZJV0XJQ0 | ST12000NM0007 | 1.20001E+13 | 0 | 82 | 173399264 |
| 02-01-2019 | ZJV02XWG | ST12000NM0007 | 1.20001E+13 | 0 | 75 | 29258232 |
| 02-01-2019 | ZJV1CSVX | ST12000NM0007 | 1.20001E+13 | 0 | 82 | 161342056 |
| 02-01-2019 | ZJV02XWA | ST12000NM0007 | 1.20001E+13 | 0 | 83 | 192689424 |
| 02-01-2019 | ZJV1CSVV | ST12000NM0007 | 1.20001E+13 | 0 | 82 | 160936720 |
| 02-01-2019 | ZJV02XWV | ST12000NM0007 | 1.20001E+13 | 0 | 79 | 73193616 |
| 02-01-2019 | ZCH0EBLP | ST12000NM0007 | 1.20001E+13 | 0 | 83 | 212894456 |
| 02-01-2019 | ZJV0XJQV | ST12000NM0007 | 1.20001E+13 | 0 | 78 | 57929328 |
| 02-01-2019 | ZJV0XJQR | ST12000NM0007 | 1.20001E+13 | 0 | 80 | 93227952 |
| 02-01-2019 | ZJV0XJQQ | ST12000NM0007 | 1.20001E+13 | 0 | 81 | 137430992 |
| 02-01-2019 | ZJV1KAD3 | ST12000NM0007 | 1.20001E+13 | 0 | 79 | 76757256 |
| 02-01-2019 | ZJV0XJQY | ST12000NM0007 | 1.20001E+13 | 0 | 83 | 200231288 |
| 02-01-2019 | ZJV0XJQB | ST12000NM0007 | 1.20001E+13 | 0 | 70 | 10147288 |
| 02-01-2019 | ZJV0XJQA | ST12000NM0007 | 1.20001E+13 | 0 | 100 | 764904 |
| 02-01-2019 | ZJV0XJQL | ST12000NM0007 | 1.20001E+13 | 0 | 81 | 135421464 |
| 02-01-2019 | ZJV03K9Z | ST12000NM0007 | 1.20001E+13 | 0 | 80 | 110830306 |
| 02-01-2019 | ZJV0T568 | ST12000NM0007 | 1.20001E+13 | 0 | 73 | 19541488 |
| 02-01-2019 | ZJV0T563 | ST12000NM0007 | 1.20001E+13 | 0 | 81 | 119246112 |
| 02-01-2019 | ZJV0T564 | ST12000NM0007 | 1.20001E+13 | 0 | 78 | 68717864 |
| 02-01-2019 | ZJV0T567 | ST12000NM0007 | 1.20001E+13 | 0 | 73 | 22185336 |
| 02-01-2019 | ZJV0T566 | ST12000NM0007 | 1.20001E+13 | 0 | 80 | 101574184 |
| 02-01-2019 | ZCH03TMJ | ST12000NM0007 | 1.20001E+13 | 0 | 77 | 48744200 |
| 02-01-2019 | ZJV02TQE | ST12000NM0007 | 1.20001E+13 | 0 | 79 | 84102696 |
| 02-01-2019 | ZJV0T56K | ST12000NM0007 | 1.20001E+13 | 0 | 84 | 242042648 |
| 02-01-2019 | ZJV0T56M | ST12000NM0007 | 1.20001E+13 | 0 | 83 | 221668744 |
| 02-01-2019 | ZJV0T56L | ST12000NM0007 | 1.20001E+13 | 0 | 82 | 146191184 |
| 02-01-2019 | ZJV0T56A | ST12000NM0007 | 1.20001E+13 | 0 | 83 | 199935272 |
| 02-01-2019 | ZJV0T56F | ST12000NM0007 | 1.20001E+13 | 0 | 79 | 82272992 |
| 02-01-2019 | ZJV0T56Y | ST12000NM0007 | 1.20001E+13 | 0 | 79 | 70917208 |
| 02-01-2019 | ZJV0T56P | ST12000NM0007 | 1.20001E+13 | 0 | 81 | 123420368 |
| 02-01-2019 | ZJV0T56R | ST12000NM0007 | 1.20001E+13 | 0 | 68 | 6385872 |
| 02-01-2019 | ZJV0T56T | ST12000NM0007 | 1.20001E+13 | 0 | 78 | 60084640 |
| 02-01-2019 | ZJV0T56V | ST12000NM0007 | 1.20001E+13 | 0 | 82 | 173420824 |
| 02-01-2019 | ZJV19R5W | ST12000NM0007 | 1.20001E+13 | 0 | 79 | 87710896 |
| 02-01-2019 | ZJV1C4C6 | ST12000NM0007 | 1.20001E+13 | 0 | 74 | 24691688 |
| 02-01-2019 | ZCH0AFTE | ST12000NM0007 | 1.20001E+13 | 0 | 83 | 207248536 |
| 02-01-2019 | ZJV03R6X | ST12000NM0007 | 1.20001E+13 | 0 | 79 | 86950680 |
| 02-01-2019 | ZCH06DVN | ST12000NM0007 | 1.20001E+13 | 0 | 75 | 33941832 |

Table 1.2 Samples or data-points taken from Backblaze datasets

## 4.2 Simulation Tools

The proposed algorithm is implemented by using Keras API framework, Cloud based platforms like Google Colab and CloudSim simulator are used to simulate the model. Python and R languages are used to implement the business logic. Tensorquant is most popular tools for ML and DL.

### 4.3 Cloud Platforms

Deep learning model requires large datasets with multiple dimensions to yield accurate predictions. Whenever datasets become voluminous and complex, training and inference process goes slow down. Cloud platform provides various appropriate system resources and services like CPU processing speed, higher capacity RAM, GPU and TPU. AWS machine learning services are well suited cloud platform to resolve such kinds of problems.

## 5 Comparison with the Existing Model

The FedLearning is the best suited model that can tolerate the arbitrary faults most probable to occur in distributed systems. By compared to other models the proposed algorithm produces higher accuracy and calculated predictions. The present model gives better performance amongst the existing model in terms of processing speeds and better predictions.

## 6 Conclusions and Discussion

Disk failure is very common among the other system failures. Through the extensive analysis and training process on the voluminous data concerning disk failures assist the prediction on disk failures on unknown data points. Disk failure causes both economic and operational loss.

## 7 Future Work

FedLearning suffers the key problem of computational communication cost across the nodes and central server. The present body of the work opens the options for further research in the direction of communication cost of computation, data security and privacy of confidential data. In future, further work may include the integration of  Bayesian optimization algorithms with FedLearning algorithm.

**Acknowledgement**

The authors are extremely grateful to AKS University, Satna for providing the relevant resources, infrastructure, and technical support during the development of the entire project work.

## References

[1] Chen Y, Qin X, Wang J, Yu C, Gao W, Fedhealth: A federated transfer learning framework for wearable healthcare. IEEE Intelligent Systems. (2019)

[2] Kang J, Xiong Z, Niyato D, Zou Y, Zhang Y, Guizani M. Reliable federated learning for mobile networks. IEEE Wireless Communications; 27 (2) pp. 72–80 (2020)

[3] Sugawara E, Fukushi M, Horiguchi S, Fault-tolerant multi-layer neural networks with ga training. in Proc. 18th IEEE Int. Symp. Defect Fault Tolerance VLSI Syst. pp. 328–335. (2003)

[4] Mahdiani HR, Fakhraie SM, and C. Lucas, Relaxed fault-tolerant hardware implementation of neural networks in the presence of multiple transient errors. IEEE Trans. Neural Netw. Learn. Syst., vol. 23, no. 8, pp. 1215–1228 (2012)

[5] Rincon CA., Paris JF, Vilalta R, Cheng AMK, D.D.E. Long, Disk failure prediction in heterogeneous environments, in 2017 International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS). IEEE, Seattle, WA: pp. 1–7 (2017)

[6] Satyanarayanan M. The emergence of edge computing. Computer 50 (1) (2017) 30–39. doi:10.1109/MC.2017.9.

[7] Khan WZ, Ahmed E, Hakak S, Yaqoob L, Ahmed A, Edge computing: A survey. Future Generation Computer Systems 97 pp. 219–235 (2019)

[8] Seide F, Fu H, Droppo J, Li G, Yu D. 1-bit stochastic gradient descent and application to data-parallel distributed training of speech dnns, in Interspeech , interspeech Edition, 2014, pp. 1–2 2014

[9] Gandikota V, Kane D, Maity RK, Mazumdar A, vqsgd: Vector quantized stochastic gradient descent, arXiv preprint arXiv:1911.07971 (2019)

[10] Brutzkus A, Globerson A, Malach E, Shalev-Shwartz S, SGD learns over-parameterized networks that probably generalize on linearly separable data, in International Conference on Learning Representations, pp. 1–2 (2018)

[11] Leduc-Primeau F, Gripon V, Rabbat MG, Gross WJ, Fault-tolerant associative memories based on c-partite graphs. IEEE Trans. Signal Process., vol. 64, no. 4, pp. 829–841 (2016)

[12] Patan K, Neural Network-Based Model Predictive Control: Fault Tolerance and Stability. IEEE Transactions on Control Systems Technology. 23 pp. 1147–1155. https://doi.org/10.1109/TCST.2014.2354981 (2015)

[13] Chandra P, Singh Y, Fault tolerance of feedforward artificial neural networks—A framework of a study, in Proc. Int. Joint Conf. Neural Netw., vol. 1. pp. 489–494 (2003)

[14] Kumari P, Kaur P, A survey of fault tolerance in cloud computing, Journal of King Saud University - Computer and Information Sciences. https://doi.org/10.1016/j.jksuci.2018.09.021 (2018)

[15] Du Z, Lingamneni A, Chen Y, Palem KV, Temam O, Wu C. Leveraging the error resilience of neural networks for designing highly energy-efficient accelerators. IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst., vol. 34, no. 8, pp. 1223–1235 (2015)

[16] Yi S, Li C, Li Q, A survey of fog computing: Concepts, applications, and issues, in Proceedings of the 2015 Workshop on Mobile Big Data, Mobidata '15, Association for Computing Machinery, New York, NY, USA, pp. 3742 (2015)

[17] Yi S, Qin Z, Li Q, Security and privacy issues of fog computing: A survey, in: in International Conference on Wireless Algorithms, Systems, and Applications (WASA), pp. 685–695. (2015)

[18] Tao Z, Xia Q, Hao Z, Li C, Ma L, Yi S, Li Q, A survey of virtual machine management in edge computing, Proceedings of the IEEE 107 (8) pp. 1482–1499 (2019)

[19] Rodrigues TG, Suto K, Nishiyama H, Kato N, Temma K, Cloudlets activation scheme for scalable mobile edge computing with transmission power control and virtual machine migration, IEEE Transactions on Computers 67 (9) pp. 1287–1300 (2018)

[20] Nishio T, Yonetani R, Client selection for federated learning with heterogeneous resources in mobile edge, in ICC 2019 - 2019 IEEE International Conference on Communications (ICC) (2019)

[21] Yoshida N, Nishio T, Morikura M, Yamamoto K, Yonetani R, Hybrid-fl for wireless networks: Cooperative learning mechanism using non-iid data, in ICC 2020 - 2020 IEEE International Conference on Communications (ICC), pp. 1–7 (2020)

[22] Yang HH, Liu Z, S TQ. Quek, Poor HV, Scheduling policies for federated learning in wireless networks, IEEE Transactions on Communications 68 (1) pp. 317–333 (2020)

[23] Dinh CT, Tran NH, H MN. Nguyen, Hong CS, Bao W, Zomaya AY, Gramoli V, Federated learning over wireless networks: Convergence analysis and resource allocation, IEEE/ACM Transactions on Networking (2020)