



# Federated Learning

Principles, Paradigms, and Applications

Jayakrushna Sahoo | Mariya Ouissa | Akarsh K. Nair  
Editors

# **FEDERATED LEARNING**

*Principles, Paradigms, and Applications*



Taylor & Francis

Taylor & Francis Group

<http://taylorandfrancis.com>

# **FEDERATED LEARNING**

*Principles, Paradigms, and Applications*

*Edited by*

**Jayakrushna Sahoo, PhD**

**Mariya Ouaissa, PhD**

**Akarsh K. Nair**



First edition published 2025

**Apple Academic Press Inc.**

1265 Goldenrod Circle, NE,  
Palm Bay, FL 32905 USA

760 Laurentian Drive, Unit 19,  
Burlington, ON L7N 0A4, CANADA

**CRC Press**

2385 NW Executive Center Drive,  
Suite 320, Boca Raton FL 33431

4 Park Square, Milton Park,  
Abingdon, Oxon, OX14 4RN UK

© 2025 by Apple Academic Press, Inc.

*Apple Academic Press exclusively co-publishes with CRC Press, an imprint of Taylor & Francis Group, LLC*

Reasonable efforts have been made to publish reliable data and information, but the authors, editors, and publisher cannot assume responsibility for the validity of all materials or the consequences of their use. The authors are solely responsible for all the chapter content, figures, tables, data etc. provided by them. The authors, editors, and publishers have attempted to trace the copyright holders of all material reproduced in this publication and apologize to copyright holders if permission to publish in this form has not been obtained. If any copyright material has not been acknowledged, please write and let us know so we may rectify in any future reprint.

Except as permitted under U.S. Copyright Law, no part of this book may be reprinted, reproduced, transmitted, or utilized in any form by any electronic, mechanical, or other means, now known or hereafter invented, including photocopying, microfilming, and recording, or in any information storage or retrieval system, without written permission from the publishers.

For permission to photocopy or use material electronically from this work, access [www.copyright.com](http://www.copyright.com) or contact the Copyright Clearance Center, Inc. (CCC), 222 Rosewood Drive, Danvers, MA 01923, 978-750-8400. For works that are not available on CCC please contact [mpkbookpermissions@tandf.co.uk](mailto:mpkbookpermissions@tandf.co.uk)

Trademark notice: Product or corporate names may be trademarks or registered trademarks and are used only for identification and explanation without intent to infringe.

---

**Library and Archives Canada Cataloguing in Publication**

---

.....

CIP data on file with Canada Library and Archives

.....

---

**Library of Congress Cataloging-in-Publication Data**

---

.....

CIP data on file with US Library of Congress

.....

---

ISBN: 978-1-77491-638-4 (hbk)

ISBN: 978-1-77491-639-1 (pbk)

ISBN: 978-1-00349-719-6 (ebk)

# About the Editors

---

## **Jayakrushna Sahoo, PhD**

*Head, Computer Science and Engineering Department,  
Indian Institute of Information Technology, Kottayam, India*

Jayakrushna Sahoo, PhD, has been associated with the Indian Institute of Information Technology (IIIT) Kottayam, since July 2019, where he also serves as the coordinator of the institute's PhD program. Prior to this, he has experience working with BML Munjal University, Gurgaon, India, in the capacity of Assistant Professor in the Department of Computer Science and Engineering. Dr. Sahoo has also worked as an ad hoc faculty in the Department of Computer Applications at the National Institute of Technology (NIT), Jamshedpur, India. He received his PhD with specialization in Data Mining from the Indian Institute of Technology (IIT) in Kharagpur, India, and obtained his MTech degree in Computer Science and Engineering from the International Institute of Information Technology (IIIT), Bhubaneswar, India. He has publications in reputed journals over the years. His current research interests include data mining, machine learning, and federated learning. Dr. Sahoo has immense experience in research as he has been guiding several PhD scholars and has been associated with some of the premier institutions of the country. He also took up the roles of resource person and technical panel member and headed several international conferences inside the country. Additionally, he has expertise in research article publishing, and he has also served as a reviewer for well-reputed journals and some of the premier conferences in the country.

## **Mariya Ouissa, PhD**

*Professor; Institute Specializing in New Information and Communication Technologies; Researcher Associate and Practitioner*

Mariya Ouissa, PhD, is currently a Professor at an institute specializing in new information and communication technologies as well as a research associate and a practitioner with industry and academic experience. She completed her PhD in 2019 in computer science and networks at the Laboratory of Modelization of Mathematics and Computer Science from Ensam-Moulay Ismail University, Meknes, Morocco. She is a networks

and telecoms engineer who graduated in 2013 from the National School of Applied Sciences in Khouribga, Morocco. She is a co-founder and consultant at its support and consulting center. She worked for the School of Technology of Meknes, Morocco, as a visiting professor from 2013 to 2021. She is a member of the International Association of Engineers and the International Association of Online Engineering, and since 2021, she has been an ACM Professional Member. She is an expert reviewer with the Academic Exchange Information Center (AEIC) and a brand ambassador with Bentham Science. She has served and continues to serve on technical programs and organizer committees of several conferences and events, and she has organized many symposiums/workshops/conferences as a general chair and also as a reviewer of numerous international journals. Dr. Ouaissa has made contributions in the fields of information security and privacy, Internet of Things security, and wireless and constrained network security. Her main research topics are IoT, M2M, D2D, WSN, cellular networks, and vehicular networks. She has published over 20 papers (book chapters, international journals, and conferences/workshops), eight edited books, and five special issues as a guest editor.

**Akarsh K. Nair**

*Doctoral Researcher, Indian Institute of Information Technology,  
Kottayam, India*

Akarsh K. Nair is a Doctoral Researcher at the Indian Institute of Information Technology (IIIT) Kottayam, specializing in distributed learning, machine learning, federated learning, and edge intelligence. Akarsh has worked as an Assistant Professor in the Department of Computer Science at TEC College, Palakkad, India. He is also associated with the iHub HCI Foundation and IIT Mandi under the capacity of a doctoral fellow. He has already published several research articles in reputed scientific journals and international platforms. He has also acted as a reviewer for prestigious scientific journals.

# Contents

---

<i>Contributors</i> .....	<i>ix</i>
<i>Abbreviations</i> .....	<i>xi</i>
<i>Preface</i> .....	<i>xv</i>
<b>1. The Evolution of Machine Learning: From Centralized to Distributed .....</b>	<b>1</b>
Jayakrushna Sahoo, Akarsh K. Nair, and Richa Sharma	
<b>2. Types of Federated Learning and Aggregation Techniques.....</b>	<b>23</b>
S. Shailesh and Joseph James	
<b>3. Federated Learning for IoT/Edge/Fog Computing Systems.....</b>	<b>47</b>
Balqees Talal Hasan and Ali Kadhum Idrees	
<b>4. Adopting Federated Learning for Software-Defined Networks .....</b>	<b>77</b>
Akarsh K. Nair, Jayakrushna Sahoo, and Gaurav Jaswal	
<b>5. Federated Learning in the Internet of Medical Things .....</b>	<b>107</b>
S. Sabapathi, N. Vijayalakhmi, and S. Sindhu	
<b>6. Federated Learning Approaches for Intrusion Detection Systems: An Overview .....</b>	<b>131</b>
Akarsh K. Nair, Jayakrushna Sahoo, and Gaurav Jaswal	
<b>7. Exploring Communication Efficient Strategies in Federated Learning Systems.....</b>	<b>153</b>
Akarsh K. Nair, Jayakrushna Sahoo, and Ebin Deni Raj	
<b>8. Federated Learning and Privacy, Challenges, Threat and Attack Models, and Analysis.....</b>	<b>183</b>
Sheema Madhusudhanan, Arun Cyril Jose, and Reza Malekian	
<b>9. Analyzing Federated Learning From a Security Perspective.....</b>	<b>213</b>
Akarsh K. Nair, Jayakrushna Sahoo, and Ebin Deni Raj	
<b>10. Blockchain Integrated Federated Learning in Edge/Fog/Cloud Systems for IoT-Based Healthcare Applications: A Survey .....</b>	<b>237</b>
Shinu M. Rajagopal, M. Supriya, and Rajkumar Buyya	
<b>11. Incentive Mechanism for Federated Learning.....</b>	<b>271</b>
Lekha C. Warrier, G. K. Ragesh, and Pao-Ann Hsiung	

<b>12. Protected Shot-Based Federated Learning for Facial Expression Recognition .....</b>	<b>297</b>
A. Sherly Alphonse Rao and J. V. Bibal Benifa	
<b><i>Index.....</i></b>	<b>321</b>

# Contributors

---

**J. V. Bibal Benifa**

Indian Institute of Information Technology, Kottayam, Kerala, India

**Rajkumar Buyya**

CloudsLab, School of Computing and Information Systems, The University of Melbourne, Australia

**Balqees Talal Hasan**

Department of Computer and Information Engineering, College of Electronics Engineering, Nineveh University, Mosul, Iraq

**Pao-Ann Hsiung**

National Chung Cheng University, Taiwan

**Ali Kadhum Idrees**

Department of Information Security, College of Information Technology, University of Babylon, Babylon, Iraq

**Joseph James**

Department of Computer Science, Mar Gregorios College of Arts and Science, Alappuzha, Kerala, India

**Gaurav Jaswal**

iHUB and HCI Foundation, Indian Institute of Technology, Mandi, Himachal Pradesh, India

**Arun Cyril Jose**

Department of Computer Science and Engineering, Indian Institute of Information Technology Kottayam (IIITK), Kerala, India

**Sheema Madhusudhanan**

Department of Computer Science and Engineering, Indian Institute of Information Technology Kottayam (IIITK), Kerala, India

**Reza Malekian**

Department of Computer Science and Media Technology, Internet of Things and People Research Center, Malmö University, Malmö, Sweden

**Akarsh K. Nair**

Department of Computer Science and Engineering, Indian Institute of Information Technology, Kottayam, Kerala, India

**G. K. Ragesh**

Department of Cyber Security, Indian Institute of Information Technology, Kottayam, Kerala, India

**Ebin Deni Raj**

Department of Computer Science and Engineering, Indian Institute of Information Technology, Kottayam, Kerala, India

**Shinu M. Rajagopal**

Department of Computer Science and Engineering, Amrita School of Computing, Amrita Vishwa Vidyapeetham, Bangalore, Karnataka, India

**A. Sherly Alphonse Rao**

Vellore Institute of Technology, Chennai, Tamil Nadu, India

**S. Sabapathi**

Saveetha College of Liberal Arts and Science, Chennai, Tamil Nadu, India

**Jayakrushna Sahoo**

Department of Computer Science and Engineering, Indian Institute of Information Technology, Kottayam, Kerala, India

**S. Shailesh**

Department of Computer Science, Cochin University of Science and Technology, Ernakulam, Kerala, India

**Richa Sharma**

Department of Mathematics, Computer Science and Forensics, Commonwealth University (Lock Haven University), Lock Haven, PA, USA

**S. Sindhu**

SRM Institute of Science and Technology, Chennai, Tamil Nadu, India

**M. Supriya**

Department of Computer Science and Engineering, Amrita School of Computing, Amrita Vishwa Vidyapeetham, Bangalore, Karnataka, India

**N. Vijayalaskhmi**

SRM Institute of Science and Technology, Chennai, Tamil Nadu, India

**Lekha C. Warrier**

Department of Computer Science, Indian Institute of Information Technology, Kottayam, Kerala, India

# Abbreviations

---

ACL	access control list
AGA	analog aggregation
AI	artificial intelligence
AMA	all model averaging
Asics	application specific integrated circuits
ATO	account take over
BAA	broadband analog aggregation
BAFFLE	blockchain based federated learning without aggregator
BCFL	blockchain based federated learning
BFAN	blockchain and fog-based architecture network
BMA	best model averaging
BN	batch normlaiztaion
C&W	carlini & wagner attack
CADSGD	compressed analog distributed sgd
CCPA	california consumer privacy act
CFD	compressed federated distillation
CNN	convolutional neural network
CS	compression sensing
CSI	channel state information details
DDOS	distributed denial of service attacks
DEM	distributed mean estimation
DFD	data flow diagram
DH	diffie-hellman
DL	deep learning
DML	distributed machine learning
Dos	denial of service
DP	differential privacy
EHR	electronic health information
FATE	federated AI technology enabler
FC	federated cloud computing
FD	federated distillation
FDBSS	federated database systems
FEMNIST	federated extended mnist
FER	facial expression recognition

FGSM	fast gradient sign method
FHE	fully homomorphic encryption
FL	federated learning
Flwr	flower
FPK	fokker-plank kolmogorov
GAN	generator adversarial network
GBMA	gradient-based multiple access
GDP	global differential privacy
GDPR	general data protection regulation
GPU	graphical processing unit
HAR	human activity recognition
HE	homomorphic encryption
HIDS	host intrusion detection systems
HJB	hamilton-jacobi-bellman
HSQ	hyper-sphere quantization
IDS	intruition detection system
IID	independent and identically distributed
IoHT	internet of health things
IoT	internet of things
JSMA	jacobian-based saliency map attack
KD	knowledge distillation
LBFGS	limited-memory broyden-fletcher-goldfarb-shanno
LDP	local differential privacy
MAC	medium access channel
MFG	mean-field game
MGAN	mixture generative adversarial network
ML	machine leanring
MLP	multi layer perceptrons
NIDS	network intrusion detection system
NLP	natural laguage procession
OMS	model selection
PEN	perceptive extraction network
PET	privacy enhancing technologies
PFL	paddle federated learning
PIA	property inference attack
PL	privacy loss
Png	persona non-grata
PoS	proof of stake
PoWw	proof of work

PPML	privacy-preserving machine learning
PSFL	protected-shot-based federated learning
QSGD	quantized stochastic gradient descent
REE	rich execution environment
RNN	recurrent neural network
SA	secure aggregation
SGD	stochastic gradient approach
SIFT	scale invariant feature transform
SITS	satellite-terrestrial integrated networks
SMC	secure multi-party competition
SPV	simplified payment verification
SQ	stochastic quantization
SWHE	somewhat homomorphic encryption
TCS	time-correlation sparsification
TEE	trusted execution environment
TFF	tensorflow federated
UAV	unmanned aerial vehicles
VEC	Vehicular edge computing



Taylor & Francis

Taylor & Francis Group

<http://taylorandfrancis.com>

# Preface

---

The rapid development in IoT and distributed computing has facilitated a rapid shift in the way most of the existing technologies have been perceived. As most of the systems are currently shifting to their lightweight distributed counterparts, it is really hard for conventional AI approaches to adapt to the changing resource-constrained environments. Thus, newer techniques such as distributed learning, edge intelligence, and federated learning are finding acceptance in the current technological scenario. Compared to other technologies, federated learning has facilitated the implementation of AI in devices of all levels, irrespective of their computational capabilities, thus gaining wide acceptability. It has also brought the added benefits of privacy preservation into classical AI approaches, thus making it a hotly researched topic of the decade. This book aims to prepare the research scholar, information technology professional, and distributed computing engineers to understand various aspects of federated learning concepts and computational techniques used for its real-life implementations and related applications in various other domains.

The book describes federated learning and various tectonics of FL, including architectural paradigms, basic terminologies, aggregation methodologies, data partition schemes, security and privacy aspects, combination applications with other technologies, and lots more.

The book contains 12 chapters that provide an in-depth understanding of federated learning, which will aid FL enthusiasts and researchers in understanding and applying technology to various actions and further research.

Chapter 1 introduces the general concepts of machine learning. It examines the rationale behind the shift from centralized learning approaches to the decentralized system. Various advancements in distributed learning, along with the introduction of federated learning, are also presented. The growth and technological formation of FL from its preliminary stages are also presented.

Chapter 2 summarizes the federated learning system setup and various terminologies associated with it. It presents a basic classification of FL based on the system architecture, data distribution, and other important categorizations. A comprehensive analysis of the various aggregation approaches based on their types and some of the state-of-the-art approaches are also performed.

Chapter 3 includes studies on the application of FL for various distributed computing scenarios. Implementation and various use cases of FL in edge-based, fog-based, and IoT systems are presented comprehensively. Various challenges associated with the integration are also listed.

Chapter 4 provides an integrated view of applications of software-defined networks, FL, and their integration. Extensive literature on their similarity in architecture, the rationale behind integration, and the benefits of integration are presented. The ability of FL to harness the features of software-defined networking (SDN) to deliver better results is also discussed, along with various advantages and disadvantages as well.

Chapter 5 highlights the role of federated learning in IoMT systems. It enlightens readers with the basic concept and functioning of the IoMT system, along with their daily use cases as well. FL can deliver an extensive solution to some of the classical problems associated with IoMT systems. Thus, the integration of FL in IoMT has a prospective future.

Intrusion detection systems are usually employed to monitor traffic in a system, both internal and external, to identify malicious activities and alert system users before the system gets affected. Even though machine learning-based predictive models have been found to be highly applicable to IDS systems, they have certain shortcomings as well. Thus, Chapter 6 provides an in-depth study of FL-based intrusion detection systems, discussing their taxonomy and functioning and showcasing their superiority over the existing systems as well.

Chapter 7 provides a pragmatic analysis of some of the strategies for developing a communication-efficient federated learning system. As communication bottlenecks are a major issue affecting the performance of FL systems, making the system communication efficient is a prime consideration for research. Thus, some of the common strategies for attaining communication efficiency, such as data comparisons, communication environment setup, and resource allocation, are discussed explicitly.

The prime motivation behind adopting the federated learning-based approaches was the added privacy and security features that the technology could offer. On the contrary, FL systems are still prone to privacy threats from various mediums. Chapter 8 evaluates the privacy aspect in federated learning and presents a comprehensive analysis of some of the common challenges, proven threats, and attack strategies affecting FL systems.

Chapter 9 evaluates federated learning from a security perspective and presents a comprehensive analysis of some of the common security challenges as well. Various security threats in the form of adversarial attacks, along

with their defenses, are presented. Finally, some of the common measures to improve the security of FL systems are also discussed.

Chapter 10 contains a detailed study of the applicability of blockchain with federated learning on IoT-based systems. A comprehensive survey on healthcare scenarios for blockchain-integrated FL is presented for both edge and fog-based systems, and the merits and demerits of the end systems are also stated.

As federated learning models are generated from on-site training at client devices, the quality of data and training determines the performance of the model generated. For improving the learning performance of FL systems, it is essential to recruit better participants and motivate existing ones. Thus, incentivizing client devices has emerged as an effective solution for these issues. Chapter 11 provides a comprehensive review of various incentive mechanisms and incentivization approaches for stimulating data owners to participate in the FL training process.

Chapter 12 focuses on employing protected shot-based federated learning for facial expression recognition. Various approaches currently employed for facial recognition are discussed, along with their pitfalls. The need for FL as an effective alternative with its outcomes is also presented as a validation for the work.



Taylor & Francis

Taylor & Francis Group

<http://taylorandfrancis.com>

## CHAPTER 1

---

# The Evolution of Machine Learning: From Centralized to Distributed

JAYAKRUSHNA SAHOO,<sup>1</sup> AKARSH K. NAIR,<sup>1</sup> and RICHA SHARMA<sup>2</sup>

<sup>1</sup>*Department of Computer Science and Engineering,  
Indian Institute of Information Technology, Kottayam, Kerala, India*

<sup>2</sup>*Department of Mathematics, Computer Science, and Forensics,  
Commonwealth University (Lock Haven University), Lock Haven, PA, USA*

---

## ABSTRACT

Motivated by the increasing concerns over privacy and the gradual growth of newer technologies such as IoT, the past few years have witnessed a shift in the perception of machine learning from centralized to distributed. An emerging distributed learning paradigm, known as federated learning (FL), is gaining popularity due to its features such as increased privacy preservation, data locality, distributed framework, and remote training. The FL network architecture is based on a client-server system, where data is fully present at the client end and the system training takes place there as well. This eliminates privacy and security concerns while ensuring efficient communication. This article initially presents various terminologies and classifications associated with traditional machine learning, and then discusses the need for a paradigm shift. We also provide an overview of the distributed learning literature, followed by an introduction to federated learning. The article further explores common applications, various network architectures, and detailed use cases. Finally, the article concludes with a discussion of some of the open challenges and future research directions in the field of FL.

## 1.1 INTRODUCTION

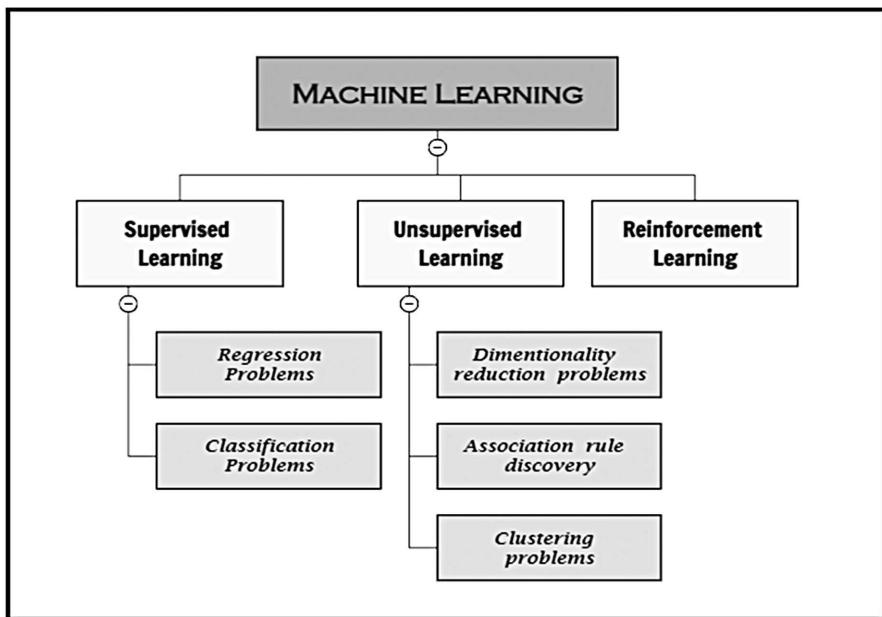
Machine learning technologies have come a long way since their inception and have demonstrated remarkable capabilities in various applications such as virtual assistants, web searches, image and speech recognition, fraud detection, and many more [1]. Machine learning (ML) is partially based on a model that simulates the interaction of neurons in the human brain. The theoretical foundations of this model were introduced in 1943 by neurophysiologist Warren McCulloch and mathematician Walter Pitts, who coined the term “neural network.” The field experienced a major breakthrough during the summer conference held at Dartmouth College in 1956. Over a period of six weeks, a group of ten researchers established standards for this emerging field, which involved working with neural networks, automata theory, and symbol-based reasoning tasks.

The term artificial intelligence (AI) was coined by the research group in an attempt to differentiate it from cybernetics, which at the time was a similar area that also focused on control systems. In later years, the members of the group became the core contributors to the development of the technology, acting as pillars and building a strong foundation. The initial versions of neural networks were not functional or deep. They mainly had a single-layer architecture and were referred to as perceptions, with limited learning capabilities that were often restricted to basic linearly separable problems. In 1969, Marvin Minsky and Seymour Papert published a book titled “Perceptron,” which was one of the first attempts to discuss existing neural networks, their functionalities, and their pros and cons [2]. The book highlighted the shortcomings of the existing neural network systems and emphasized the need for high-quality research in AI, marking the beginning of an evident shift. Although the term “Machine Learning” was coined in 1959, it took several years to standardize and reach the level of visibility we see today.

Basically, ML can be divided into three stages in total, beginning with data processing, followed by model building, and summed up by model deployment and monitoring [3]. The middle process can be termed the “apple of the eye,” as the model is essentially the ML algorithm that uses the data to predict an output through learning. ML and Deep Learning (DL) can never be considered mutually exclusive since even in ML, the model encompasses “DL” capability. In short, we can define DL as just a subcategory of ML that utilizes dense neural networks to comprehend and learn highly complex procedures and data relationships, which can be used to derive outputs

from given data inputs [4]. In such systems, the number of layers and the complexity of the system are directly proportional, meaning that the higher the number of layers, the more complex tasks the system can perform.

By definition, ML is a subset of AI where the system applies algorithms to extract patterns from input data using statistical and mathematical methods and applies them to another set of data to devise solutions. As shown in Figure 1.1, ML can be broadly classified into three main subcategories depending on their autonomy of training:



**FIGURE 1.1** The taxonomy of machine learning.

### 1.1.1 SUPERVISED LEARNING

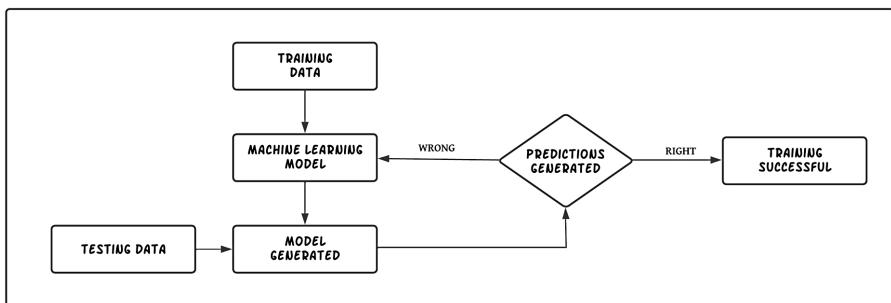
Supervised learning is one of the most popular forms of traditional machine learning (ML). The system attempts to derive relationships among a set of input data and output data that are provided to the system for model training. Supervised learning algorithms mainly deal with two types of problems: classification and regression [5]. Although both approaches make use of predictive modeling principles, they differ in terms of their target variables.

### 1.1.1.1 CLASSIFICATION PROBLEMS

Classification refers to a task where the expected output variables are usually categorical values such as class names or labels. Such models generate observations from a set of data and use the inferences made to make decisions on any given data from a similar context [6]. Some of the common examples are binary classification and multi-class classification problems.

### 1.1.1.2 REGRESSION PROBLEMS

As with regression problems, the target variables usually tend to be continuous values. In regression-based approaches, several models are employed, ranging from very simple models such as linear regression to higher-end models. The functionality of the regression approach is based on a hyperplane, and the objective is to identify the ideal location for the hyperplane so that it can connect the majority of the points [7]. Figure 1.2 shows a basic data flow during model generation using ML algorithms.



**FIGURE 1.2** Schematic diagram of model generation in traditional ML systems.

### 1.1.2 UNSUPERVISED LEARNING

When it comes to unsupervised learning, the only form of input given to the system is raw data, and the approach doesn't make use of any target variable as in supervised learning [8]. The systems function by discovering hidden patterns from the given data without any sort of human input, thus the name. The high level of discrimination skill possessed by the system makes it an ideal choice for applications related to exploratory data analysis, identification, recognition tasks, and so on. Unsupervised learning can be further broken down into three subcategories:

### ***1.1.2.1 CLUSTERING PROBLEMS***

The clustering approach attempts to divide the given data into various subgroups based on certain similarity coefficients. Once separated into subgroups, each member of the group will possess certain levels of similarity with everyone else in the group when compared to every other member of the other group. Such groups are referred to as clusters, and any point that doesn't belong to a cluster due to deviating characteristics is referred to as an outlier [9]. Some of the commonly employed clustering approaches are density-based approaches, hierarchy-based approaches, partitioning approaches, grid-based approaches, and so on.

### ***1.1.2.2 ASSOCIATION RULE DISCOVERY PROBLEMS***

In association rule-based learning approaches, the major factor to be considered is the relationship and dependency of one item or data point in a given dataset with another one. The inferences made are mapped accordingly to infer certain patterns that can be later employed in various use cases. As the name implies, certain rules are used to identify relations, and it varies depending on the application. Some of the main applications of this technology include market basket analysis, frequent item set mining, and so on [10]. In real-life applications, such systems generate predictions about the occurrences of a particular event in relation to the occurrence of another event with respect to an individual's previous history in the same context.

### ***1.1.2.3 DIMENSIONALITY REDUCTION PROBLEMS***

Dimensionality reduction is a machine learning approach derived from statistics. Its aim is to significantly reduce the number of variables in a given problem by identifying a set of principal variables. The process can be divided into two main steps: feature selection and feature extraction. Feature selection involves identifying a smaller subset of features from a larger set, often using procedures like filtering, wrapping, and embedding. Feature extraction reduces the number of dimensions for modeling variables and involves techniques such as eliminating data redundancy, reducing model over fitting, and simplifying complex problem statements. Common methods for dimensionality reduction include factor analysis, low variance filter, high correlation filter, PCA, random forest, and more.

### **1.1.3 REINFORCEMENT LEARNING**

The third and final category is known as reinforcement learning and is widely employed to perform Markov decision processes. Reinforcement learning is similar to a self-learning mechanism with the addition of the entity referred to as an “agent.” Basically, the motto of the system is to maximize the rewards received by the agent and through that, refine the system over multiple iterations [12]. It can be compared to an incentive-based mechanism as they both have a lot of similarities. Additionally, the system makes use of a mapping-based mechanism for individual inputs with their outputs which is similar to unsupervised learning up to a certain degree. However, both can be differentiated by the fact that in the unsupervised learning system, the ultimate aim is to identify similarities and differences whereas in reinforcement learning the goal is to identify a suitable course of action for increasing the total rewards gained by the agent thus optimizing the system performance as well.

## **1.2 IS AI REALLY EMPLOYED IN REAL-LIFE APPLICATIONS?**

AI, generally, and ML and DL, specifically, have found their way into a wide array of applications over time. The reach has been so overwhelming that in most current-day scenarios, we can find such applications either replacing humans or at least being used as technical assistance for skilled human labor. One of the most prominent domains where AI can be seen extensively is in healthcare. In modern healthcare, most diagnostic applications are either employing AI-based systems or already have developed systems capable of surpassing human capabilities. Not just limited to diagnostics, such applications have found their way into various tasks such as robot-aided surgical procedures, medicine vending robots, expert systems, wearable systems, and so on.

## **1.3 FROM CENTRALIZED TO DISTRIBUTED**

Over the years, research on AI has been so intense that we have seen the current research possibilities becoming saturated in terms of optimization aspects. As a result, the focus is gradually shifting towards other aspects of the technology. The traditional methods of AI have been available for a long time, to the point where a centralized model has become synonymous with AI systems. However, with the extensive application of ML technologies in highly complex applications, various issues with the traditional ML

approach have started to arise. Common problems include system security, latency-related issues, and more. Centralized AI also brings concerns regarding device-based issues, such as inadequate computational resources and the need for high-performing systems. A common approach to addressing such issues was either migrating the program to highly complex systems or consolidating multiple smaller systems to create a high-performing system as a whole. However, the first proposed solution posed a major challenge in terms of high economic costs and the static nature of the system, which could result in idle periods when the scale of the running process decreased. The process of enhancing the computational power of the system by adding more resources is commonly known as scaling up. Various strategies are employed for scaling up, including the use of programmable GPUs, application-specific integrated circuits (ASICs), tensor processing units, neuro functional units, and MIMD architecture-based CPUs. Similarly, scaling down, which involves reducing system complexity, is also a significant concern. While there are several methodologies for increasing the processing power of an individual system based on the scale and resource requirements of the machine learning task, there are also reasons to support the scaling down approach or even a hybrid approach that combines both methods, as observed in some high-performance computer system architectures. Therefore, the hardware-based requirements and the challenges associated with increasing resource needs have driven the ML technology to transition from a fully centralized to a distributed architectural paradigm [13].

A globally applicable distributed architecture enabling efficient distribution of traditional machine learning algorithms has always been challenging, as each algorithm has its own working methodologies and requirements. The classical machine learning problem can be divided into two phases: the training phase and the prediction phase. In the training phase, a machine learning model is trained by providing it with a substantial amount of training data and updating it with an ML algorithm. In addition to choosing the right algorithm, it is also ideal to perform hyperparameter tuning to generate optimal results. The final result of the trained model is deployed for performing ML tasks. The second phase, known as the prediction phase, is where the trained model is deployed in real-life instances. The deployed model takes new data as input and generates predictions as the final output. The resource-intensive task is primarily limited to the training phase, while the inference phase can be performed in environments with limited resources as well [14]. Although they are referred to as two different phases, they are never mutually exclusive.

When a sudden shift from centralized to distributed ML occurs, two methods can be employed as part of the distributed approach. One method involves the parallelization of the data, while the other method involves the parallelization of the model. Hybrid approaches can also be used, where both of these methods are employed simultaneously. In the data parallelization approach, the data is partitioned multiple times until the number of partitions matches the number of worker nodes participating in the training. Each worker performs training using the same algorithm but on different datasets. To ensure that the same model is available to all nodes, centralization or replication procedures are used. The goal of using a single model is to generate consistent output [15]. These techniques are applicable to every ML algorithm and use cases where the model works with IID data, which means independent and identically distributed data.

In the model parallelization approach, perfect replicas of the datasets are processed by the nodes that will be working on various segments of the ML model. As a result, the final model generated is an aggregate of all the different model parts. However, a major limitation of the model parallelization approach is its inapplicability to every ML algorithm, as the model parameters usually cannot be split up. An alternative solution is to employ model ensembling approaches where different fragments of the output are aggregated at a location after the training has been done on the same or similar models at different instances. Another factor affecting the distributed machine learning (DML) system architecture is the topology of the system. As the training is performed on highly distributed systems, all the nodes participating in the training should be connected to a central system through a dedicated path to fulfill their sole motto of distributed model generation. The network architecture plays a direct role in deciding the role of the nodes in the training and other network factors such as communication, model deployment, and so on. Thus, DML is just the beginning of a new era of ML approaches from where similar but better and more efficient technologies are generated and taken forward. Collaborative learning and Federated Learning are just some of the examples where distributed ML has served as an inspiration or as a building platform to be developed.

## **1.4 INTRODUCTION TO FEDERATED LEARNING**

The research community has shown keen interest in these aspects over the past few years, leading to the development of a newer technology in AI called federated learning. Federated Learning (FL) was primarily designed to

address the limitations of DML technology, which is essentially a distributed version of centralized AI. In distributed AI, there were certain issues such as the inability to transfer gradients according to the user's discretion, privacy concerns, and various other problems. FL has emerged as a solution to these existing problems and a promising perspective for the future of AI [16]. The fundamental idea behind FL is to keep the data with the data holder and restrict any raw data transfer from the client's end. The concept is based on the notion that private data is relatively more secure than a centralized data entity. As a result, applications that require high security and extended privacy have started adopting this AI approach. While there are several other distributed AI methodologies available, none have gained as much popularity as FL. It is worth mentioning that the widespread use of IoT and Edge computing technologies has also contributed significantly to the outreach of FL. This is due to their high compatibility and ability to complement each other ideally.

#### **1.4.1 WHY DO WE NEED FL?**

Developing ML models with high-performance accuracy is a highly valued asset for companies. In the current technological scenario, with companies possessing a wide variety of heterogeneous client devices, it is never an easy task, especially in centralized learning scenarios. As mentioned previously, with the advent of edge computing and IoT systems, regular centralized ML systems have started facing a lot of shortcomings related to continual learning, data privacy issues, latency issues, computational complexity issues, data heterogeneity, and so on [17]. FL has appeared to be a single-shot solution for all such issues as of now. In traditional ML systems, a central server is used to build the model, and all the available data in the system is aggregated at the server for the training procedure. Such a methodology is ideally suited for networks with powerful server systems in terms of storage space, response rate, processing capability, and every other training aspect [18]. When it comes to mobile computing systems or any low-specification systems, this is not a viable option. Client devices always require quick responses, and with increased data load, it just works the reverse way by adding too much latency to the system, significantly affecting the quality of service and experience. A possible way to overcome this challenge is by shifting the model placement to client devices. However, this approach may hinder "continual learning" as these models are typically generated using complete datasets, which may not be available on individual user systems. Another significant challenge associated with centralized machine learning is related to data privacy.

Aggregating data in a single location is not suitable for data with high privacy requirements. For such applications, this approach contradicts their basic privacy policy, forcing them to either refrain from using machine learning or invest a substantial amount in developing their private models. Additionally, the existence of a data silo always poses a vulnerability issue, as a successful attack on a single point can render the entire network useless [19]. In summary, federated learning overcomes most of the aforementioned challenges by enabling continual learning on client devices while safeguarding user privacy through decentralized data storage, indirectly reducing latency as well.

#### **1.4.2 WHERE CAN FL BE DELIVERED?**

As federated learning operates on a distributed paradigm, the system requires a group of clients and a server (in the typical scenario) to collaboratively train the model. The server acts as a coordinator and oversees the entire process, while the actual training takes place on the client's devices. Therefore, the client devices that perform the training play a crucial role in FL systems as they are responsible for both training and uploading gradients. In terms of the devices being trained on, FL can be broadly classified into two categories:

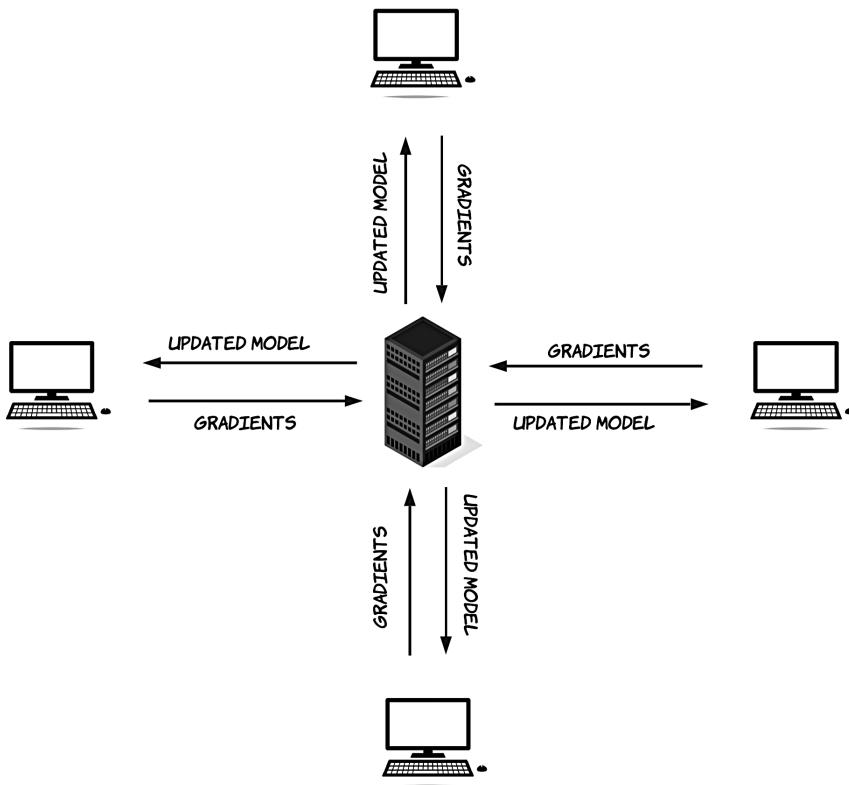
##### **1.4.2.1 CROSS-DEVICE FL**

As the name suggests, cross-device federated learning (FL) refers to the FL training conducted on small user devices. Typically, these user devices are lightweight distributed entities, such as small IoT or mobile computing devices, with limited processing capabilities. Due to their compact nature and limited resources, the data they can contribute to model training is also relatively low. Therefore, in cross-device FL systems, a large number of devices are required to accumulate a substantial volume of data [20]. However, as the number of devices increases, so do the challenges related to device heterogeneity and communication issues. This becomes a major bottleneck for the technology, highlighting the relevance of cross-silo FL systems.

##### **1.4.2.2 CROSS SILO FL**

As with cross-silo federated learning (FL), the client devices are usually data silos, meaning that they possess huge amounts of data and are not singular

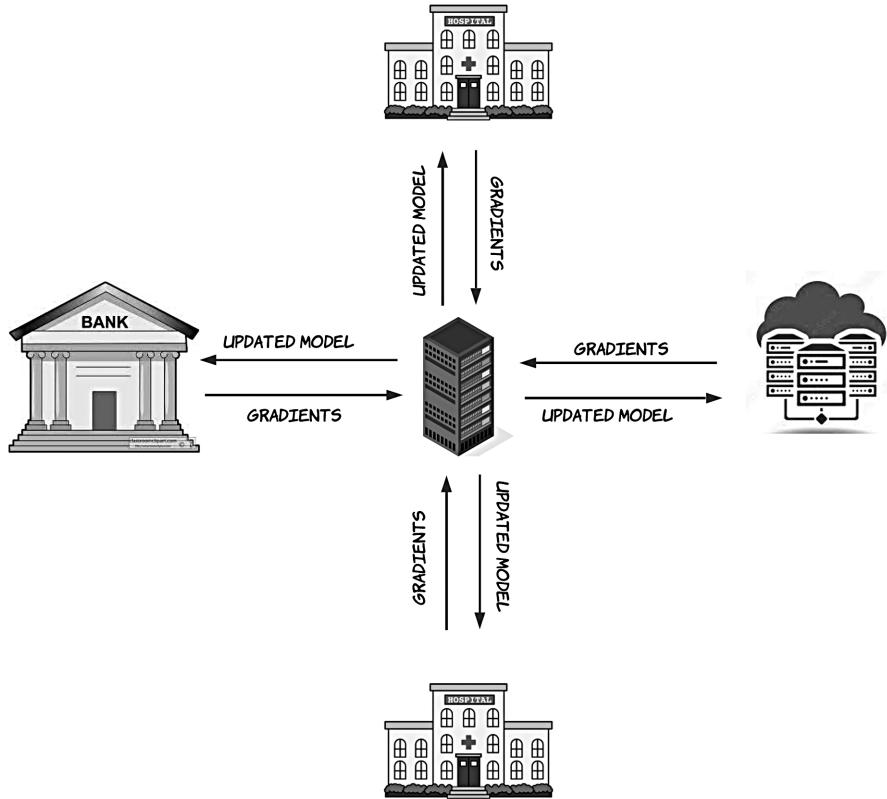
devices. In real-life scenarios, the end-users of cross-silo FL are institutions such as hospitals, banks, and so on [21]. The security aspect of FL is given more importance in such systems, as the client possesses huge volumes of valuable data, thus requiring high levels of caution. Figures 1.3 and 1.4 show the respective images of cross-device and cross-silo FL systems.



**FIGURE 1.3** Cross-device FL architecture.

#### 1.4.3 HOW DOES FL TRAINING WORK?

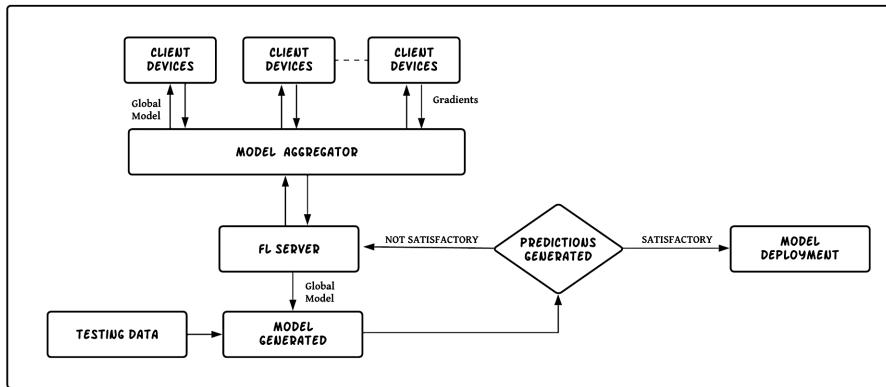
Unlike centralized ML, FL training procedures also involve certain additional steps. The first step in FL training is client selection. In distributed system environments, there may be multiple client devices, and choosing the right set of clients for training is a crucial process. Several factors, such as availability, energy status, and networking capability, are taken into consideration during client selection. Ideally, these clients are expected to actively



**FIGURE 1.4** Cross-silo FL architecture.

participate throughout the training procedure, only exiting after the training is completed. After client selection, the global model is broadcasted to all devices participating in the training. This unified global model is necessary for efficient training. Once the global model is received, each client performs system training on their own. Due to the heterogeneity of data and devices in such systems, the training time is device-dependent and can vary significantly [22]. Following client training, model aggregation is performed at the server end. The clients send their finished model updates to the server, where they are aggregated using techniques such as federated averaging, which is a variation of weighted averaging. The exchange of model updates and the global model between client and server devices continues until a preset accuracy is reached. At this point, the aggregated model is considered the final model and is ready for deployment.

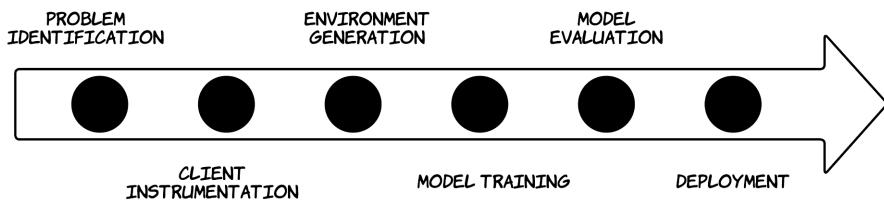
Figure 1.5 illustrates the data flow during FL training, as described above.



**FIGURE 1.5** Federated learning generation phases.

#### 1.4.4 LIFE CYCLE OF A MODEL IN FEDERATED LEARNING

As with any research problem, the building blocks of FL system research also include problem identification. A well-structured problem serves as a strong foundation to build upon. Following problem identification, client instrumentation and prototype simulations are performed. Upon successful completion of the first three steps, the system progresses to the final three steps: model training, model evaluation, and deployment. All the iterative procedures discussed in the previous section fall under these final three steps. A basic visual representation of the aforementioned steps is provided in Figure 1.6.



**FIGURE 1.6** The life cycle of federated learning models.

#### 1.4.5 FEDERATED LEARNING CLASSIFICATIONS

Apart from the classification based on user devices, FL systems are also classified in terms of data distribution. They are mainly classified into three categories: horizontal FL, vertical FL, and federated transfer learning.

### 1.4.5.1 HORIZONTAL FL

Horizontal FL refers to a FL system where the dataset is distributed in a way that all parties share the same feature space but differ in terms of the samples. This type of FL is also known as sample-based FL. Additionally, when the same features are used, it is called homogeneous FL. As the name suggests, the data distribution in such systems is horizontal [23].

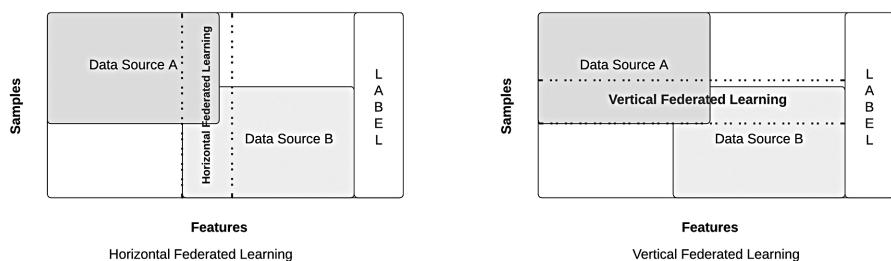
### 1.4.5.2 VERTICAL FL

When it comes to vertical FL, the system behaves in exactly the opposite way compared to horizontal FL. From vertical data distribution to data heterogeneity, they are dissimilar. Additionally, in vertical FL systems, the sample space is shared by datasets, and they differ in terms of feature space, hence the name heterogeneous FL [24]. To efficiently implement vertical FL, other technologies such as private set intersection can be employed for initial feature identification and matching.

### 1.4.5.3 FEDERATED TRANSFER LEARNING

Federated transfer learning is a highly complex topic in the sense that it has yet to have a strong implementation instance as a solid reference. Even though the technology is being studied [25], there is still a lack of extensive literature that comprehensively evaluates the technology. The basic concept behind the technology is analogous to transfer learning but in the context of federated terms.

Figure 1.7 describes the various types of classifications mentioned above also giving a picture of their working as well.



**FIGURE 1.7** Federated learning classifications.

#### 1.4.6 WHERE CAN FEDERATED LEARNING BE USED?

From the current day scenario, FL is gaining vast applicability in various domains. Some of the instances where FL is best suited are as follows:

- Enhancing natural language processing models in robotic process automation solutions by utilizing data from multiple enterprises.
- Improving the accuracy of fraud detection models using data from credit card companies and banks.
- Enhancing personalization and recommendation systems using data from various consumer enterprises.
- Improving computer vision models for healthcare diagnostics using data from multiple hospitals.
- Avoiding the need to migrate databases into a centralized location for machine learning purposes.

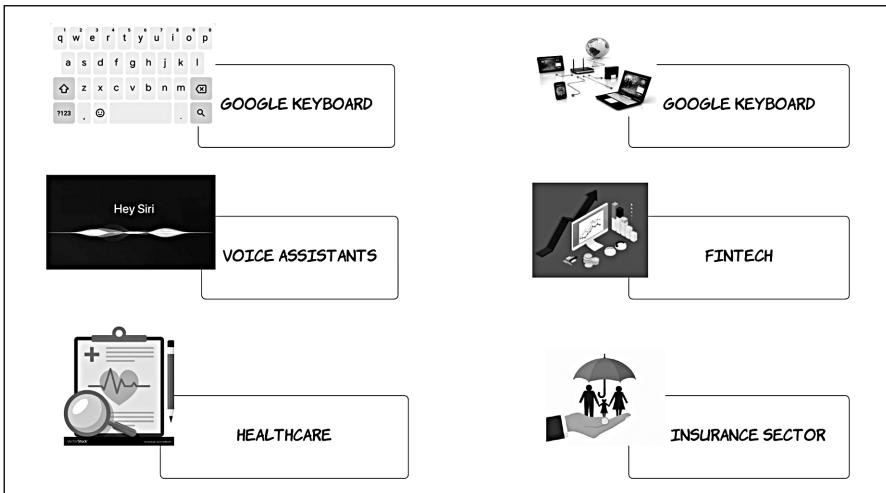
Additionally, certain criteria must be met before categorizing an application or system as a prospective FL system. Figure 1.8 describes the “must-haves” for a system to be able to transform into an FL environment.

- **DISTRIBUTED DATA STORAGE**
- **DATA CANNOT BE SHARED DUE TO SENSITIVITY AND CONFIDENTIALITY**
- **DISTRIBUTED OWNERSHIP OF DATA**
- **SIMILARITY IN DATA IN VARIOUS SOURCES**

**FIGURE 1.8** Federated learning system requirements.

#### 1.5 REAL LIFE APPLICATIONS OF FL

FL is a highly applicable learning paradigm for applications that are privacy-sensitive and have a distributed system architecture as well. With time, FL has proven its mettle in theory as well as real-life application as the perfect alternative for overcoming the shortcomings in traditional ML-based applications. Some of the common use cases of FL can be seen in Figure 1.9.



**FIGURE 1.9** Real-life applications of federated learning.

When it comes to applications of FL, we can categorize them into industry-specific applications as well as technology-specific applications. Some of the common industries that have adopted FL-based technologies can be listed as follows:

### 1.5.1 **FL IN HEALTHCARE**

With the outbreak of the COVID-19 pandemic, the healthcare industry experienced a severe bottleneck due to a lack of resources and automated infrastructure. Developing multiple automated and AI-aided systems can significantly reduce the workload on healthcare workers. However, a major challenge in implementing such systems is the sensitivity of healthcare data. AI applications typically require a large amount of data, and sharing critical information, especially in healthcare, raises privacy concerns. This is where FL-based systems gain popularity in healthcare, as they preserve privacy [26]. In FL, participants such as hospitals and research centers can train models using their own data without transferring it to a third party. This not only protects privacy but also adds value to their data, which may have previously been underutilized. Additionally, the compact nature of FL models enables training and deployment on small-scale devices, making them well-suited for wearable devices and other systems that are shaping the future of the healthcare sector.

### **1.5.2 FL IN FINTECH**

We usually use the term “fintech” to refer to a group of endeavors that are technology-oriented and belong to the financial sector. Fintech is a domain that has to abide by high levels of data protection laws to ensure safe functioning. This ensures high levels of trust between the customers and service providers, as both parties believe that data is kept safe at both ends. When employing traditional machine learning, fintech-based applications faced several issues. Some of the common issues included legal formalities related to personal data distribution, cost of data collection, data transfer cost, and so on. FL is currently employed in fintech as a simple yet effective solution that traditional machine learning has faced in fintech technology. As FL keeps the data local, it reduces the legal formalities and allied technological burden manifold. Additionally, it encourages the usage of lightweight devices, thus increasing the popularity of fintech technology as a whole [27]. FL also can resolve some of the classical problems in fintech. It can easily prevent data breaches and account takeover (ATO) frauds as well. It can also be employed for detecting credit scores and learning users’ footprints in an attempt to prevent fraudulent activities. FL has the complete capability to aid the development of fintech in the coming years. It opens up several innovative solutions for existing problems from the customer’s as well as the service provider’s perspectives. It also helps in building trust among both parties, thus helping in building a long-term healthy relationship as well.

### **1.5.3 FL IN THE INSURANCE SECTOR**

The insurance industry is one such industry that is highly prone to fraudulent activities. Even though several methodologies have been applied to prevent such activities, they could only be limited to a certain degree as frauds find some way to bypass the existing measures. False claims are severe problems faced in the insurance industry. As insurance companies have highly connected networks over various other industrial sectors such as vehicular insurance being connected to healthcare insurance, such fraudulent claims can be controlled to a very high extent if the maximum can be made of such networks and connections. One sector’s data can be cross-checked with another sector before confirming a claim request, thus increasing the authenticity of the procedure. As with other sectors, this data also falls under private data, and sharing it with a third party is highly risky. This is where

the application of FL can provide an efficient solution for the industry [28]. If FL can be applied properly in such instances, companies can identify the real status of such claims without breaching any of the data privacy policies. The algorithms could be trained on private data, thus not causing any harm to the insured's privacy as well as the company's integrity.

#### **1.5.4 APPLICABILITY OF FL IN OTHER INDUSTRIES AND TECHNOLOGIES**

FL is currently being adopted in the Smartphone industry as well. By studying the usage patterns of devices, manufacturers and programmers are finding ways to develop apps that better cater to individual users. Due to privacy concerns, many users are hesitant to share their data. However, FL allows for on-site training and prediction generation without compromising user privacy. The advertising industry heavily relies on personalization, as each user's preferences can vary. In today's scenario, users are cautious about sharing personal data directly, but unknowingly reveal their opinions and related information through virtual platforms like social networking and e-commerce websites. FL can be employed by the advertising industry to acquire data from such platforms and generate better results tailored to individual needs. Additionally, FL is currently being used in the autonomous vehicle industry due to its ability to generate real-time predictions and results. Real-time data can be gathered from multiple sources in such systems, and the real-time processing nature of FL enables it to perform continuous learning and faster, more efficient decision-making in automotive applications. This ensures a better and safer experience in automobiles. The applicability of FL is not limited to the few use cases mentioned above but can also be applied to several other technologies. Currently, the combination of FL with other technologies such as blockchain, IoT networks, and others is being tested for functionality. The future of AI is expected to be dominated by FL, and we may see FL being used in a wide variety of applications, similar to how AI is used today.

### **1.6 CHALLENGES AND FUTURE DIRECTIONS**

With all the pros being presented well, the authors may get a feeling that applications based on FL are idealistic with near-zero cons, which is not the case. As with any other technology, FL is also presented with a set of

hurdles or challenges to be overcome for the overall development of the technology. Some of the most prominent challenges are directly related to the privacy-preserving aspect of the technology. Even though presented as a highly secure alternative for traditional ML, FL systems are also prone to security breaches through a wide variety of attacks. Therefore, dedicated security measures need to be incorporated into the system to ensure the claimed security feature. Apart from this, another issue faced is due to the sheer heterogeneity of the system.

Even though system heterogeneity is permissible in FL systems, some systems are too heterogeneous to question the basic existence of the system itself for several reasons. When presenting heterogeneity, a closely related factor can also be brought up, which is the unreliability of client devices. Such a system comprises various training devices, making a client device highly unreliable in the sense that it may not even participate in the entire training process. This can be due to reasons such as device mobility, power issues, and so on. Mitigating the aforementioned shortcomings itself has a huge scope for future research in FL [29]. Additionally, future research directions may include steps to improve system efficiency and effectiveness, enhanced privacy preservation and attack mitigation schemes, greater decentralization through multi-center systems, and so on. With the ever-advancing trend of compact and mobile systems, the need for technologies like Federated Learning is evident. It is just a question of whether FL will be able to evolve to meet the changing requirements, as traditional ML did. If possible, FL is here to stay; otherwise, it will join a long list of technologies that have perished due to their inability to evolve and meet the mark.

## 1.7 CONCLUSION

In recent times, FL has been demonstrated as an emerging distributed learning paradigm that is capable of utilizing lightweight devices such as mobile phones, IoT devices, and so on to the best of their capabilities. Growing awareness about data privacy and security, along with the security issues associated with traditional ML systems, has also boosted the growth of FL as a privacy-preserving alternative for traditional ML. Additionally, it also facilitates features such as data localization, reduced communications, and so on, adding to its popularity. Extensive research is being conducted over various domains, and attempts to integrate the technology with various technologies are in progress. Through this article, we attempt to present some of the basics of traditional ML, FL, and the rationale behind shifting

from a centralized architecture to a distributed one. We also provide a basic overview of the predecessors of FL and later on, various terminologies associated with FL as well. We conclude the work by presenting some real-life use cases along with a few challenges and future perspectives associated with the technology.

## KEYWORDS

- **distributed computing**
- **internet of things**
- **artificial intelligence**
- **distributed intelligence**
- **edge intelligence**
- **collaborative learning**
- **federated learning**

## REFERENCES

1. Simeone, O., (2018). A very brief introduction to machine learning with applications to communication systems. *IEEE Transactions on Cognitive Communications and Networking*, 4(4), 648–664.
2. Haglin, J. M., Genesis, J., & Adam, E. M. E., (2019). Artificial neural networks in medicine. *Health and Technology*, 9(1), 1–6.
3. El Naqa, I., & Martin, J. M., (2015). What is machine learning? In: *Machine Learning in Radiation Oncology* (pp. 3–11). Springer, Cham.
4. Jakhar, D., & Ishmeet, K., (2020). Artificial intelligence, machine learning, and deep learning: Definitions and differences. *Clinical and Experimental Dermatology*, 45(1), 131, 132.
5. Singh, A., Narina, T., & Aakanksha, S., (2016). A review of supervised machine learning algorithms. In: *2016 3rd International Conference on Computing for Sustainable Global Development (INDIACoM)* (pp. 1310–1315). IEEE.
6. Sen, P. C., Mahimarnab, H., & Mitadru, G., (2020). Supervised classification algorithms in machine learning: A survey and review. In: *Emerging Technology in Modeling and Graphics* (pp. 99–111). Springer, Singapore.
7. Jui-Chan, H., Kuo-Min, K., Ming-Hung, S., & Bi-Min, H., (2020). Application and comparison of several machine learning algorithms and their integration models in regression problems. *Neural Computing and Applications*, 32(10), 5461–5469.

8. Alloghani, M., Al-Jumeily, D., Jamila, M., Abir, H., & Ahmed, J. A., (2020). A systematic review on supervised and unsupervised machine learning algorithms for data science. *Supervised and Unsupervised Learning for Data Science*, 3–21.
9. Kassambara, A., (2017). *Practical Guide to Cluster Analysis in R: Unsupervised Machine Learning* (Vol. 1). STHDA.
10. Mazid, M. M., Shawkat, A. A. B. M., & Kevin, S. T., (2009). A comparison between rule-based and association rule mining algorithms. In: *2009 Third International Conference on Network and System Security* (pp. 452–455). IEEE.
11. Kramer, O., (2013). *Dimensionality Reduction with Unsupervised Nearest Neighbors* (Vol. 51). Berlin: Springer.
12. Oh, J., Matteo, H., Wojciech, M. C., Zhongwen, X., Hado, P. V. H., Satinder, S., & David, S., (2020). Discovering reinforcement learning algorithms. *Advances in Neural Information Processing Systems*, 33, 1060–1070.
13. Gupta, O., & Ramesh, R., (2018). Distributed learning of deep neural network over multiple agents. *Journal of Network and Computer Applications*, 116, 1–8.
14. Froelicher, D., Troncoso-Pastoriza, J. R., Apostolos, P., Sinem, S., Joao Sa, S., Jean-Philippe, B., & Jean-Pierre, H., (2021). Scalable privacy-preserving distributed learning. *Proceedings on Privacy Enhancing Technologies*, 2021(2), 323–347.
15. Acharya, J., Chris De, S., Dylan, F., & Karthik, S., (2019). Distributed learning with sub linear communication. In: *International Conference on Machine Learning* (pp. 40–50). PMLR.
16. Abdul, R. S., Hanine, T., Ould-Slimane, H., Azzam, M., Chamseddine, T., & Mohsen, G., (2020). A survey on federated learning: The journey from centralized to distributed on-site learning and beyond. *IEEE Internet of Things Journal*, 8(7), 5476–5497.
17. Nair, A. K., Chinju, J., & Jayakrushna, S., (2022). Implementation of intelligent IoT. In: *AI and IoT for Sustainable Development in Emerging Countries: Challenges and Opportunities* (pp. 27–50). Cham: Springer International Publishing.
18. Yang, Q., Yang, L., Yong, C., Yan, K., Tianjian, C., & Han, Y., (2019). Federated learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 13(3), 1–207.
19. Bonawitz, K., Hubert, E., Wolfgang, G., Dzmitry, H., Alex, I., Vladimir, I., Chloe, K., et al., (2019). Towards federated learning at scale: System design. *Proceedings of Machine Learning and Systems*, 1, 374–388.
20. Karimireddy, S. P., Martin, J., Satyen, K., Mehryar, M., Sashank, R., Sebastian, U. S., & Ananda, T. S., (2021). Breaking the centralized barrier for cross-device federated learning. *Advances in Neural Information Processing Systems*, 34, 28663–28676.
21. Huang, Y., Lingyang, C., Zirui, Z., Lanjun, W., Jiangchuan, L., Jian, P., & Yong, Z., (2021). Personalized cross-silo federated learning on non-IID data. In: *AAAI* (pp. 7865–7873).
22. Zhang, C., Yu, X., Hang, B., Bin, Y., Weihong, L., & Yuan, G., (2021). A survey on federated learning. *Knowledge-Based Systems*, 216, 106775.
23. Feng, S., & Han, Y., (2020). *Multi-Participant Multi-Class Vertical Federated Learning*. arXiv preprint arXiv:2001.11154.
24. Yang, Q., Yang, L., Tianjian, C., & Yongxin, T., (2019). Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(2), 1–19.
25. Liu, Y., Yan, K., Chaoping, X., Tianjian, C., & Qiang, Y., (2020). A secure federated transfer learning framework. *IEEE Intelligent Systems*, 35(4), 70–82.

26. Nair, A. K., Jayakrushna, S., & Ebin, D. R., (2023). Privacy-preserving federated learning framework for IoMT-based big data analysis using edge computing. *Computer Standards & Interfaces*, 103720.
27. Dash, B., Pawankumar, S., & Azad, A., (2022). Federated learning for privacy-preserving: A review of PII data analysis in fintech. *International Journal of Software Engineering & Applications*, 13(4), 1–13.
28. Rahman, A., Md Hossain, Ghulam, M., Dipanjali, K., Tanoy, D., Muaz, R., Md Khan, S. I., et al., (2022). Federated learning-based AI approaches in smart healthcare: Concepts, taxonomies, challenges, and open issues. *Cluster Computing*, 1–41.
29. Mammen, P. M., (2021). *Federated Learning: Opportunities and Challenges*. arXiv preprint arXiv:2101.05428.

## CHAPTER 2

---

# Types of Federated Learning and Aggregation Techniques

S. SHAILESH<sup>1</sup> and JOSEPH JAMES<sup>2</sup>

<sup>1</sup>*Department of Computer Science, Cochin University of Science and Technology, Ernakulam, Kerala, India*

<sup>2</sup>*Department of Computer Science, Mar Gregorios College of Arts and Science, Alappuzha, Kerala, India*

---

### ABSTRACT

The contemporary advancements in the areas of machine learning and artificial intelligence (AI) open up many opportunities to adopt these technologies in real life. However, concerns also arise when dealing with confidential user data. In traditional machine learning models, the distributed learning models themselves fail to maintain the privacy of the autonomous data from each client. They collect all the data to a single point and perform the learning process at the central server. Researchers in this field have come up with a promising solution called Federated Learning, which ensures the privacy of each client's data. The federated learning approach is capable of training machine learning models for each client independently and combining those independent models to build a general global model. This chapter explains different variants or types of federated learning approaches such as 'Horizontal Federated Learning,' 'Vertical Federated Learning,' and 'Federated Transfer Learning'. Horizontal federated learning utilizes different features of the same sample set, while in vertical federated learning, all the clients utilize different sets of samples with a common set of features. In federated transfer learning, the clients have the freedom to use any sample set and any feature

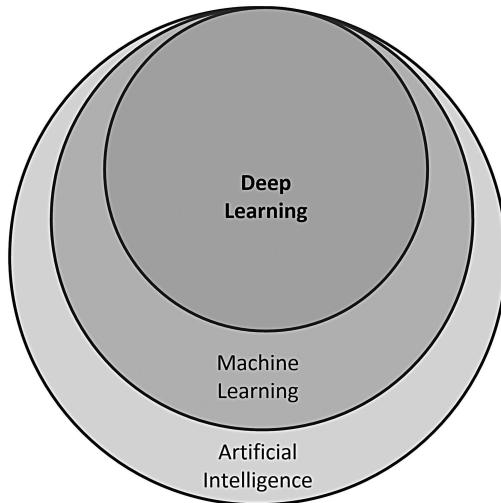
set. This chapter also explains different aggregation techniques used in federated learning architectures. The idea of aggregation is to collect the model parameters from all autonomous clients and combine them to create a single global model. This task requires more technical soundness as it is the most critical phase in federated learning. The three specific methods: One Model Selection (OMS), All Model Averaging (AMA), and Best Model Averaging (BMA) are elaborated with detailed algorithms and use cases. Along with the classical aggregation techniques, the article explains three secure aggregation protocols: Homomorphic encryption-based aggregation, Blockchain-based aggregation, and TEE-based aggregation. All the techniques are explained in the context of homogeneous federated learning architecture, which uses the same learning algorithm specifications in all client endpoints.

## 2.1 INTRODUCTION

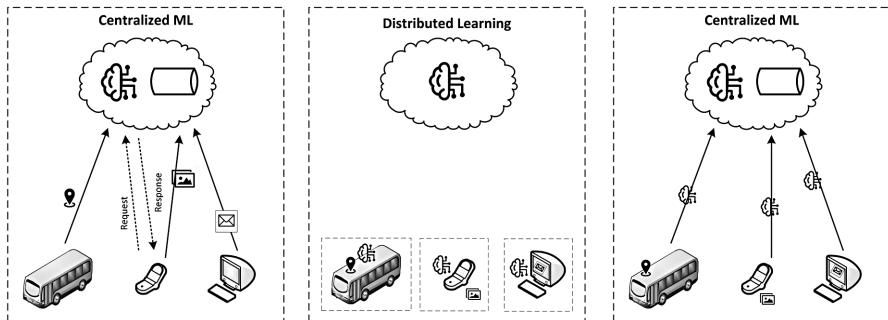
In our current technical scenario, AI and machine learning are becoming increasingly important at a rapid pace. AI is a prominent branch of computer science that focuses on imbuing intelligence into a system. Machine learning, as depicted in Figure 2.1, is a subdomain of AI that relies on the concept of “Learning from experience.” It involves studying algorithms to enhance performance in practical tasks using past data and experience.

In the traditional centralized machine learning concept [1], the client who participates in the learning process is connected with the server, and the client needs to share the dataset with the server as a part of the process. As data is very critical in all aspects, it must be kept secure and its privacy must be a concern. Therefore, every client who participates in the learning process is not interested in sharing their data due to privacy concerns. This situation points to an important limitation of the centralized machine learning process. Another way of handling large volumes of data scattered over different sites is through distributed learning. Distributed machine learning (DML) refers to multimode machine learning algorithms that improve system performance, accuracy, and feasibility in handling a large amount of data used in learning. This system’s feasibility can control the learning error and is also sufficient for the learning process compared to using other complex methods [2]. DML is very useful in decision-making from a large dataset. It is convenient for researchers, companies, and scholars to fulfill their needs in decision-making by analyzing the system results. This chapter discusses one of the state-of-the-art techniques called federated learning, which has high potential for

solving the privacy issues that arise while building a machine learning model. A gist of the differences between centralized machine learning, DML, and federated learning is given in Figure 2.2. Particularly, the core discussion focuses on different types of federated learning models and the aggregation techniques used for model aggregation.



**FIGURE 2.1** AI vs. ML vs. DL.



**FIGURE 2.2** Comparison of centralized ML, distributed learning, and federated learning.

## 2.2 FEDERATED LEARNING

Federated learning [3] overcomes problems pointed out by the traditional learning method. Unlike the traditional method, the clients do not need to

share the data with the server. The clients keep their dataset for training. Some selection criteria for the participants include a smart device, bandwidth, etc. In general, all clients can participate in the training, but there are some exceptions. In this technique, the clients participate in learning to develop their model, i.e., a local model [4], and each client sends that developed model to the central server. The central server receives all the local models and develops an aggregate model, i.e., a global model [4]. The global model is sent to all the clients in participation, and the clients update their models with the global model and then send it to the server for aggregation. This process will continue until a better model is accepted for all datasets. Figure 2.3 shows the concept of federated learning.

Figure 2.4 illustrates the concept of federated learning between a single client and a central server. It is the same for  $n$  clients in the learning process. Let  $\mathcal{O}^n$  be the particular iteration of the local server and  $M_f^n$  be the particular iteration during model aggregation of the central server. The  $\mathcal{O}^n$  is the collection of local models  $(M_1, M_2, M_3, \dots, M_n)$  developed by the clients, then the client models will be as follows  $\mathcal{O}^1(M_1^1, M_1^2, M_1^3, \dots, M_1^n)$ ,  $\mathcal{O}^2(M_2^1, M_2^2, M_2^3, \dots, M_2^n)$ ,  $\mathcal{O}^3(M_3^1, M_3^2, M_3^3, \dots, M_3^n)$  and  $\mathcal{O}^n(M_n^1, M_n^2, M_n^3, \dots, M_n^n)$ . Let  $N$  be the client who participated in the learning. For each client, after  $then^{th}$  iteration  $\mathcal{O}^n$  contains a set of models developed by the client and the model aggregation will be  $M_f^n = \mathcal{O}^n(M_1^n, M_2^n, M_3^n, \dots, M_n^n)$ . The  $M_f^n$  is again used for further iteration. This process continues until the  $n^{th}$  iteration. The aggregated model developed in the  $n^{th}$  iteration is used by the client to update the local models, which are acceptable for all data.

There are some advantages to federated learning:

- It helps in decision-making with a wide sense of reusability;
- Federated learning addresses concerns about the safety of the dataset by developing a model to share with the server instead of the entire dataset. This approach makes the client more comfortable.

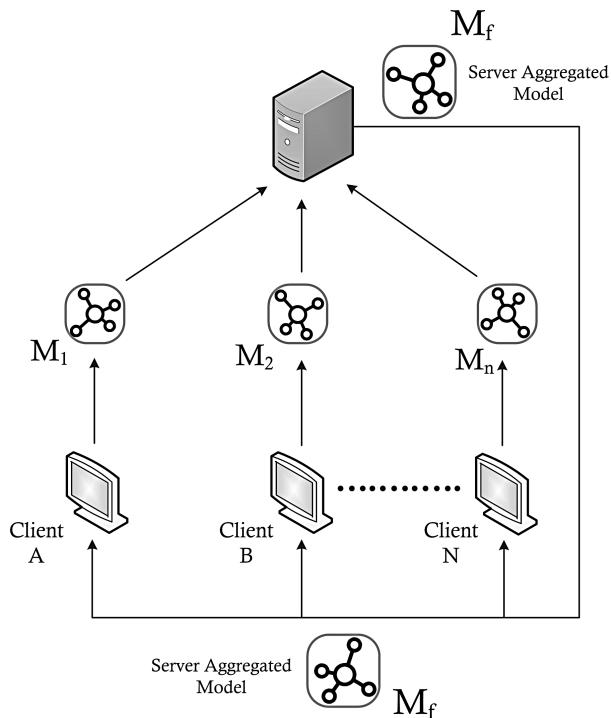
There are some disadvantages, that are pointed out:

- Differences in the devices;
- The limitation in the federated network bandwidth presents a significant challenge in federated learning.

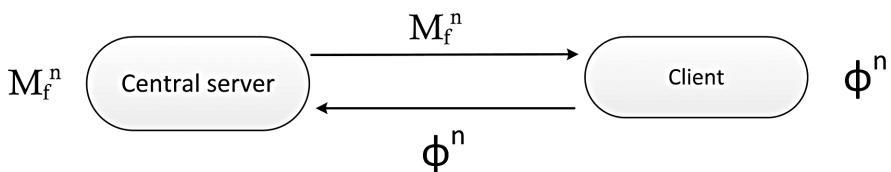
There are 4 main acts in federated learning [5]:

- Model training on the client side with the client dataset.
- Client-side encrypts the local model and shares it with the server.

- The central server produces a global model by aggregating the models sent by the clients who participated in the training.
- The central server dispatches the global model to the clients for further training.



**FIGURE 2.3** The concept of federated learning.



**FIGURE 2.4** The architecture of federated learning for a single client.

## 2.3 TYPES OF FEDERATED LEARNING

Based on the sample selection from the dataset, there are three different types of federated learning.

- i. Horizontal federated learning;
- ii. Vertical federated learning; and
- iii. Federated transfer learning.

Let us consider X and Y as arbitrary variables for the feature space, and i and j as the label space of clients. D denotes the client dataset, and S represents the sample space ID. The training dataset (S, X, Y) consists of X features, Y labels, and S sample IDs.

### **2.3.1 HORIZONTAL FEDERATED LEARNING**

In Horizontal Federated Learning [7], the client selects different samples with the same features for training. The intersection features of the samples are used for training, which is referred to as sample-based federated learning. Figure 2.5 illustrates the concept of sample selection. This approach is also known as homogeneous federated learning. The expression for the given federated learning is provided below:

$$X_i = X_j, Y_i = Y_j, S_i \neq S_j \forall (D_i), i \neq j \quad (1)$$

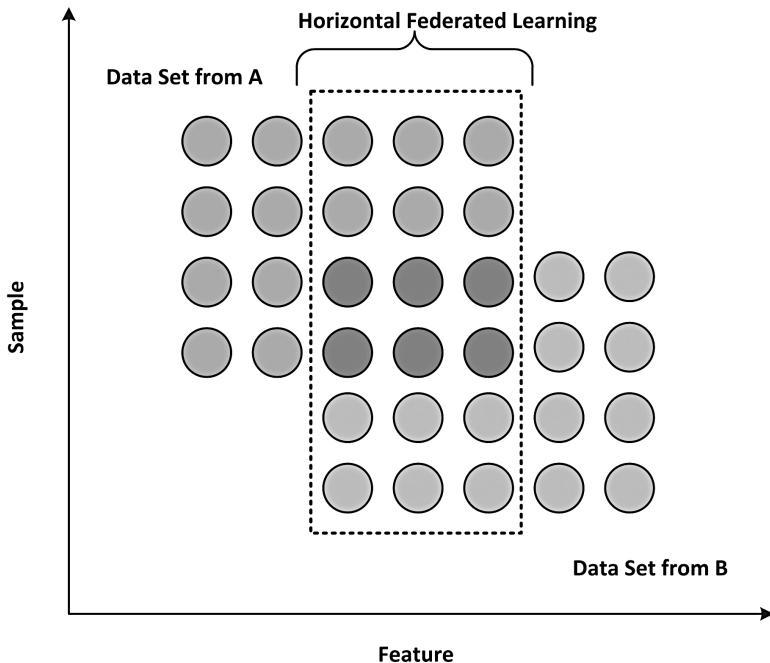
*Example:* We are familiar with social media sites in different languages. If a user needs to change the language, the client changes it locally and sends the updated language to the server. The server then updates the changes and sends them back to the client. The features remain the same but differ in samples. Figure 2.3 also illustrates the concept of horizontal federated learning.

### **2.3.2 VERTICAL FEDERATED LEARNING**

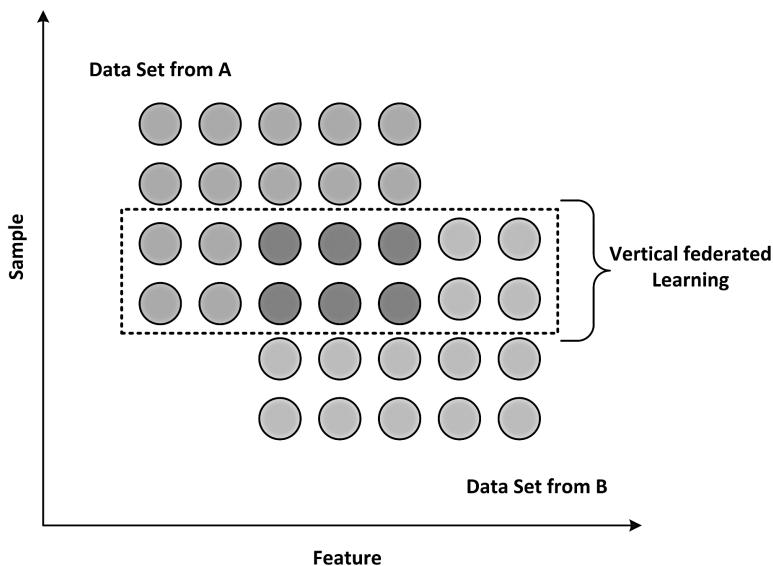
Vertical Federated Learning [7] is a feature-based federated learning. In this method, the client uses the same samples with different features from the dataset. Figure 2.6 shows the concept of sample selection in training. It is also referred to as heterogeneous federated learning. The expression for vertical federated learning is given below:

$$X_i \neq X_j, Y_i \neq Y_j, S_i = S_j \forall (D_i), i \neq j \quad (2)$$

As mentioned earlier, it is feature-based learning. However, the clients who participate in the learning are not interested in sharing the data. Initially, they extract the intersected features of the samples and share them by encrypting the content, without compromising the confidentiality of the data. Figure 2.7 illustrates the concept. Based on the shared content, the clients develop their

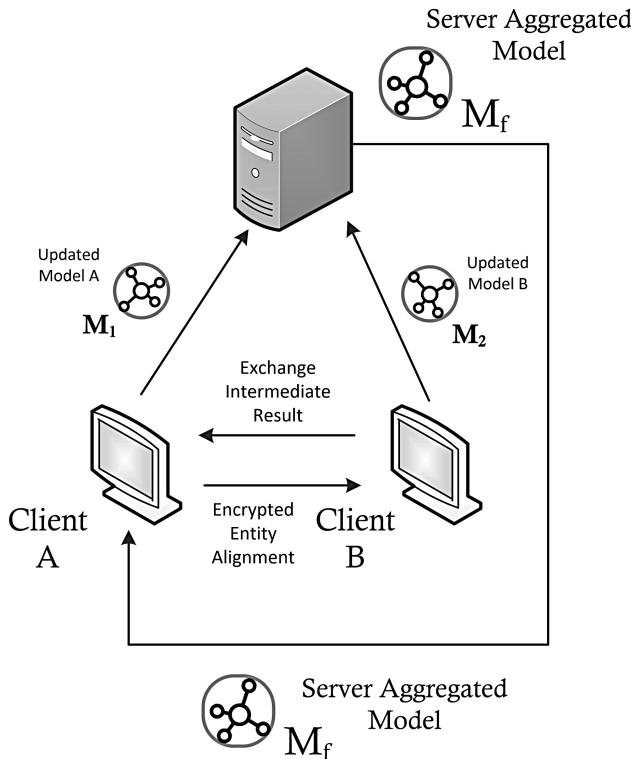


**FIGURE 2.5** Horizontal federated learning.



**FIGURE 2.6** Vertical federated learning.

local models. They send the encrypted models to the server for aggregation. The server decrypts the content and initiates the learning process to update the global model. The encrypted global model is then sent back to the clients to update their local models. The clients update their local models with the decrypted parameters.

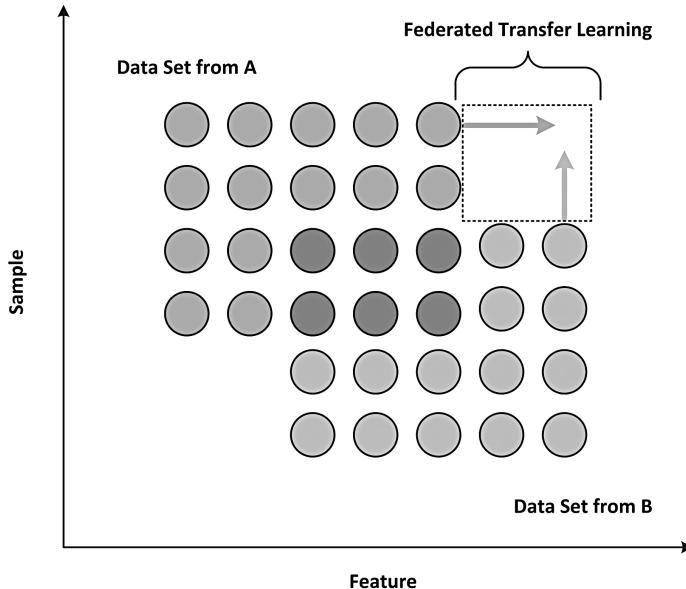


**FIGURE 2.7** The concept of vertical federated learning.

### 2.3.3 FEDERATED TRANSFER LEARNING

It is applicable to samples from the dataset with different sample IDs and different features. This type exhibits some similarities with vertical federated learning. Figure 2.8 illustrates the sample selection for learning, where the client utilizes the common features in the samples. However, the gradient calculation exhibits some differences when compared to vertical federated learning. The following expression is for federated transfer learning.

$$X_i \neq X_j, Y_i \neq Y_j, S_i \neq S_j \forall (D_i), i \neq j \quad (3)$$



**FIGURE 2.8** Federated transfer learning.

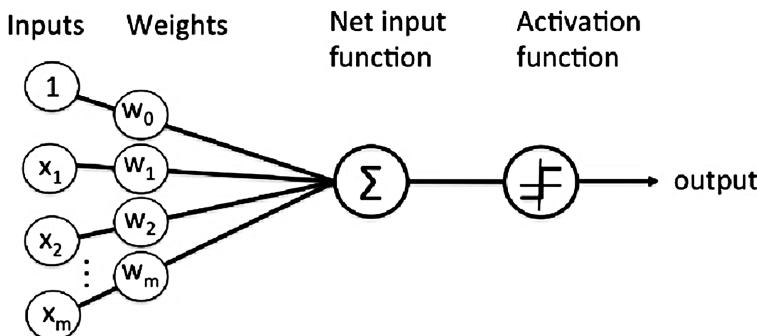
## 2.4 FEDERATED LEARNING AND NEURAL NETWORKS

As explained before, the federated learning technique achieves a machine learning model by using model aggregation techniques on a centralized server, utilizing different models trained on a dataset distributed over many clients. The model of interest must be compatible with the learning processes occurring in the federated learning architecture. Neural networks, which encompass a variety of learning architectures, are well-suited for creating the machine learning models required in federated learning architectures. The following section will provide an overview of the different neural network models and the important components within a neural network that need to be considered when used in a federated learning architecture.

### 2.4.1 PERCEPTRON

A perceptron [8] is a simple neural network model widely used in many classification and regression applications. It is very similar to a biological

neuron. The essential components of a perceptron are inputs and their corresponding weights, a summation function, an activation function, and an output, as shown in Figure 2.9.



**FIGURE 2.9** Perceptron.

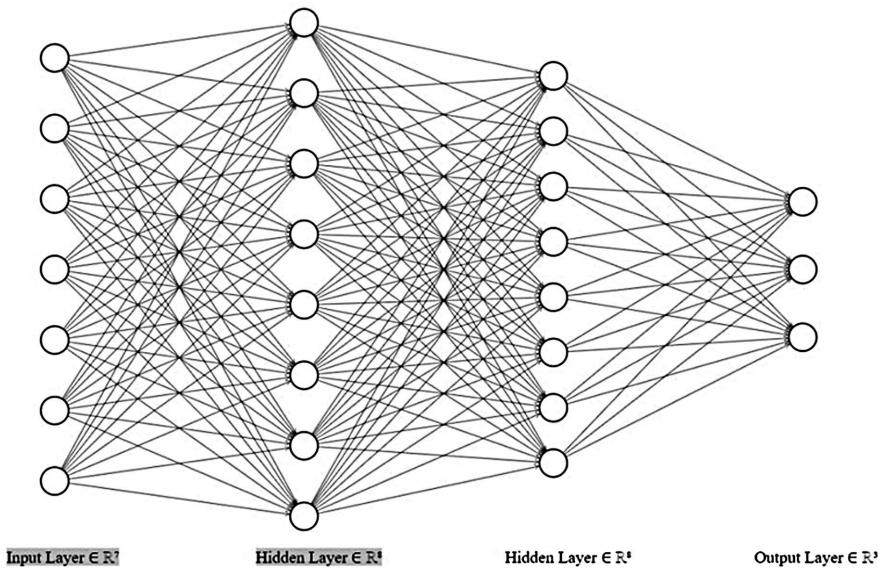
A perceptron generates an output for an input instance by multiplying it with the corresponding weights. These products are then summed and given as an input to an activation function, which generates the final output. Additionally, there is a fixed input, usually ‘1’, multiplied by a special weight called bias, which is included in the summation. The above process can be mathematically represented as:

$$y = f\left(\sum_{i=1}^m w_i x_i + \theta\right)$$

where  $x_i$  is each input,  $w_i$  is the corresponding weight,  $\theta$  is the bias, and  $f$  is the activation function. The learning process that occurs with a perceptron is governed by the perceptron learning algorithm. It takes a labeled dataset with a set of input values represented as numerical vectors and numerical labels. The goal of the perceptron learning algorithm is to adjust the weights and bias in such a way that the error between the predicted outputs and the actual outputs is minimized. Geometrically, the perceptron tries to find the coefficients of a hyperplane, while the bias term provides translational freedom for the hyperplane. However, a major limitation of the perceptron is that it is unsuitable for nonlinear problems. In the context of federated learning architecture, an individual perceptron is learned at each client using autonomous data. Later, the learned model parameters are passed to the central server. Here, the parameters refer to the set of weights  $w_1, w_2, w_3, \dots, w_m$  and the bias  $\theta$ .

### 2.4.2 MORE NEURAL NETWORK MODELS

Considering the wide areas of application and the complex problems included in them, one should move to advanced neural network learning models to tackle them.



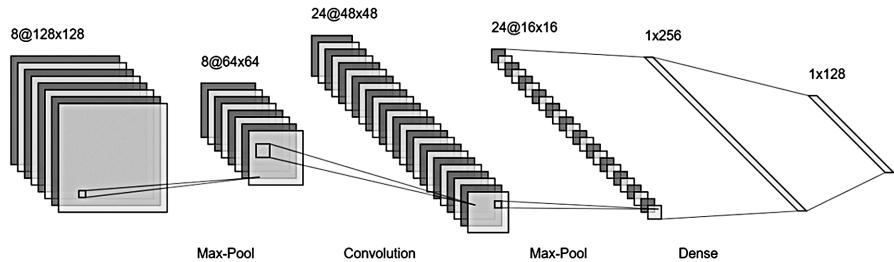
**FIGURE 2.10** Multi-layer perceptron.

Multi-layer perceptrons (MLP) [9] are used to solve nonlinearity. Similarly, for two-dimensional data, convolutional neural networks (CNN) [10] and for sequential data, recurrent neural networks (RNN) [11] are the options. After the learning process, the model parameters for all these models are the set of weights organized as a collection of weight matrices that need to be propagated to the central server.

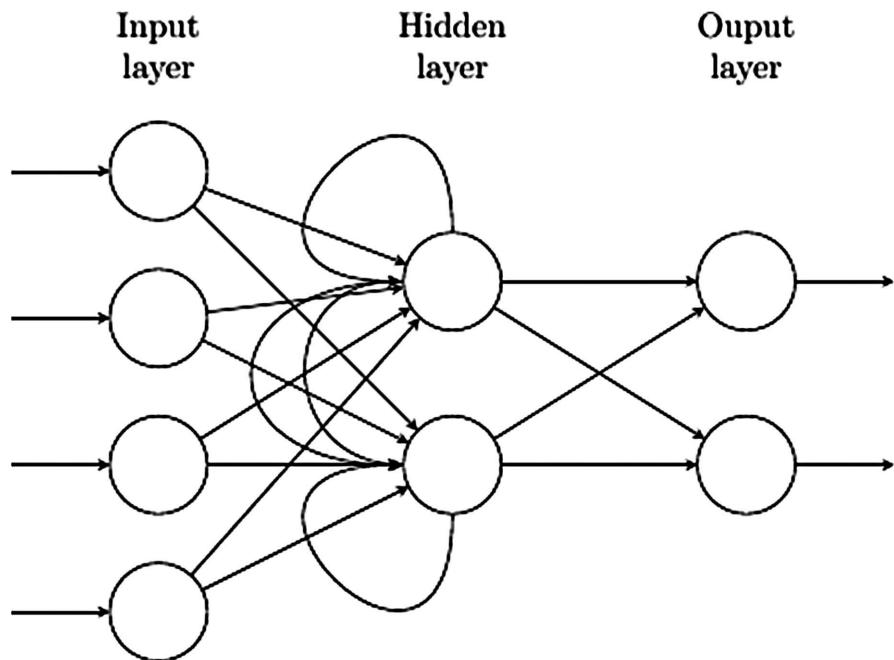
The sample structures of MLP, CNN, and RNN are shown in Figure 2.10, Figure 2.11, and Figure 2.12, respectively.

## 2.5 COMPONENTS OF FEDERATED LEARNING

We have already discussed the concept of federated learning. Let's now discuss the components of federated learning based on Figure 2.13.



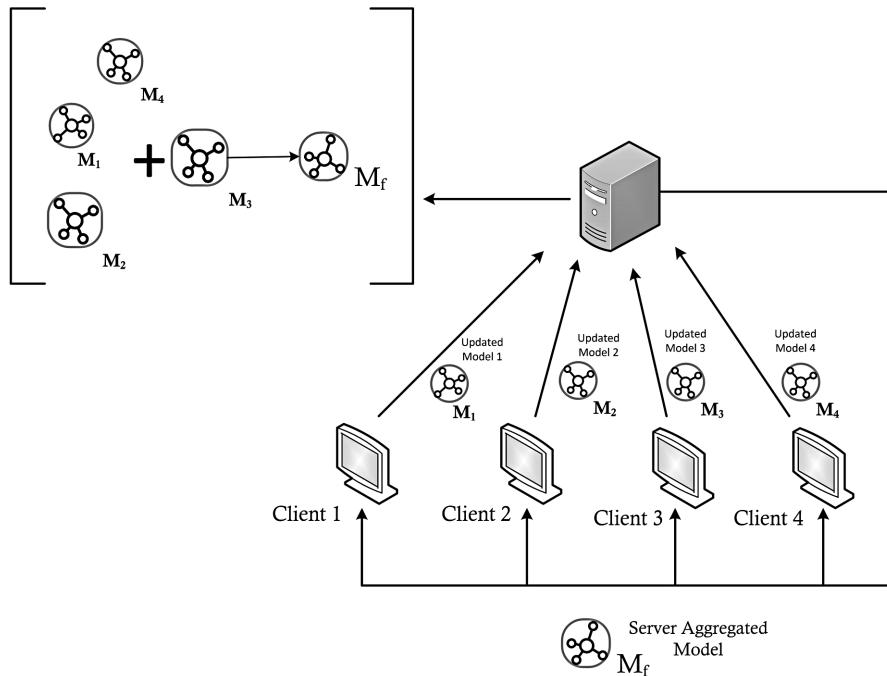
**FIGURE 2.11** Convolutional neural network.



**FIGURE 2.12** Recurrent neural network.

There are five main components in federated learning:

- i. Client;
- ii. Central server;
- iii. Fetching global model;
- iv. Local model training; and
- v. Model aggregation.



**FIGURE 2.13** The structure of federated learning.

### 2.5.1 CLIENT

In federated learning, the client is a device that contains a processed dataset. Here, the clients do not need to send the dataset to the server. Instead of sharing the dataset, clients develop a local model with their dataset. Each model is different from one another.

### 2.5.2 CENTRAL SERVER

One of the main components in federated learning is the central server. The local model developed by the client does not apply to all the datasets. It shows some differences due to the varying samples in the different datasets. The central server is responsible for creating a unified model that applies to all. This is achieved by aggregating all the local models sent by the clients, allowing the central server to develop a global model.

### **2.5.3 FETCHING GLOBAL MODEL**

The clients who participated in the training are active and they hold their trained local model. The global model is an outcome of the central server by taking the aggregate of the local models  $M = (M_1, M_2, \dots, M_n)$  from the clients. Later, this global model  $M_f = G(M)$  is then sent to the clients to update their model.

### **2.5.4 LOCAL MODEL TRAINING**

Local model training occurs on the client side. Each client maintains its own dataset for training. The selection of samples from the dataset is discussed in the previous section. Since clients use different datasets, the models they develop are also different. These models are not applicable to all datasets. To address this issue, we utilize a global model  $M = (M)$ , which is an aggregation of the local models.  $M \leftarrow M - \sigma \mathcal{O}^n(M; b)$  is used to calculate the gradients on the dataset during local model training. Here,  $b$  represents the batch number and  $\mathcal{O}^n$  is used to identify the client. After model training, the models  $(M_1, M_2, \dots, M_n)$  are sent to the server for the aggregation process. Computation is performed using Algorithm1.

### **2.5.5 MODEL AGGREGATION**

The model aggregation is a process in the central server. The central server creates an aggregate model called the global model from the local models  $(M_1, M_2, \dots, M_n)$ . The global model is an updated version for all clients who participated in the training. It addresses any defects present in each local model. Model aggregation techniques will be discussed in the next section.

## **2.6 MODEL AGGREGATION TECHNIQUES**

The aggregation technique is performed by calculating the average of the coefficients of the neural network model. In federated learning, the aggregation technique aids in updating the local model. The central server utilizes the aggregation technique.

### **Algorithm 1**

NN Coefficients Averaging: The N clients are identified by n. B represents the mini-batch size. R represents the number of local epochs. D represents the data allocated to a client.  $\phi^n(M)$  represents the set of local models (i.e.,  $M_1, M_2, M_3, \dots, M_n$ ). Lastly, represents the learning rate.

Input: Global model or update model,  $M_f = G(M)$ .

Output: Clients model after training,  $\mathcal{O}^n(M)$

1. Central Server ( $\phi^n(M)$ ):
2. Model Aggregation ( $\phi^n(M)$ );
3. Client Update ( $M_f = G(M)$ ):
4.  $\beta \leftarrow$  (split  $D_n$  into batches of size  $B$ )
5. For each local epoch i from 1 to R do
  - for batch b  $\beta$  do
  - $M \leftarrow M - \sigma\phi^n(M; b) \equiv W^r \leftarrow W^r - \sigma\phi^n(W^r)$
  - End
6. End
7. Return ( $\phi^n(M)$ ) to a central server.

The local and global models [12] are updated using a gradient descent algorithm. Several techniques are listed here, including model averaging [13], OMS [14], and the best model averaging (BMA).

#### **2.6.1 ALL MODEL AVERAGING (AMA)**

The AMA technique is conducted by taking the average of the coefficients of the neural network model. During training, the client uses Algorithm 1 to develop the local model and then sends it to the central server for the aggregation process. During processing, the central server computes the sum and average of the neurons of the local model using Algorithm 2. The aggregate model is then sent back to the clients who participated in the training to update their local models. The clients use Algorithm 1 to train their local models, while the central server uses Algorithm 2 to aggregate the local models.

$$M_f = AMA \left( \frac{\sum_{i=1}^n \sum_{j=1}^m (W_{i,j}^1 + W_{i,j}^2 + \dots + W_{i,j}^n) i}{n} \right) \quad (1)$$

The above equation  $M_f$  is used to identify the aggregated model. During the aggregation of the models, we sum the weights of all models and then divide it by the total number of models (n). ‘I’ represents the layer. ‘J’ represents the neuron of each layer, and ‘W’ represents the weight of the model.

### **Algorithm 2**

Model Aggregation (AMA): Global model structure Initialized:  $G(M)$ ; W is denoting model weights; Central server receives four client models in our research, where  $\phi^n = (M_1, M_2, M_3, \dots, M_n)$

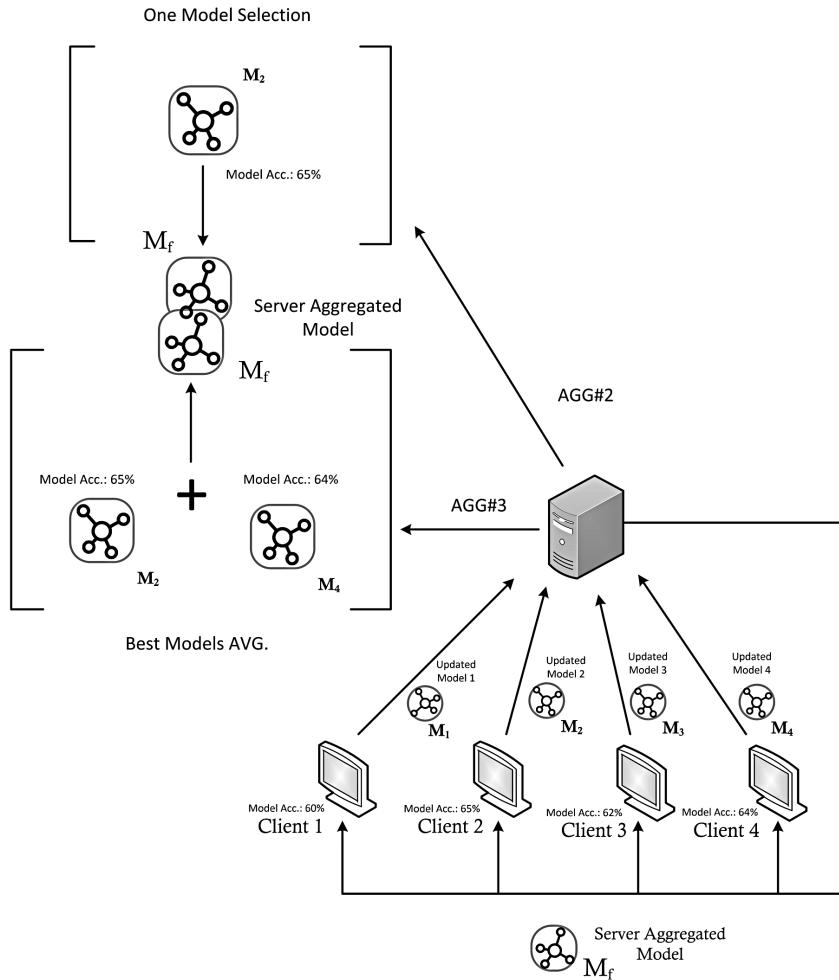
Input: Client’s trained model  $\phi^n (M)$ .

Output: Global model or update,  $M_f = G (M)$ .

1. Model Aggregation ( $\phi^n (M)$ ):
2. Model List =  $[M_1, M_2, M_3, \dots, M_n]$
3.  $|M|$  = Number of total models.
4. For each I in P do
  - For each j in Q do
    - o  $\text{avg} = M_{W(i,j)}^1 + M_{W(i,j)}^2 + M_{W(i,j)}^3 + \dots + M_{W(i,j)}^n$
    - o  $G(M_{W(i,j)}) = \frac{\text{avg}}{|M|}$
  - End
5. End
6. Return  $M_f \equiv G(M)$  to clients.

### **2.6.2 ONE MODEL SELECTION (OMS)**

The OMS technique is based on the performance of the models. The procedures are the same for developing the global model, but some differences are pointed out. Instead of computing all the computational values, they select the model with the highest accuracy for developing the global model. In this technique, the clients send not only the model but also the computed accuracy to the server. Then the client trains their local model with the global model. Figure 2.14 shows how they select the model for the aggregation process, and Algorithm 3 illustrates the OMS technique.



**FIGURE 2.14** The structure of OMS and BMA.

### Algorithm 3

**Model Aggregation (OMS):** Global model structure initialized:  $G(M)$ ;  $W$  is denoting model weights; Central server receives four client models in our research, where  $\phi^n = (M_1, M_2, M_3, \dots, M_n)$

Input: Client's trained model  $\phi^n (M)$ .

Output: Global model or update,  $M_f = G (M)$ .

1. Model Aggregation ( $\phi^n(M)$ )
2.  $A = [M_{acc}^1, M_{acc}^2, M_{acc}^3, \dots, M_{acc}^n]$
3. set  $G(M) = \max(A)$
4. return  $M_f \equiv G(M)$  to clients.

Eqn. (2) performs the same as Algorithm 3, as both help to sort out the best model based on the performance of the local model.

### 2.6.3 BEST MODEL AVERAGE

As the name suggests, it calculates the average of the best models. It shares some similarities with the AMA technique. The process of computing the average of the models is the same as the AMA technique. The difference lies in the fact that the client sends the model accuracy performance to the central server. We select the best model from the local models. Figure 2.14 illustrates the framework. This model may be a combination of two models or half of the models. By utilizing Algorithm 4, the sum and average of the neuron weights are calculated, and then the central model aggregates and the global model for the local model training. The equation for the calculation is provided below:

$$M_f = (W_i^1, W_{i,j}^2, \dots, W_{i,j}^n) \quad i, j \quad (2)$$

$$M_f = \frac{BMA \left( \sum_{i=1}^n \sum_{j=1}^m (W_{i,j}^1 + W_{i,j}^2 + \dots + W_{i,j}^n) \right) i, j}{(best \ 2 \ models \ out \ of \ 4)} \quad (3)$$

Eqn.(3) sorts the local model from the local models developed by all the participants. Around 50% of the models were selected for the next iteration. These are the models with high model accuracy. By using Eqn.(1) of AMA for the aggregate model.

#### **Algorithm 4** Model Aggregation:

Global model structure initialized:  $G(M)$ ;  $W$  is denoting model weights; Central server receives four client models in our research, where  $\phi^n = (M_1, M_2, M_3, \dots, M_n)$

Input: Client's trained model  $\phi^n(M)$ .

Output: Global model or update,  $M_f = G(M)$ .

1. Model Aggregation ( $\phi^n(M)$ )
2. A = Sorted [
3.  $|M|$  = Number of total models.
4.  $k = \frac{|M|}{2}$
5. for each i in P do
  - for each j in Q do
    - $G(M_{w(i,j)}) = \frac{A[0]_{w(i,j)} + A[1]_{w(i,j)} + \dots + A[k]_{w(i,j)}}{k}$
  - End
6. End
7. Return  $M_f \equiv G(M)$  to clients

## 2.7 SECURE AGGREGATION PROTOCOL

### 2.7.1 HOMOMORPHIC ENCRYPTION-BASED AGGREGATION

Homomorphic Encryption-based aggregation utilizes a powerful encryption technique throughout the aggregation process. Homomorphic Encryption is a form of public key encryption, where a private key and a public key are used to encrypt and decrypt the data. The unique feature of Homomorphic encryption is its ability to evaluate functions over encrypted data. In the given example, each client encrypts their models  $x_1, x_2, \dots, x_n$  using Homomorphic encryption. Then, the aggregation function ‘f’ is applied to the encrypted models. The result of the aggregation is then decrypted to obtain the global model. Since this aggregation technique employs Homomorphic encryption, the decrypted global model at the server will be equivalent to the result of the aggregation function applied to the actual models. This is illustrated in the following equation:

$$\begin{aligned} C &= f(E_{k1}(x_1), E_{k2}(x_2), \dots, E_{k3}(x_3)) \\ y &= f(x_1, x_2, \dots, x_n) \\ y &\Leftrightarrow D_k(C) \end{aligned}$$

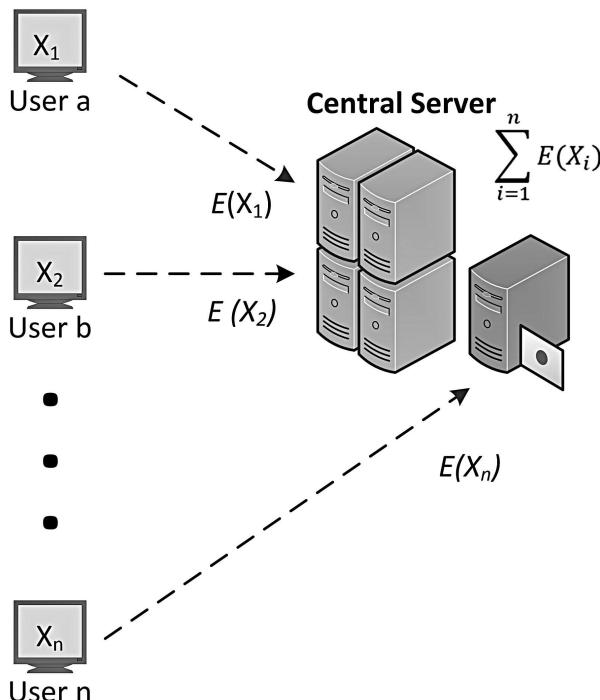
During the aggregation process, the clients encrypt their models and send them to the central server. The central server receives the encrypted models and, using the server’s key, decrypts them and develops a global model by aggregating the local models. This concept is illustrated in Figure 2.15. The

key management between the clients and the server is still in the development phase.

\*As we know, in cryptography, keys need to be kept confidential. The two cases mentioned below are regarding key management.

- **Case 1:** The secret key is available to the users but not to the central server.
- **Case 2:** The secret key is available to the central server but not to the user.

In Case 1, the central server is unaware of the secret key, resulting in uncertainty regarding the security of the global model. Case 2 addresses the issue from Case 1. In Case 2, the secret key is accessible to the central server but not to the user. Therefore, the server handles the decryption process, ensuring the security of the global model. Concerns were raised regarding the security of the local model. To resolve this, users participating in the process mask their models before sending them to the server, and then aggregate the mask using the Sec Agg protocol for unmasking.

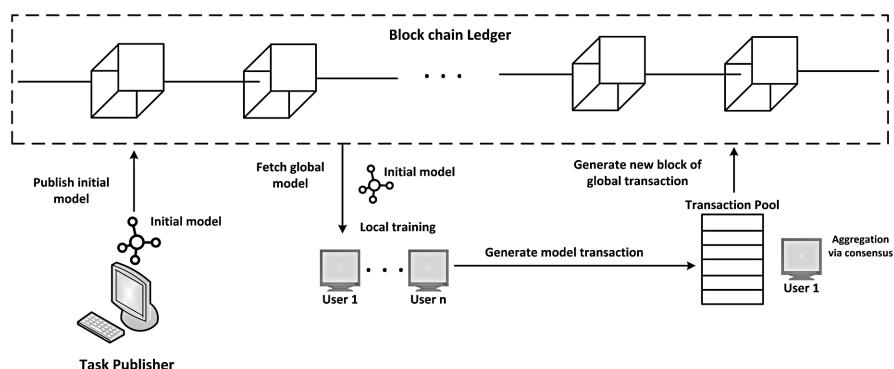


**FIGURE 2.15** Homomorphic encryption-based aggregation.

### 2.7.2 BLOCKCHAIN-BASED AGGREGATION

The blockchain is a technology that utilizes a chain of blocks in which users can make transactions without using a centralized concept. Each block contains information and a cryptographic hash to trace and link other blocks in the network. This makes a blockchain a tamper-proof, decentralized backend network. By utilizing this property, a blockchain-based federated learning architecture and aggregation technique are introduced. Based on Figure 2.16, the process of blockchain aggregation is given below.

1. An initial global model is published to the blockchain by the publisher.
2. The global model is then retrieved by the user from the blockchain ledger.
3. Each user in participation starts training with the global model.
4. The model developed by the user is stored in the transaction pool
5. By using a consensus protocol, the consensus node aggregates the local model and develops a global model.
6. For the next iteration, the resulting model, i.e., the global model, is then appended to the blockchain.

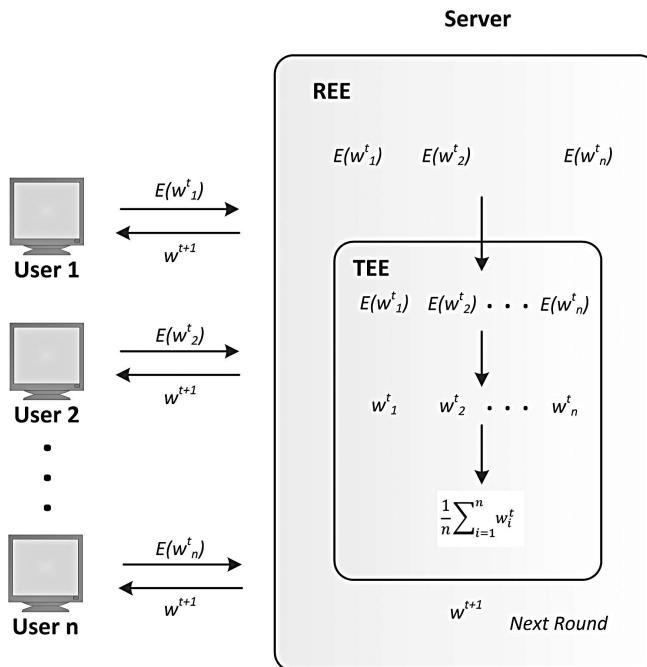


**FIGURE 2.16** Blockchain-based aggregation.

### 2.7.3 TEE-BASED AGGREGATION

A rich execution environment (REE) is an open part of the operating system where the untrusted application runs, whereas a Trusted Execution Environment (TEE) is an isolated, trusted, and secure global platform to store, process, and protect sensitive data. This environment protects data against attacks from the REE. The TEE cannot compromise the untrusted application

that runs on the REE, even if the operating system is compromised. The process is illustrated in Figure 2.17.



**FIGURE 2.17** TEE-based aggregation.

The processes of adapting TEE in federated learning are given below:

1. The user participating in the training encrypts their local model and sends it to REE.
2. The REE transfers the encrypted model to the TEE.
3. The TEE then decrypts the content from REE and develops an aggregate model.
4. The aggregate model is transferred from TEE to REE for distribution among all participants in the training.

## 2.8 CONCLUSION

The chapter discusses federated learning, its types, and model aggregation techniques in detail. For implementing different machine learning models over

distributed data, there are various approaches such as Centralized Machine Learning, DML, and Federated Machine Learning. However, both centralized machine learning and DML have limitations in terms of privacy preservation and generalization. In the federated learning approach, each client's machine learning model can be trained independently, and then the combined models can be used to generate a generalized global model. The chapter also explains various federated learning approaches, including "Horizontal Federated Learning," "Vertical Federated Learning," and "Federated Transfer Learning." In horizontal federated learning, the same sample set is used to make use of various features. On the other hand, in vertical federated learning, each client uses a separate set of samples with a similar set of features. The clients in federated transfer learning are free to employ any feature set and any sample set. The various aggregation techniques utilized in federated learning architectures are also explained in this chapter. The goal of aggregation is to gather all autonomous clients' model parameters and merge them into a single, global model. This duty is the most crucial one in federated learning, calling for better technological soundness. AMA, BMA, and OMS are three distinct strategies that were refined with individual algorithms and use cases. Along with the classical aggregation techniques, the article explains three secure aggregation protocols: Homomorphic Encryption-based aggregation, Blockchain-based aggregation, and TEE-based aggregation. Even though the chapter explains these approaches and types of federated learning, it is all with the assumption that all the clients use identical machine learning architecture specifications. Nowadays, federated learning architectures with heterogeneous client architectures and machine learning models are in practice. Therefore, the aggregation techniques need more sophisticated algorithms that can perform aggregation even with a heterogeneous collection of client models.

## KEYWORDS

- **algorithms**
- **blockchain**
- **centralized machine learning**
- **federated machine learning**
- **heterogeneous collection**
- **Homomorphic encryption**

## REFERENCES

1. Drainakis, G., Katsaros, K. V., Pantazopoulos, P., Sourlas, V., & Amditis, A., (2020). Federated vs. centralized machine learning under privacy-elastic users: A comparative analysis. In: *2020 IEEE 19<sup>th</sup> Int. Symp. Netw. Comput. Appl. NCA 2020*. doi:10.1109/NCA51143. 2020.9306745.
2. McMahan, B., Moore, E., Ramage, D., Hampson, S., & Arcas, B. A., (2017). Communication-efficient learning of deep networks from decentralized data. In: *Artificial Intelligence and Statistics* (pp. 1273–1282). PMLR.
3. Li, T., Sahu, A. K., Zaheer, M., Sanjabi, M., Talwalkar, A., & Smith, V., (2018). *Federated Optimization in Heterogeneous Networks*. [Online]. Available: <http://arxiv.org/abs/1812.06127>.
4. Qin, Y., & Kondo, M., (2021). MLMG: Multi-local and multi-global model aggregation for federated learning. In: *2021 IEEE Int. Conf. Pervasive Comput. Commun. Work. Other Affil. Events, PerCom Work. 2021* (pp. 565–571). doi:10.1109/PerComWorkshops51409. 2021.9431011.
5. Yang, Q., Liu, Y., Cheng, Y., Kang, Y., Chen, T., & Yu, H., (2020). *Federated Learning*. doi:10.1007/978-3-031-01585-4.
6. Yang, Q., Liu, Y., Chen, T., & Tong, Y., (2019). Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(2), 1–19.
7. Du, Z., Wu, C., Yoshinaga, T., Yau, K. L. A., Ji, Y., & Li, J., (2020). Federated learning for vehicular internet of things: Recent advances and open issues. *IEEE Comput. Graph. Appl.* doi:10.1109/OJCS.2020.2992630.
8. Rosenblatt, F., (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6), 386.
9. Marius-Constantin, P., Balas, V. E., Perescu-Popescu, L., & Mastorakis, N., (2009). Multi-layer perceptron and neural networks. *WSEAS Trans. Circuits Syst.*, 8(7), 579–588.
10. Hijazi, S., Kumar, R., & Rowen, C., (2015). Using convolutional neural networks for image recognition. *Cadence Des. Syst. Inc. SanJose, CA, USA*, 8925, 572–578.
11. Sherstinsky, A., (2020). Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network. *Phys. D Nonlinear Phenom.*, 404. doi:10.1016/j.physd.2019.132306.
12. Dinh, C. T., Tran, N. H., Nguyen, T. D., Bao, W., Zomaya, A. Y., & Zhou, B. B., (2020). Federated learning with proximal stochastic variance reduced gradient algorithms. *ACM Int. Conf. Proceeding Ser.* doi:10.1145/3404397.3404457.
13. Michieli, U., & Ozay, M., (2021). Are all users treated fairly in federated learning systems? *IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. Work.* (pp. 2318–2322). doi:10.1109/CVPRW53098.2021.00263.
14. Ye, D., Yu, R., Pan, M., & Han, Z., (2020). Federated learning in vehicular edge computing: A selective model aggregation approach. *IEEE Access*, 8, 23920–23935. doi:10.1109/ACCESS.2020.2968399.

## CHAPTER 3

---

# Federated Learning for IoT/Edge/Fog Computing Systems

BALQEES TALAL HASAN<sup>1</sup> and ALI KADHUM IDREES<sup>2</sup>

<sup>1</sup>*Department of Computer and Information Engineering,  
College of Electronics Engineering, Nineveh University, Mosul, Iraq*

<sup>2</sup>*Department of Information Security, College of Information Technology,  
University of Babylon, Babylon, Iraq*

---

### ABSTRACT

With the help of a new architecture called Edge/Fog (E/F) computing, cloud computing services can now be extended closer to data generator devices. E/F computing, in combination with Deep Learning (DL), is a promising technique that is widely applied in numerous fields. In conventional DL architectures with E/F computing, data producers can train their models by repeatedly transmitting and communicating data with third-party servers, such as Edge/Fog or cloud servers. However, this architecture is often impractical due to extensive bandwidth needs, legal issues, and privacy risks. To address these challenges, a centralized server can be used to co-train the models through Federated Learning (FL) with distributed clients, including cars, hospitals, and mobile phones, while preserving data localization. FL facilitates group learning and model optimization, making it a motivating element in the E/F computing paradigm. Although previous studies have considered FL applications in E/F computing environments, the execution and hurdles of FL in the E/F computing framework have not been thoroughly covered. In order to identify advanced solutions, this chapter will provide a review of the application of FL in E/F computing systems. By conducting this study, researchers can gain a better understanding of how E/F computing

---

Federated Learning: Principles, Paradigms, and Applications.

Jayakrushna Sahoo, Mariya Ouassa, & Akarsh K. Nair (Eds.)

© 2025 Apple Academic Press, Inc. Co-published with CRC Press (Taylor & Francis)

functions and how FL enables related concepts and technologies. Several case studies on the implementation of federated learning in E/F computing are currently being investigated. The open issues and future research directions will also be introduced.

### **3.1 INTRODUCTION**

The number of Internet of Things (IoT) devices has significantly increased, resulting in a remarkable increase in generated data. By 2025, it is projected that the number of nodes connected to the Internet will reach 80 billion, and a staggering 180 trillion gigabytes (GB) of data will be produced globally [37]. IoT systems typically rely on cloud-based IoT architecture, where cloud servers are utilized to manage physical objects. However, cloud-based architecture faces various challenges, including security, latency, and bandwidth. To address these challenges, new technologies such as fog and edge computing have emerged. These technologies aim to extend the cloud computing model by bringing storage, computing, and networking resources closer to data sources or end-user applications [7].

Smart objects must support some form of artificial intelligence (AI). The success of deep learning (DL) in video and speech applications, two of the most essential IoT services, has made it an extremely important area of research to adapt its models for deployment on edge computing devices with limited resources [29]. In order to support DL, new hardware has been introduced; typically, an external chip or graphical processing unit (GPU) is introduced for this purpose and is referred to as an AI chip. However, AI chips are costly, making it uneconomical or unaffordable for them to be utilized in a variety of products. Furthermore, as dataset sizes grow, machine learning (ML) models get more complex, with a DNN using millions of parameters [26]. Besides this, to train traditional DL models in edge computing devices, owners of data must regularly share raw data with external servers such as edge or cloud servers. Because of the legalization, high bandwidth requirements, and privacy vulnerabilities, this architecture is frequently impractical [1]. As a result, decentralized approaches, like Federated Learning (FL), became an unavoidable substitute for centralized ones [26].

In 2016, Google offered Federated Learning (FL) as a distributed machine learning (DML) approach to offload the processing of AI applications to a growing number of end devices while ensuring the privacy of end users [50]. Since then, FL has rapidly evolved and become a popular topic of AI research [51]. FL enables fully distributed training by dividing the process

into two stages: client-side parallel model updating based on local datasets, and server-side global model aggregation [24]. FL is considered a successful strategy for leveraging distributed resources to collaboratively train a machine learning model. It maintains the decentralized nature of raw data, keeping it away from a single server or data center, while allowing multiple clients to work together in training a model. In FL, only intermediate data is exchanged between distributed computing resources, while transmission of training data is prohibited [23]. FL incorporates techniques from various research areas, including distributed systems, machine learning, and privacy [21]. The FL approach finds applications in diverse business sectors, such as pharmaceuticals, medical AI, telecommunications, IoT, transportation, traffic prediction, and monitoring [37].

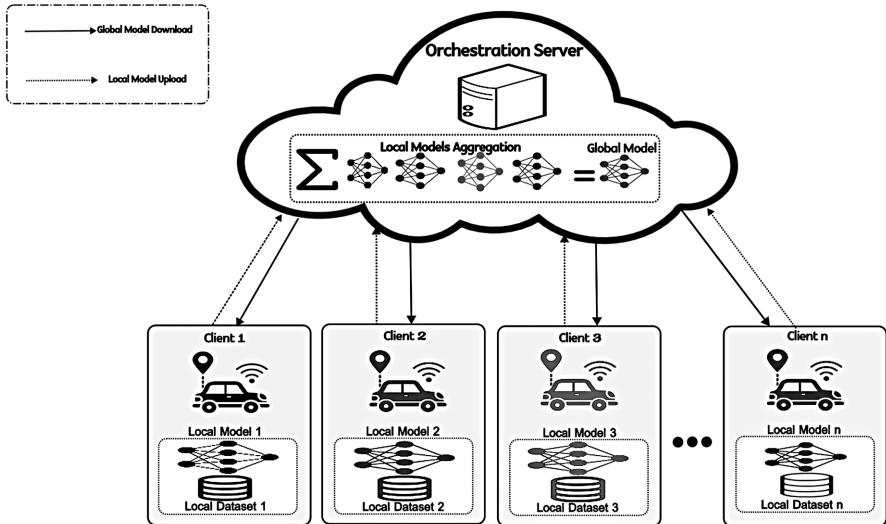
### 3.2 A FEDERATION CONCEPT

There are real-world applications for the federation concept, including business and sports. The fundamental concept of federation is the collaboration of multiple independent parties. Federation is not only common in society, but it is also crucial in computing. Federated computing systems have long been a popular field of research in computer science. In the 1990s, federated database systems (FDBSs) were the subject of numerous studies. An FDBS is a group of independent databases that work together for common advantages. As cloud computing has developed, numerous studies on federated cloud computing (FC) have been carried out. Multiple internal and external cloud computing services are deployed and managed under the umbrella of FC. By outsourcing some tasks to more economically advantageous locations, the cloud federation concept enables further cost savings [22]. In 2016, McMahan et al. coined the term “Federated Learning” which refers to a distributed ML technique in which a model is trained collaboratively by a group of participating clients (such as mobile devices or entire organizations) under the orchestration of a centralized server. Rather than sending or receiving raw data between clients and servers, the central server instead receives targeted updates meant for instant aggregation [18].

### 3.3 FEDERATED LEARNING WORKFLOWS

In an FL system, the common architecture and training process workflow can be illustrated in Figure 3.1. Two roles often predominate in an FL system:

(i) clients that maintain their local data; and (ii) a server that manages the process of training the ML model and updating the global model without access to client data. Although there is typically only one server, there may be many clients. In a specific FL system, a particular client can take on the role of the orchestrating server during the training phase [51].



**FIGURE 3.1** The federated learning architecture.

Assume there are  $n$  clients, each of which is represented by  $C_i$ , where  $i \in [1, n]$  and  $C_i$  maintains a dataset  $D_i$ . In a traditional ML approach, all the clients' datasets will be combined in  $D$ , making  $D = D_1 \cup \dots \cup D_n$ , and then the ML model will be trained using  $D$  [49]. Whereas in an FL approach, a global model is jointly trained by all  $n$  clients utilizing their local datasets [51].

In the FL training process, there are normally three main steps [51, 53]:

1. **Initialization:** First, the orchestration server determines the architecture of the global model, initializes the parameters of the global model randomly or by pre-training on a public dataset, and initializes the hyper-parameters (e.g., number of FL rounds, total number of clients, and number of clients chosen for each training round). Then, the server broadcasts the parameters of the initial global model  $w_0^g$  to the selected clients.
2. **Training Local Models:** In the  $t$  round, the selected clients receive the most recent global model information from the server (such as

gradients or weights), and they utilize their local datasets to update the parameters of their individual local models  $w_t^i$ . Each participating client will transmit the updated local model parameters  $w_{t+1}^i$  to the server once the local training is completed. In the  $t$  round, client  $i$  seeks to obtain the ideal local model parameters through the reduction of the loss function  $F(w_t^i)$ , which is expressed by the following equations:

$$w_t^i = \operatorname{argmin}_{w_t^i} F(w_t^i) \quad (1)$$

$$F(w_t^i) = \frac{1}{|D_i|} \sum_{j \in D_i} f_j(w_t^i) \quad (2)$$

where;  $|D_i|$  denotes how many samples are in the dataset  $D_i$ . Each client's update process can be accomplished by using stochastic gradient descent (SGD) with mini-batches sampled from its local dataset:

$$w_t^i = w_t^i - \eta \nabla F(w_t^i) \quad (3)$$

where;  $\eta$  denotes the learning rate, and  $\nabla F(w_t^i)$  is the loss function's gradient.

- 3. **Global Model Aggregation:** In each round, the server collects the updated local parameters of the selected clients. It replaces the global model parameters with the updated model parameters and then transmits the updated global model parameters  $w_{t+1}^g$  back to the clients for the following training round. In particular, the goal is to obtain the ideal global model parameters  $w_t^g$  through the reduction of the global loss function, which can be described as follows:

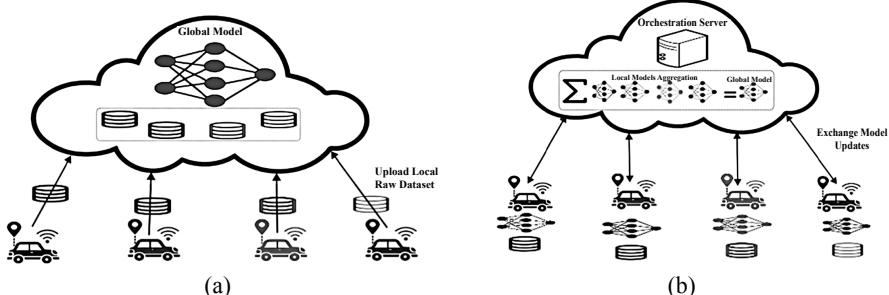
$$F(w_t^g) = \frac{1}{|D|} \sum_{i=1}^n |D_i| F(w_t^i), i \in 1, 2, \dots, n \quad (4)$$

Upon fulfillment of the termination condition (e.g., when the number of rounds is increased to its maximum or the global model's accuracy surpasses the threshold), the server terminates the training process, aggregates the updates, and broadcasts the global model to all clients [51].

### 3.4 FEDERATED LEARNING VS. NON-FEDERATED LEARNING

Traditional centralized ML approaches build ML models by aggregating distributed raw data generated from various organizations or devices into a

single cluster or server [23, 40]. Figure 3.2(a) depicts a traditional centralized ML system, which consists of a server and several edge devices. In order to create the centralized training data that the server will use to train the ML model, the edge devices upload their local dataset to the central server. However, centralized ML approaches encounter critical issues like privacy violations, high communication overheads, and scalability. To resolve these issues, the idea of FL has been put forth as a practical alternative to traditional centralized ML. As depicted in Figure 3.2(b), typically, an FL system consists of a server as well as several edge devices. In FL, training data is kept locally while multiple edge devices train ML models cooperatively and distribute them under the supervision of a centralized server. In particular, the edge devices implement local training on their dataset, and they send the results of that training to the server, which updates the global ML model after aggregating the received results. The global ML model can attain adequate accuracy after numerous interactions between the edge devices and the server, indicating that training is finished [40].



**FIGURE 3.2** A general framework for a centralized ML system and an FL system: (a) Centralized ML system; and (b) FL system.

FL differs from the traditional centralized ML approach in several important aspects. In FL, edge devices send model update parameters to the centralized server instead of raw data, which minimizes the communication data size and improves network bandwidth usage. Another crucial aspect is latency, which is vital in time-critical applications such as real-time media, industrial control, remote control, and mobility automation. Additionally, real-time decision-making applications like augmented reality and event detection can be executed locally at edge devices to enhance performance. As a result, FL systems exhibit much-reduced latency compared to centralized ML systems. The third aspect is privacy: because raw data is not transmitted

to a central server, each user's privacy is ensured. Furthermore, FL is simple and uses less power because the models are trained collaboratively on multiple edge devices, indicating that edge computing is an appropriate environment for FL [1].

In general, centralized ML approaches pose a number of challenges, including training time, computational power, and most significantly, security and privacy of data. Three aspects set FL apart from the centralized ML approach. First, direct raw data communication is allowed under the centralized ML approach, but FL forbids it. Second, the centralized ML approach typically uses one server or cluster in one region owned by a single business, whereas FL uses distributed computing resources across several regions or businesses. Thirdly, FL frequently employs encryption or other security measures to ensure security or data privacy, in contrast to the centralized ML approach, which is less concerned with these security issues [23].

### 3.5 SCALE OF FEDERATION

Depending on the scale of the federation, FL systems are divided into two categories: cross-silo and cross-device FL systems. This section will discuss the categories of the federation's scale.

#### 3.5.1 CROSS-SILO

Clients in this system are typically small in scale, indexed, and almost always accessible for training rounds, ranging from 2 to 100 devices. A cross-silo FL system is more flexible than a cross-device FL system. Clients are typically organizations [30]. For example, using shopping information gathered from hundreds of data centers worldwide, Amazon aims to recommend products to users. Each data center has ample computational power and access to a vast amount of data [22].

#### 3.5.2 CROSS-DEVICE

There are many clients in the cross-device Federated Learning (FL) system, but each client only has limited data and processing power. Typically, these clients are mobile devices. An example of a cross-device FL system is

Google Keyboard. FL can be utilized to improve the query suggestions of Google Keyboard [22].

### 3.6 FEDERATED LEARNING NETWORK TOPOLOGY

This category focuses on FL's core architecture, or how FL's environment is built from its main parts. FL can be either centralized or decentralized depending on the network topology as listed below [30]:

#### 3.6.1 *CENTRALIZED-FL*

Hub-and-spoke topology, which includes a single server and numerous clients, is considered the fundamental and most common FL topology that provides a centralized authority in charge of overseeing and managing the training process. Despite the fact that FL is built on the approach of decentralized data nature, there still exists a demand for a central server in the FL environment to aggregate trained models from participating clients, create a global model, and share it with all FL environment clients [30]. Cross-device systems typically employ this topology [21]. Centralized FL uses an iterative method called a “federated learning round” that involves multiple client-server exchanges [37]. The following are the primary steps in each round of iterative learning [18, 27];

- **Client Selection:** The server chooses participating clients that meet the eligibility criteria.
- **Parameter Broadcasting:** The selected participating clients receive a broadcast of the global model parameters from the server.
- **Local Model Training:** The participating clients will simultaneously retrain their local models using their local datasets.
- **Model Aggregation:** The server aggregates the updates from participating clients. If a sufficient number of devices have provided results, stragglers can be eliminated for improved effectiveness.
- **Model Update:** Based on the aggregated update computed from the clients who took part in the current round, the server updates the shared model locally.

The central server requests the termination of the iterative training process when a specified criterion for termination is met. The global trained model is then considered a robust model by the FL server [37].

### 3.6.2 DECENTRALIZED-FL

The peer-to-peer topology is used in the decentralized FL approach. In this approach, a group of clients can build an ML model and achieve high accuracy without the aid of a third-party centralized server for model aggregation [30]. This topology is commonly used in cross-silo systems with powerful client computing capabilities [21]. Peer-to-peer topology eliminates the central server that could serve as a breeding ground for threats and vulnerabilities. However, the overall FL system may necessitate the involvement of an independent entity in control and arbitration roles. This is not always ideal due to increased expenditures and procedural viscosity [6]. The training process should be orchestrated using a protocol since there is no central server. Generally, there are two protocols to orchestrate the training process [51].

- **Cyclic Transfer:** The clients in this protocol are arranged in a circular chain. Client C<sub>1</sub> transmits its most recent model information to client C<sub>2</sub>. Client C<sub>2</sub> acquires model information from client C<sub>1</sub> and modifies it using its own data before sending the modified model information to client C<sub>3</sub>. The training process is terminated when the termination condition is met.
- **Random Transfer:** In this protocol, a client C<sub>k</sub> selects a client C<sub>i</sub> at random with equal probability and sends its model information to C<sub>i</sub>. After receiving the model data from C<sub>k</sub>, C<sub>i</sub> updates it using its local dataset before randomly selecting client C<sub>j</sub> and providing the new model information to C<sub>j</sub>.

When using networks with high latency or limited bandwidth, it has been demonstrated that decentralized learning is faster than centralized learning. Similarly, decentralized algorithms in FL have the potential to reduce the central server's high communication costs [14].

## 3.7 CLASSIFICATION OF A FEDERATION

FL is often broken down into three categories: vertical, horizontal, and transfer learning depending on the distribution of the data over the sample and feature spaces. These categories are discussed below:

### **3.7.1 VERTICAL FEDERATED LEARNING**

Vertical Federated Learning (also referred to as feature-based FL) is used when each participating client has a dataset with different features but shares the same sample space [49]. Organizations that have data on the same group of people but with varying feature sets can employ vertical FL to build a shared ML model [27]. Vertical FL is suitable for cooperation between general and specialized hospitals. Despite the fact that they may share the same patient's data, the general hospital is the owner of the patient's generic information, and the specialized hospital is the owner of the patient's specific test results. They can, therefore, utilize Vertical FL to cooperatively train a model that anticipates a specific disease tested by the specialist hospital based on the features of the general hospital [43].

### **3.7.2 HORIZONTAL FEDERATED LEARNING**

Horizontal Federated Learning (also referred to as sample-based FL) is used when each participating client has the same feature space but different sample instances in their dataset [49]. Horizontal FL is suitable for credit risk management in banks. Suppose a group of banks all have the same set of features but have a relatively small number of customers. If those banks aggregate their samples, they can create a larger dataset from which to train a more accurate model. However, banks would be wary of disclosing their private information to outside parties for business reasons. Through the use of horizontal federated learning, local models from many banks may be combined to train a global model.

### **3.7.3 FEDERATED TRANSFER LEARNING**

Federated Transfer Learning is used when each participating client has a different feature space and sample of data. Traditional ML approaches are typically restricted to resolving issues in one specific domain, i.e., training and testing data must follow the same distribution. Retraining the model on the new dataset is frequently required when the training and testing data's feature distributions are different. However, in real-world scenarios, reacquiring data can be very expensive or even impossible. It's essential to transfer practical knowledge acquired from an existing domain to a new domain without retraining models. Using knowledge from other domains,

federated transfer learning can be used to create an efficient model for the target domain [54].

### 3.8 FEDERATED LEARNING AGGREGATION ALGORITHMS

Several federated learning (FL) algorithms have been proposed to improve model update aggregation. The aggregation algorithms can be categorized as centralized, hierarchical, or decentralized. In the following sections, we will cover the most popular aggregation algorithms and provide a summary of their main advantages and disadvantages, as shown in Table 3.1.

**TABLE 3.1** Summary of the FL Aggregation Algorithms

Algorithm	Type	Merits	Demerits
FedSGD	Centralized	FedSGD can be performed on resource-constrained devices because it only requires one epoch of SGD implementation in each round.	Since aggregation happens after each local gradient step, this leads to more communication rounds and slowly converges global model training.
FedAvg	Centralized	FedAvg, in comparison to FedSGD, reduces the high communication cost and accelerates global model convergence by increasing the number of local training epochs performed on the edge device.	Weights may perform drastically worse when averaged along coordinates, which reduces communication effectiveness.
HierFAVG	Hierarchical	In comparison to cloud-based FL, HierFAVG can decrease both the end devices' energy consumption and the model training time.	Due to the close coupling between the two communication types, any failures of an edge server or client device will make the cloud wait incredibly long.
LanFL	Hierarchical	LanFL can greatly speed up training and lower traffic on WANs.	LAN introduces additional bias into the device selection process.
HADFL	Decentralized	HADFL effectively makes use of heterogeneous computing power while relieving the central server's communication burden.	HADFL's accuracy loss can occasionally be slightly greater than that of other algorithms.
IPLS	Decentralized	IPLS is resource-efficient and resistant to sporadic connectivity and dynamic participant arrivals/departures.	There is no incentive mechanism in IPLS.

### **3.8.1 CENTRALIZED AGGREGATION**

Central servers are used in centralized aggregation algorithms to compute the average models or gradients sent from multiple clients [23].

Federated stochastic gradient descent (FedSGD) is a standard FL aggregation algorithm built on SGD. FedSGD considers a single SGD step because the chosen participating clients perform one epoch of gradient descent per round. In each round, the participating clients compute the gradients for the local ML models using their local data and upload the results to the server. The global model is then updated by averaging the gradients from all participating clients [28]. The federated averaging algorithm (FedAvg), suggested by McMahan et al. as a more generalized version of FedSGD, is currently the most commonly used aggregation algorithm [28]. FedAvg allows each client to do an SGD optimization on its local dataset in more than one batch or to run multiple epochs of local training. In this way, less client-server communication is needed, and instead of exchanging gradients with the server, they do so by exchanging delta weights [21, 26]. In this context, a round refers to the uploading procedure of information from the client to the server, updating the global model, and downloading global model information from the server. It differs from an epoch, which is concerned with running the centralized ML model and altering the weights and biases of the local model [26].

### **3.8.2 HIERARCHICAL AGGREGATION**

A hierarchical aggregation depends on the use of multiple aggregation servers [23]. HierFAVG aims to facilitate partial local model aggregation at the edge server level. This algorithm involves two types of communication: client-edge and edge-cloud. In client-edge communication, each edge server aggregates local model updates from its own participating clients. On the other hand, in edge-cloud communication, the aggregated updates from the edge servers are sent to the main cloud server [24].

Due to the LAN's abundant bandwidth and low financial cost compared to the WAN, the LanFL algorithm proposes the use of a hierarchical aggregation mechanism in LAN. LanFL enables peer-to-peer device organization. A locally shared model is trained by frequently exchanging model updates between devices connected to the same LAN domain. To jointly train a globally shared model, many LAN domains also upload their model updates to a centralized server, although this occurs infrequently [52].

### 3.8.3 DECENTRALIZED AGGREGATION

Traditional centralized federated learning (FL) has a number of issues, including a single point of failure and the fact that the training process would be abruptly and fully stopped if the central server went down. To handle the transfer of potentially massive volumes of data with all of the clients, the central server also needs to have reliable and high-bandwidth communication links with them. Lastly, each client needs to have trust in the server [34]. Decentralized FL has been introduced to address the aforementioned issues. In decentralized FL, each device performs the calculation equally under the control of the decentralized aggregation algorithms [23].

Device homogeneity is a presumption taken by FL. When FL is used with heterogeneous devices, fast devices waste computing power waiting for slow devices, which is inefficient. The HADFL algorithm has been proposed as a solution to this issue. HADFL enables decentralized peer-to-peer training on heterogeneous devices, allowing these devices to run different local steps in accordance with their computing capabilities [9]. The IPLS FL algorithm, which is partly based on the interplanetary file system, has been described as a fully decentralized FL algorithm. Each device in IPLS is in charge of some model partitions, and devices communicate with one another by exchanging model updates or partitioned gradients [34].

## 3.9 FUNDAMENTALS OF CLOUD, EDGE, AND FOG COMPUTING

The main backbone for any machine or device, including IoT, is computing. As seen in Figure 3.4, computing may take the form of edge computing, fog computing, or cloud computing (EC)[39].

Over the last decade, computing paradigms have evolved significantly. Undoubtedly, cloud computing is the most well-known one. This paradigm was developed in response to the need to use “computing as a utility” to simplify the development of new internet services. Before the IoT revolution, cloud computing was a common research topic; however, the IoT revolution exposed all the flaws in such a centralized paradigm, sparking interest in decentralized paradigms [13]. As a result, fog and edge computing were developed as complements to cloud computing to bridge the gap between the cloud and IoT devices that are geographically dispersed [7].

In 2006, the term “Cloud computing” was first used by Amazon and Google [19]. The NIST defines cloud computing as a model for providing practical, global, on-demand network access to a shared pool of reconfigurable

computing resources (such as services, storage, servers, networks, and applications) that may be quickly provided and released with little administration work or service provider involvement [13]. Despite the various advantages of cloud computing, such as scalability, effectiveness, cost savings, and reliability, there are also some disadvantages when dealing with large amounts of data. Challenges in cloud computing include real-time analytics, load balancing, high internet bandwidth, latency, energy consumption, data management, security, and privacy. Additionally, since it utilizes a centralized computing model, most operations are performed directly in the cloud. The limitations of cloud computing can be overcome with edge and fog computing [19].

The emergence of IoT marked the beginning of the post-cloud computing era. The close relationship between cloud and IoT, exemplified by the direct processing of IoT data on the cloud, raises several issues that cloud computing alone cannot fully address [13]. Edge computing is utilized to meet the industry's demands for connectivity agility, application intelligence, data optimization, real-time business, privacy, and security [10]. Edge Computing, which enables end devices to process and store data locally, has thus emerged to enhance cloud performance and resolve cloud-related challenges [19]. The edge computing model avoids uploading data to a cloud computing platform and instead stores and processes it on edge devices. This characteristic makes edge computing superior in the following areas:

- **Fast Data Processing:** Due to the proximity of edge computing nodes to the data source, they can perform computing and data storage tasks locally. This reduces the need for intermediate data transmission, minimizing delay time and ensuring real-time processing.
- **Security:** All data should be uploaded to the cloud to utilize the unified processing feature of traditional cloud computing, which adopts a centralized processing approach. However, this process carries certain risks, such as data leakage and data loss. On the other hand, edge computing ensures data security as there is no need to upload data to the cloud.
- **Low Network Bandwidth:** Edge computing does not require significant bandwidth on the network because processing the data does not involve uploading it to cloud computing [10].

Analytically, even when cloud and edge computing are utilized in tandem, some limitations remain because edge nodes are lightweight with limited processing and storage capabilities, resulting in resource contention and increased processing latency. Fog computing has been suggested as a middle resource that can easily combine edge and cloud resources [8]. At CISCO

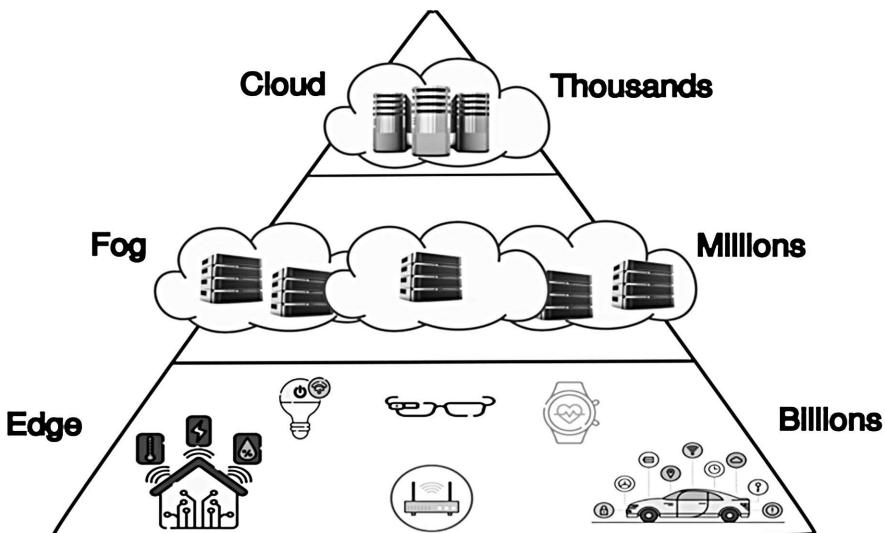
in 2012, Bonomi coined the term “Fog Computing” for the first time. Fog computing, which puts processing and intelligence near the data sources, expands on the cloud computing architecture [17]. Fog computing offers a new layer of computing that sits between conventional cloud computing and distributed IoT devices [42]. As a result, the fog computing layer’s data reduction can result in a faster response time to smart-end devices while also lowering the size of the data uploaded to the cloud platform, resulting in network bandwidth savings [17]. Fog computing and edge computing share many of the same principles such as mobility support, low bandwidth costs, low latency, high scalability, and virtualization service. It does, however, have constrained resources, fewer computational and storage options, and is closer to end devices than Fog computing [19]. In particular, Fog and Edge Computing offers five key benefits, which include:

- **Security:** Fog and Edge computing support provide extra security for IoT devices to ensure transaction security and reliability.
- **Cognitive:** Fog and Edge computing enable clients to have awareness of their objectives, facilitating autonomous decision-making regarding the deployment of storage, computing, and control functions.
- **Agility:** Fog and Edge computing improve the agility of the wide-spread deployment of an IoT system.
- **Latency:** Fog and Edge computing offer fast responses for applications that demand extremely low latency.
- **Efficiency:** Fog and Edge computing enhance the efficiency of cloud computing by improving performance and reducing unnecessary costs [7]. Figure 3.3 illustrates the hierarchical relationship between the cloud server, fog gateway, and edge devices.

### 3.10 ENABLING DEEP LEARNING AT THE EDGE/FOG

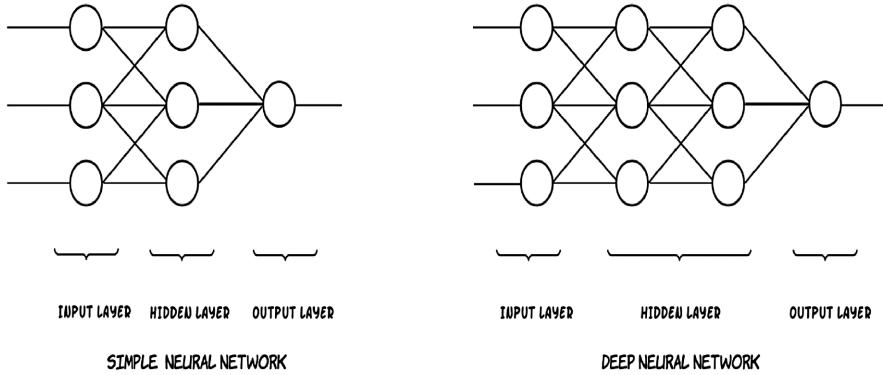
Machine learning (ML), which has been around since the 1950s, has revolutionized numerous fields in recent years. DL emerged primarily after 2006 from a branch of ML known as neural networks (NN)[3]. DL is also referred to as representation learning (RL)[4]. Since its inception, DL has demonstrated exceptional success in almost every application domain [3]. DL draws inspiration from how the human brain processes information. DL does not require any rules created by humans to function; instead, it extensively utilizes data to match the input to specific labels [4]. Using traditional ML techniques, the classification task must be completed in a series of sequential

actions, such as preprocessing, feature extraction, feature selection, learning, and classification. Additionally, the performance of ML techniques is significantly influenced by feature selection. Biased feature selection may lead to incorrect class distinction. In contrast, unlike conventional ML techniques, DL can automatically learn feature sets for a variety of tasks [4].



**FIGURE 3.3** Cloud, fog, and edge computing hierarchy.

DL is defined as the use of multiple-neuron, multiple-layer neural networks to carry out learning processes such as clustering, classification, and others [16]. A basic neural network consists of three layers: input, hidden, and output. In a simple neural network, no computation occurs in any of the input nodes; instead, the information is passed on to the hidden nodes. As a result, this network only has two layers. Traditional computing systems based on central processing units are insufficient to support multilayer architectures because there are thousands of interconnections between these layers in deep architectures. By utilizing GPU computing and having access to a large amount of training data, it becomes possible to add additional layers between the input and output layers. A simple neural network is transformed into a deep neural network by including two or more hidden layers [20]. Figure 3.4 illustrates a simple neural network and a DNN with two hidden layers.



**FIGURE 3.4** Neural network vs. deep neural network.

Applications of DL can be found in many aspects of our lives, including image recognition, image description or caption generation, speech recognition, demographic prediction, earthquake prediction, and election result prediction [20].

Depending on the training approach, DL can be classified into one of two categories: Centralized and Federated Training. In a traditional centralized training approach, the server requests that clients send their training samples to the cloud. After initializing the DNN in the cloud, the central server uses training samples to train the DNN until the ideal parameters are found. Finally, the cloud server gives the users the ideal parameters. In FL, as opposed to centralized training, to train a single DL model, each user and server cooperate with each other. The cloud server receives local parameters (i.e., gradients) from each user, aggregates them, and then sends the results to each user, accelerating the model's convergence. The best network parameters will ultimately be used by the server and each user [48].

Recently, IoT has become more prevalent in several fields, including health, transportation, agriculture, education, and smart cities. The majority of applications in these fields rely on intelligent learning strategies for data mining, pattern recognition, prediction, or data analytics. In recent years, DL (Deep Learning) has been commonly used in various IoT applications [29]. Currently, DL models are primarily trained on powerful computing platforms, such as cloud data centers, using large centralized datasets. However, as many applications generate and distribute data at network edge devices like sensors and smartphones, transferring this data to a centralized server for model training presents several challenges [24], such as:

- **Latency:** Sending data to the cloud for training cannot satisfy the stringent low-latency criteria required for real-time applications.
- **Scalability:** These issues arise when data is moved from sources to the cloud because, as more devices are connected, network access to the cloud may become more congested.
- **Privacy:** Transferring data to the cloud exposes users who own the data to potential privacy issues [11]. To address these issues, distributed training has begun to draw a lot of attention. Google has suggested using FL to train a DNN without storing the data in a central server [24].

DL has gradually shifted towards network edges and fog computing as new computing paradigms to improve decision-making quality while addressing the aforementioned latency, scalability, and privacy issues. Because IoT data is generated close to the end user, the newly emerging concept of DL at network edges and fog computing is thought to be a better solution. It has built-in advantages in a wide range of industries, including healthcare, self-driving cars, smart cities, and numerous other applications [42].

Pushing DL processing from the cloud to the edge has a number of benefits, such as:

- The cost and latency of data transmission for processing are greatly reduced due to the deployment of DL services near the users that are submitting requests, and the cloud only takes part when more computation is needed.
- User privacy is better protected because DL services require local storage of the raw data on the edge rather than cloud storage.
- A more trustworthy DL computation is offered by the hierarchical computing architecture.
- Edge computing, which makes use of a more diverse range of data and application scenarios, can enable the widespread use of DL and achieve the potential of “offering AI for every individual and every company everywhere.”
- Diverse and beneficial DL services can increase the commercial value of edge computing and accelerate its adoption and expansion [44].

### **3.11 FEDERATED LEARNING IMPLEMENTATION IN EDGE/FOG COMPUTING**

Several tools and frameworks are now available due to the increasing demand for FL technology. This section will introduce the most popular FL frameworks:

- **TensorFlow Federated (TFF):** The open-source FL framework TFF is built on top of TensorFlow. Google developed TFF in 2019 to enable developers and researchers to simulate federated learning scenarios using TFF's built-in FL algorithms, as well as to allow developers to create their own new FL algorithms. However, the TFF framework does not support the implementation of true federated learning on real devices [41].
- **Federated AI Technology Enabler (FATE):** It is an open-source FL computing framework launched in January 2019. To implement secure computation protocols, it makes use of Homomorphic encryption and multi-party computation. Developers can deploy FATE on a single node for simple testing or on multiple nodes for scalability, reliability, and manageability [45].
- **Paddle Federated Learning (PFL):** Baidu scientists developed PFL in 2020 based on Parallel Distributed DL (Paddle). PFL supports both vertical and horizontal FL. Developers can use PFL to deploy FL systems on massively distributed clusters [33].
- **PySyft:** A Python library developed by Open Mined uses the concepts of Encrypted Computation, Differential Privacy, and Federated Learning to decouple the training of the global model from the private data. It can be used within TensorFlow and PyTorch, two popular deep learning frameworks [36].
- **Flower (flwr):** It was developed by Daniel et al. as a unified solution for federated learning, analytics, and evaluation. flwr was built to support federated learning at large-scale real-world systems. flwr is compatible with a variety of operating systems and hardware platforms, allowing it to function effectively in heterogeneous edge device environments [5].
- **NVIDIA Federated Learning Application Runtime Environment (NVIDIA FLARE):** It is an open-source software development kit that enables federated learning among various parties. It includes privacy-protecting algorithms that ensure that the changes to the global model are all hidden and guard against the server's ability to reverse-engineer the submitted weights [32].

### 3.12 FL DATASETS

There were several datasets available for the implementation of FL. While some were open to the public, others were not (37). For instance, LEAF was one of the first federated learning dataset proposals. It includes six

datasets from various domains, such as image classification, next-character prediction, and sentiment analysis (22). The Shakespeare dataset is based on each speaking role in William Shakespeare's complete works. Federated Extended MNIST (FEMNIST) addresses natural heterogeneity caused by writing style. The StackOverflow dataset is comprised of StackOverflow data maintained by the Google team (23).

### **3.13 FEDERATED LEARNING IN EDGE/FOG COMPUTING: CASE STUDIES**

One of the newest and most exciting technologies is Federated Learning (FL). When protecting privacy and making the best use of available resources are priorities, FL on Edge/Fog (E/F) computing is widely applicable. This section will introduce a few use cases for implementing FL in E/F computing, as well as some recent research that has been applied to them.

#### **3.13.1 HEALTHCARE SYSTEMS**

Although the edge node stores and processes each medical institution's data separately, the model trained on a small dataset from a single medical institution performs poorly when applied to unseen data. As a result, a significant number of actual electronic health records (EHR) are required to train a robust medical model. However, due to the privacy and sensitivity of medical data, meeting the demand for real datasets is challenging. To address this issue and comply with the Health Insurance Portability and Accountability Act without disclosing patient data [46], institutions in the medical field can collaborate on training models using federated learning (FL) on edge computing.

Hakak et al. [15] proposed a personalized edge-based federated learning approach to assist healthcare practitioners by providing insights for disease diagnosis and prognosis through distributed wearable device data analysis. The incorporation of federated learning into the system architecture is crucial to ensure the privacy of sensitive medical data generated by wearable devices, as well as to ensure fast response time. Qayyam et al. [35] proposed a clustered federated learning model that can diagnose COVID-19 based on the classification of chest images from different sources, including X-rays and ultrasound, without disclosing any information about the training data samples. This model enables remote healthcare centers without advanced diagnostic equipment to benefit from the collaborative learning paradigm.

A collaborative machine learning model can be built in a cloud-edge infrastructure using visual data, such as X-rays and ultrasounds, collected from various healthcare facilities. The edges of the cluster represent healthcare organizations, such as remote medical imaging facilities, while the cloud server represents large hospitals or other governmental bodies, such as the Ministry of Health, that facilitate weight aggregation.

### **3.13.2 VEHICULAR NETWORKS**

Vehicular edge computing (VEC), a rapidly developing field of vehicular technology, utilizes moving vehicles and roadside servers at the network's edge to provide storage, computation, communication, and data resources to nearby vehicular users. To meet the increasing technical requirements of AI applications in vehicular networks, it is crucial to incorporate Federated Learning (FL) as a key technical approach in VEC [50].

Ye et al. [50] proposed a method for Federated Learning (FL) in VEC for image classification using selective model aggregation. They assessed image quality in terms of motion blur using a geometric model. A resource consumption parameter is then used to quantify the computational ability. After assessing the quality of local images and computational ability, the "fine" participating clients' DNN models are chosen and transferred to the main server for aggregation. Lu et al. [25] proposed a VEC asynchronous FL scheme. Local differential privacy (LDP) was incorporated into the gradient descent local training process to keep each participant client's local models secure. They proposed replacing the conventional update system between clients and a centralized server with a random peer-to-peer update mechanism because the centralized curator raised security and privacy concerns.

### **3.13.3 UNMANNED AERIAL VEHICLES (UAVS)**

UAVs are gaining popularity because they can perform a variety of difficult tasks in three-dimensional space [31]. The ability to control a large number of UAVs in real-time enables mission-critical applications such as search-and-rescue missions, delivering first-aid kits, covering large disaster sites, and firefighting scenarios. UAVs are very difficult to control in windy conditions, which increases the risk of collisions and slows down travel times. Mean-field game (MFG) has been used for massive UAV control to restrict inter-communications between UAVs by resolving two stochastic differential

equations that are coupled: the Hamilton-Jacobi-Bellman (HJB) equation for optimal control decision and the Fokker-Plank Kolmogorov (FPK) equation for the population distribution estimation. To estimate the HJB and FPK equation solutions, each UAV uses two DNNs. Federated learning is used to ensure convergence and expedite DNN training by periodically exchanging the DNNs' parameter sets for solving the HJB and FPK equations. MFG control based on federated learning can cut travel distance and collision risk by half [38].

A jamming attack, in essence, obstructs device communication by disrupting the reception of communications at the receiver using minimal transmission power. To enhance a defense strategy based on reinforcement learning, Mowla et al. proposed a defense strategy-based Federated Learning (FL) approach that enables the detection of on-device jamming attacks. The FL model's input is utilized in the suggested defense strategy to update a Q-table using the Bellman equation. Additionally, an adaptive epsilon-greedy policy is employed in a combined FL and reinforcement learning model to determine the optimal defense paths for Unmanned Aerial Vehicles (UAVs). Consequently, the UAVs effectively avoid jamming areas that have been identified in advance through the federated learning approach [31].

### **3.13.4 SMART CITY**

Smart cities offer reliable solutions to crucial issues with traffic, healthcare, education, security, and other areas. For a variety of uses, smart cities make extensive utilization of intelligent IoT devices [2]. To develop a framework for managing video data in smart cities, Chiu et al. [12] proposed an edge learning system that incorporates FL and semi-supervised learning. Live street videos can be gathered from connected vehicles' cameras, which are then sent to edge devices. Following that, a semi-supervised learning algorithm is run on each edge device to perform local video analytics on a variety of video segments. A "Federated Swapping" operation is suggested as a solution to the non-IID data issue to decrease data diversity and improve image classification accuracy.

Albaseer et al. [2] proposed the FedSem technique of semi-supervised FL that utilizes unlabeled data in smart cities. Their system comprises a fleet of autonomous vehicles traveling on various highways. These vehicles are controlled by a central edge server, which oversees the training process. Leveraging the vehicle's data (traffic sign images), each vehicle trains a local DNN model, and the central edge server only receives gradients from the edge

devices. In each learning round for model orchestration, multiple participants are selected under the supervision of a central server.

### **3.13.5 HUMAN ACTIVITY RECOGNITION (HAR)**

HAR aims to identify a person's activities based on observations about humans and their surroundings. Application areas for HAR include healthcare, behavior analysis, electroencephalography, surveillance, and gesture recognition [47]. Zhao et al. [55] proposed a system that uses FL to assist with training and edge devices to perform local health and activity monitoring. The proposed system collects and manages sensor and IoT data using the Databox platform, then utilizes this data to infer activities locally on an edge device, enabling health and activity monitoring. It utilizes a cloud server to manage various devices and conduct global LSTM model training without exposing their raw data.

Xiao et al. [47] proposed HARFLS, a FL system for HAR using wearable-based sensors. It extracts features with a perceptual extraction network (PEN) and shares model weights with the Federated averaging method. Furthermore, to lower the risk of data leakage during the distribution and uploading of weights, Homomorphic encryption is used.

## **3.14 OPEN ISSUES AND FUTURE RESEARCH DIRECTIONS**

The future research directions that we think would be exciting to work on and explore are discussed in this section.

### **3.14.1 INTELLIGENT BROKERS**

In a traditional hierarchical federated learning (FL) system, the role of intermediate aggregation servers lacks intelligent decision-making as they primarily function as intermediaries between the central server and the edge devices. Therefore, we emphasize the significance of implementing an intelligent broker within a hierarchical FL architecture. These intelligent brokers should make crucial decisions prior to distributing the central server's global model update to the edge devices, as well as before forwarding the local model updates from the edge devices to the central server. For example, instead of randomly selecting edge devices to participate in the training

process, they can choose the most suitable edge devices based on real-time computing resource information, dataset size, and historical data such as the accuracy of their local model updates. Intelligent brokers can also identify edge devices that are submitting irrelevant local updates and prevent their updates from being uploaded to the central server.

### **3.14.2 STRAGGLERS IDENTIFICATION MODULE**

Despite being applied to a heterogeneous system, FL is unable to efficiently utilize the system's heterogeneous resources. The performance of the FL training process on heterogeneous systems is significantly impacted by stragglers, as they take an excessive amount of time to report local updates. Therefore, it is crucial to establish a stragglers identification and mitigation module as a key enabler for heterogeneous FL systems. This module should be capable of identifying the causes of stragglers' existence, such as data skew, failures, and resource contention. It should then attempt to mitigate the effect of the stragglers, for example, by marking straggler nodes as deprecated when the central server collects local model updates.

### **3.14.3 CONTAINERIZED FL**

The variety of hardware devices employed in edge and IoT ecosystems poses a major challenge. These devices differ in terms of their sensors, computing power, and cost of network connectivity. Consequently, it is challenging to dynamically deploy local ML models to edge and IoT devices. Since we can remotely create, upgrade, and destroy applications in an elastic manner with minimum overhead thanks to containerization technology, we strongly support the need to dynamically deploy and manage ML models on diverse hardware devices through containerization technology.

### **3.14.4 INTELLIGENT SINGLE-POINT OF FAILURE PREDICTION**

In the vast majority of existing FL systems, the centralized architecture is extensively adopted. This architecture may result in several severe issues, such as a single point of failure. As a result, we emphasize the importance of intelligently predicting central server failures using ML methodologies before they occur.

### **3.14.5 DEVELOPING CLIENT-SPECIFIC ML MODEL**

The edge computing network is more diverse than traditional central networks, with clients differing based on compute, storage, and network resources, as well as data collection resources. Therefore, a client-specific ML model should be developed to deal with such heterogeneities.

## **3.15 CONCLUSIONS**

FL permits distributed client devices to jointly train a global ML model while preserving all training datasets on their devices, relieving ML from the requirement to maintain data on the cloud. This extends beyond the simple utilization of local models on mobile devices by additionally transferring model training to the device. This chapter introduces a review of the application of FL in E/F computing systems. The federation concept, FL workflows, federated learning vs. non-federated learning, the scale of the federation, FL network topology, and classification of a federation are presented and explained. We look into federated learning aggregation algorithms. The fundamentals of cloud, fog, and edge computing are introduced. Then, enabling DL at the E/F is presented. The investigations are being conducted into a number of case studies involving the application of federated learning in E/F computing. The unresolved problems and potential research directions are presented.

## **KEYWORDS**

- **edge computing network**
- **federated learning**
- **FL network topology**
- **intelligent brokers**
- **internet to things**
- **network connectivity**
- **traditional central networks**

## REFERENCES

1. Abreha, H. G., Hayajneh, M., & Serhani, M. A., (2022). Federated learning in edge computing: *A systematic survey*. *Sensors*, 22(2), 450.
2. Albaseer, A., Ciftler, B. S., Abdallah, M., & Al-Fuqaha, A., (2020). Exploiting unlabeled data in smart cities using federated edge learning. In: *2020 International Wireless Communications and Mobile Computing, IWCMC 2020* (pp. 1666–1671). <https://doi.org/10.1109/IWCMC48107.2020.9148475>.
3. Alom, M. Z., Taha, T. M., Yakopcic, C., Westberg, S., Sidiqe, P., Nasrin, M. S., Hasan, M., et al., (2019). A state-of-the-art survey on deep learning theory and architectures. *Electronics (Switzerland)*, 8(3). <https://doi.org/10.3390/electronics8030292>.
4. Alzubaidi, L., Zhang, J., Humaidi, A. J., Al-Dujaili, A., Duan, Y., Al-Shamma, O., Santamaría, J., et al., (2021). Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions. In: *Journal of Big Data* (Vol. 8, No. 1). Springer International Publishing. <https://doi.org/10.1186/s40537-021-00444-8>.
5. Beutel, D. J., Topal, T., Mathur, A., Qiu, X., Fernandez-Marques, J., Gao, Y., Sani, L., et al., (2022). *Flower: A Friendly Federated Learning Framework*. <https://flower.dev>.
6. Bouacida, N., & Mohapatra, P., (2021). Vulnerabilities in federated learning. *IEEE Access*, 9, 63229–63249. <https://doi.org/10.1109/ACCESS.2021.3075203>.
7. Buyya, R., & Srivama, S. N., (2019). Fog and edge computing: Principles and paradigms. Wiley. <https://doi.org/10.1002/9781119525080>.
8. Cao, H., (2019). *Developing an Analytics Everywhere Framework for the Internet of Things in Smart City Applications*.
9. Cao, J., Lian, Z., Liu, W., Zhu, Z., & Ji, C., (2021). HADFL: Heterogeneity-aware decentralized federated learning framework. *Proceedings—Design Automation Conference, 2021-Decem*, 1–6. <https://doi.org/10.1109/DAC18074.2021.9586101>.
10. Cao, K., Liu, Y., Meng, G., & Sun, Q., (2020). An overview of edge computing research. *IEEE Access*, 8, 85714–85728. <https://doi.org/10.1109/ACCESS.2020.2991734>.
11. Chen, J., & Ran, X., (2019). Deep learning with edge computing: A review. *Proceedings of the IEEE* (pp. 1–19). <https://doi.org/10.1109/JPROC.2019.2921977>.
12. Chiu, T. C., Shih, Y. Y., Pang, A. C., Wang, C. S., Weng, W., & Chou, C. T., (2020). Semi supervised distributed learning with non-IID data for the AIoT service platform. *IEEE Internet of Things Journal*, 7(10), 9266–9277. <https://doi.org/10.1109/JIOT.2020.2995162>.
13. De Donno, M., Tange, K., & Dragoni, N., (2019). Foundations and evolution of modern computing paradigms: Cloud, IoT, edge, and fog. *IEEE Access*, 7, 150936–150948. <https://doi.org/10.1109/ACCESS.2019.2947652>.
14. Directions, F., & Smith, V., (2020). Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 1–21.
15. Hakak, S., & Ray, S., (2020). A framework for edge-assisted healthcare data analytics using federated learning. *IEEE International Conference on Big Data, C*.
16. Hatcher, W. G., & Yu, W., (2018). A survey of deep learning: Platforms, applications, and emerging research trends. *IEEE Access*, 6, 24411–24432. <https://doi.org/10.1109/ACCESS.2018.2830661>.
17. Idrees, S. K., & Idrees, A. K., (2022). New fog computing enabled lossless EEG data compression scheme in IoT networks. *Journal of Ambient Intelligence and Humanized Computing*, 13(6), 3257–3270. <https://doi.org/10.1007/s12652-021-03161-5>.

18. Kairouz, P., McMahan, H. B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A. N., Bonawitz, K., et al., (2021). Advances and open problems in federated learning. In: *Foundations and Trends in Machine Learning* (Vol. 14, No. 1, 2). <https://doi.org/10.1561/2200000083>.
19. Kalyani, Y., & Collier, R., (2021). A systematic survey on the role of cloud, fog, and edge computing combination in smart agriculture. *Sensors*, 21(17). <https://doi.org/10.3390/s21175922>.
20. Khamparia, A., & Singh, K. M., (2019). A systematic review of deep learning architectures and applications. *Expert Systems*, 36(3), 1–22. <https://doi.org/10.1111/exsy.12400>.
21. Kholod, I., Yanaki, E., Fomichev, D., Shalugin, E., Novikova, E., Filippov, E., & Nordlund, M., (2021). Open-source federated learning frameworks for IoT: A comparative review and analysis. *Sensors (Switzerland)*, 21(1), 1–22. <https://doi.org/10.3390/s21010167>.
22. Li, Q., Wen, Z., Wu, Z., Hu, S., Wang, N., Li, Y., Liu, X., & He, B., (2021). A survey on federated learning systems: Vision, hype, and reality for data privacy and protection. *IEEE Transactions on Knowledge and Data Engineering* (pp. 1–44). <https://doi.org/10.1109/TKDE.2021.3124599>.
23. Liu, J., Huang, J., Zhou, Y., Li, X., Ji, S., Xiong, H., & Dou, D., (2022). From distributed machine learning to federated learning: A survey. *Knowledge and Information Systems*, 64(4), 885–917. <https://doi.org/10.1007/s10115-022-01664-x>.
24. Liu, L., Zhang, J., Song, S. H., & Letaief, K. B., (2020). Client-edge-cloud hierarchical federated learning. *IEEE International Conference on Communications*. <https://doi.org/10.1109/ICC40277.2020.9148862>.
25. Lu, Y., Huang, X., Dai, Y., Maharjan, S., & Zhang, Y., (2020). Differentially private asynchronous federated learning for mobile edge computing in urban informatics. *IEEE Transactions on Industrial Informatics*, 16(3), 2134–2143. <https://doi.org/10.1109/TII.2019.2942179>.
26. Malekijoo, A., Malekijoo, H., Alizadeh-Shabdiz, F., Fadaeieslam, M. J., Homayounfar, M., & Rawassizadeh, R., (2021). *FEDZIP: A Compression Framework for Communication-Efficient Federated Learning* (pp. 1–17). ArXiv Preprint ArXiv:2106.05508.
27. Mammen, P. M., (2021). *Federated Learning: Opportunities and Challenges*. <http://arxiv.org/abs/2101.05428>.
28. McMahan, B., Moore, E., Ramage, D., Hampson, S., & Arcas, B. A., (2017). Communication-efficient learning of deep networks from decentralized data. In: *Artificial Intelligence and Statistics* (pp. 1273–1282). PMLR.
29. Mohammadi, M., Al-Fuqaha, A., Sorour, S., & Guizani, M., (2018). Deep learning for IoT big data and streaming analytics: A survey. *IEEE Communications Surveys and Tutorials*, 20(4), 2923–2960. <https://doi.org/10.1109/COMST.2018.2844341>.
30. Mothukuri, V., Parizi, R. M., Pouriyeh, S., & Huang, Y., (2020). A survey on security and privacy of federated learning. *Future Generation Computer Systems*. <https://doi.org/10.1016/j.future.2020.10.007>.
31. Mowla, N. I., Tran, N. H., Doh, I., & Chae, K., (2020). AFRL: Adaptive federated reinforcement learning for intelligent jamming defense in FANET. *Journal of Communications and Networks*, 22(3), 244–258. <https://doi.org/10.1109/JCN.2020.000015>.
32. NVIDIA FLARE, (n.d.). Retrieved from: <https://nvidia.github.io/NVFlare/> (accessed on 25 January 2024).
33. Paddle Paddle, (2022). *Paddle Federated Learning*. <https://paddlefl.readthedocs.io/en/latest/introduction.html> (accessed on 25 January 2024).

34. Pappas, C., Chatzopoulos, Di., Lalis, S., & Vavalis, M., (2021). IPLS: A framework for decentralized federated learning. In: *2021 IFIP Networking Conference, IFIP Networking 2021* (Vol. 1, No. 1). Association for Computing Machinery. <https://doi.org/10.23919/IFIPNetworking52078.2021.9472790>.
35. Qayyum, A., Ahmad, K., Ahsan, M. A., Al-fuqaha, A., & Qadir, J., (n.d.). *Collaborative Federated Learning for Healthcare: Multi-Modal COVID-19 Diagnosis at the Edge* (pp. 1–10). ArXiv Preprint.
36. Ryffel, T., Trask, A., Dahl, M., Wagner, B., Mancuso, J., Rueckert, D., & Passerat-Palmbach, J., (2018). *A Generic Framework for Privacy-Preserving Deep Learning* (pp. 1–5). <http://arxiv.org/abs/1811.04017>.
37. Shaheen, M., Farooq, M. S., Umer, T., & Kim, B. S., (2022). Applications of federated learning: taxonomy, challenges, and research trends. *Electronics (Switzerland)*, 11(4). <https://doi.org/10.3390/electronics11040670>.
38. Shiri, H., Park, J., & Bennis, M., (2020). Communication-efficient massive UAV online path control: Federated learning meets mean-field game theory. *IEEE Transactions on Communications*, 68(11), 6840–6857. <https://doi.org/10.1109/TCOMM.2020.3017281>.
39. Sufian, A., Alam, E., Ghosh, A., Sultana, F., & De, D., (2021). Deep learning in computer vision through mobile edge computing for IoT. *Mobile Edge Computing*. Springer, Cham. <https://doi.org/10.1007/978-3-030-69893-5>.
40. Tan, J., Liang, Y., Luong, N. C., & Niyato, D., (n.d.). *Toward Smart Security Enhancement of Federated Learning Networks*, 1–7.
41. TensorFlow. (2022). *TensorFlow Federated: Machine Learning on Decentralized Data*. Tensorflow.Com. <https://www.tensorflow.org/federated> (accessed on 25 January 2024).
42. Tmamna, J., Ayed, E. B., & Ayed, M. B., (2020). Deep learning for the internet of things in fog computing: Survey and open issues. In: *5<sup>th</sup> International Conference on Advanced Technologies for Signal and Image Processing (ATSIP)*. IEEE.
43. Wang, C., & Yao, Y., (2021). *Vertical Federated Learning Without Revealing Intersection Membership*. ArXiv Preprint ArXiv:2106.05508.
44. Wang, X., Han, Y., Leung, V. C. M., Niyato, D., Yan, X., & Chen, X., (2020). Convergence of edge computing and deep learning: A comprehensive survey. *IEEE Communications Surveys and Tutorials*, 22(2), 869–904. <https://doi.org/10.1109/COMST.2020.2970550>.
45. We bank, (2019). *Federated AI Technology Enabler. (FATE)*. <https://github.com/FederatedAI/FATE> (accessed on 25 January 2024).
46. Xia, Q., Ye, W., Tao, Z., Wu, J., & Li, Q., (2021). A survey of federated learning for edge computing: Research problems and solutions. *High-Confidence Computing*, 1(1), 100008. <https://doi.org/10.1016/j.hcc.2021.100008>.
47. Xiao, Z., Xu, X., Xing, H., Song, F., Wang, X., & Zhao, B., (2021). A federated learning system with enhanced feature extraction for human activity recognition. *Knowledge-Based Systems*, 229, 107338. <https://doi.org/10.1016/j.knosys.2021.107338>.
48. Xu, G., Member, S., Li, H., Member, S., Liu, S., & Member, S., (2019). *Verify Net: Secure and Verifiable Federated Learning*. <https://doi.org/10.1109/TIFS.2019.2929409>.
49. Yang, Q., Liu, Y., Chen, T., & Tong, Y., (2019). Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology*, 10(2), 1–19. <https://doi.org/10.1145/3298981>.
50. Ye, D., Yu, R., Pan, M., & Han, Z., (2020). Federated learning in vehicular edge computing: A selective model aggregation approach. *IEEE Access*, 8, 23920–23935. <https://doi.org/10.1109/ACCESS.2020.2968399>.

51. Yin, X., Zhu, Y., & Hu, J., (2021). A comprehensive survey of privacy-preserving federated learning: A taxonomy, review, and future directions. *ACM Computing Surveys*, 54(6). <https://doi.org/10.1145/3460427>.
52. Yuan, J., Xu, M., Ma, X., Zhou, A., Liu, X., & Wang, S., (2020). *Hierarchical Federated Learning Through LAN-WAN Orchestration*. ArXiv Preprint ArXiv:2010.11612, 2020. <http://arxiv.org/abs/2010.11612>.
53. Zhan, Y., Zhang, J., Hong, Z., Wu, L., Li, P., & Guo, S., (2022). A survey of incentive mechanism design for federated learning. *IEEE Transactions on Emerging Topics in Computing*, 10(2), 1035–1044. <https://doi.org/10.1109/TETC2021.3063517>.
54. Zhang, P., Sun, H., Situ, J., Jiang, C., & Xie, D., (2021). Federated transfer learning for IIoT devices with low computing power based on blockchain and edge computing. *IEEE Access*, 9, 98630–98638. <https://doi.org/10.1109/ACCESS2021.3095078>.
55. Zhao, Y., Haddadi, H., Skillman, S., Enshaeifar, S., & Barnaghi, P., (2020). Privacy-preserving activity and health monitoring on data box. *EdgeSys 2020—Proceedings of the 3<sup>rd</sup> ACM International Workshop on Edge Systems, Analytics and Networking, Part of EuroSys 2020* (pp. 49–54). <https://doi.org/10.1145/3378679.3394529>.



Taylor & Francis

Taylor & Francis Group

<http://taylorandfrancis.com>

## CHAPTER 4

---

# Adopting Federated Learning for Software-Defined Networks

AKARSH K. NAIR,<sup>1</sup> JAYAKRUSHNA SAHOO,<sup>1</sup> and GAURAV JASWAL<sup>2</sup>

<sup>1</sup>*Department of Computer Science and Engineering,  
Indian Institute of Information Technology, Kottayam, Kerala, India*

<sup>2</sup>*iHUB and HCI Foundation, Indian Institute of Technology, Mandi,  
Himachal Pradesh, India*

---

### ABSTRACT

Federated learning (FL) is a distributed machine learning (DML) method that utilizes client device participation to train large-scale models. It is considered a privacy-preserving mechanism as client devices only share the model updates while keeping the training data private. However, the quality of the generated model and the FL system as a whole depend on the total count of active client devices and their gradient contributions. In traditional computer networks, integrating FL can be challenging due to the need to gather a higher number of clients and develop an effective collaborative strategy. The introduction of Software Defined Network (SDN) has facilitated a more open and flexible networking approach with separate control and local planes, making the integration of FL a genuine possibility. The standardization of protocols for multiple devices in the network has also accelerated integration attempts. The control plane in SDN is structured in a way that already has multiple layers, with lower layers consisting of multiple controllers and the upper layer usually housing an individual controller. This architecture is well-suited for FL settings, as FL systems also follow a similar pattern with a single server and multiple client devices. Therefore, integration efforts have been accelerated, along with

---

Federated Learning: Principles, Paradigms, and Applications.

Jayakrushna Sahoo, Mariya Ouaisse, & Akarsh K. Nair (Eds.)

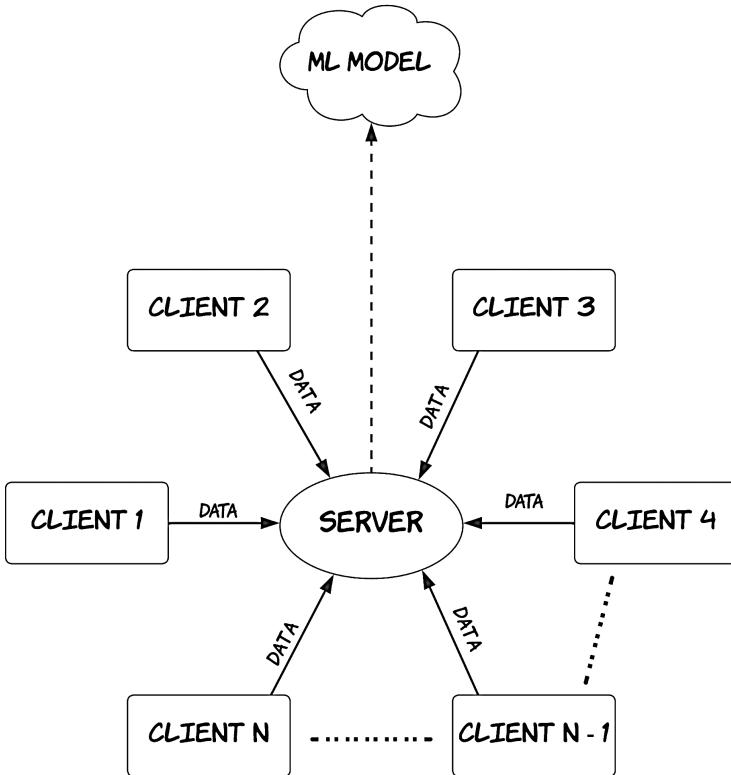
© 2025 Apple Academic Press, Inc. Co-published with CRC Press (Taylor & Francis)

attempts to improve privacy preservation and ensure effective collaboration among clients. This article aims to provide a comprehensive analysis of various mechanisms associated with FL, SDN, and their potential integration attempts. It also highlights major challenges such as incentive mechanisms in FL architecture, incentive distribution policies, privacy concerns, and issues related to model aggregation. Additionally, a basic classification of state-of-the-art mechanisms that address the aforementioned challenges is presented, followed by a discussion of prospective research paths.

## **4.1 INTRODUCTION**

Machine learning (ML) is the term for computational methods that draw on past performance to achieve goals such as prediction accuracy or performance enhancement. Experience typically refers to the skills gained from data and applied to subsequent projects. As a result, ML approaches are often data-driven and have gained popularity as the benefits of big data, cloud computing, and the Internet of Things (IoT) become more apparent. In the typical scenario, traditional ML approaches are centralized, as shown in Figure 4.1, where a central device accumulates data received from various client devices and performs model training. The performance metrics of the generated model are thus dependent on the volume and usefulness of the centralized data.

A major issue with such traditional approaches is the associated centralized data storage. Sometimes, such data may not be credible and may also be unavailable. In several instances, clients prefer not to share their data due to security reasons, especially with sensitive data like medical records. Thus, ensuring the availability of credible data becomes a challenge in centralized ML approaches, which also acts as a significant hurdle for further development. Federated learning is an emerging distributed ML approach that overcomes most of the issues associated with traditional ML. It does not rely on centralized training data and model training. As shown in Figure 4.2, FL systems consist of a central server and multiple client devices. During model training, the central server initially sends a global model to all the client devices. The client devices then train the received model using their local data and generate an intermediate model on the device itself. The updated weights of the generated model are transferred to the server as gradients, which are aggregated to generate the updated global model. This process continues over a set of iterations until a pre-set accuracy is achieved by the global model. Some of the primary benefits of FL include improved



**FIGURE 4.1** Conventional machine learning model training.

data security, privacy preservation, and storage efficiency at the server end. FL servers do not require huge memory requirements, resulting in economic benefits. In addition to these, FL servers also face less computation overload as they only take part in model training in a limited manner. FL servers also help in generating and transmitting higher quality global models compared to the availability of in-house data and individual contributions to training. As a result, FL approaches have been applied in various domains such as healthcare, smart city-based applications, human-computer interaction systems, autonomous vehicular systems, and more. FL systems rely on client devices to generate local models from onsite data and cooperate with the FL server to generate the global model. However, this gives rise to a new set of challenges. These client devices are typically part of a larger network and may differ from each other in multiple ways. They may not have a common communication protocol, specifications, or any other means of connection other than their collaborative FL training. Consequently, FL servers face

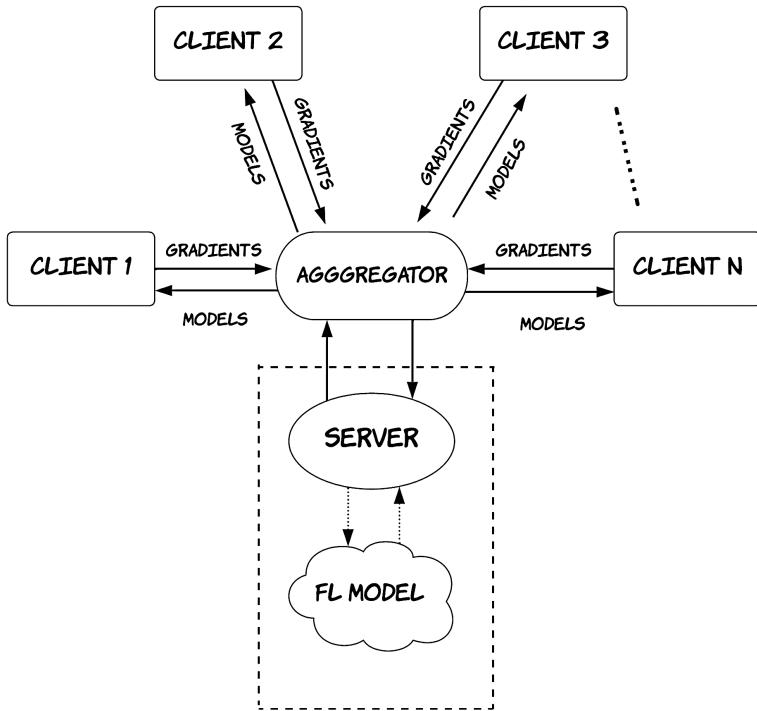
significant difficulties in coordinating these clients, providing global security measures, intrusion detection systems, and dealing with malicious clients. Despite being a popular topic in academia and research, FL techniques have not been widely adopted in practice.

Similar to FL, SDNs are an emerging paradigm in networking where the classical network architecture and working methodologies are transformed into software-based techniques. The initial proposal of SDN was made by the Censlate research group at Stanford University, and the system architecture typically consists of an application, control plane, and data planes. In SDN, the control function and the forwarding function are decoupled. The control functionalities are performed in a centralized manner at the control plane, while the forwarding functionalities are carried out at the data plane. SDN architecture also enables standardization in protocols and specifications among various devices, creating opportunities for inter-controller collaborations and collaborations between controllers and SDN switches. SDN controllers can use open and standardized protocols to program the behaviors of SDN switches, enforce global regulations on SDN switches, and aggregate data generated at SDN switches, thus serving as a potential primitive for FL methods. In terms of architecture, SDN controllers can be stacked with several local controllers at the lower layer and a single controller located at the upper layer.

The placement of the controllers at the lower level can be done in close proximity or at the network edge, thus facilitating efficient collection of local data and models. The single controller from the upper layer can be treated as the central server, where the local models get aggregated and the global model gets generated via classical standardization procedures. The distributed nature of the SDN control plane facilitates smooth implementation of FL approaches in such environments. Additionally, the distributed control plane enables unified processing for clients, regardless of the structure of the network topology, distribution of users, or data storage. Therefore, SDN controllers have the full capability to manage the willingness of each client to participate in certain events, generate solutions for security issues such as intrusion detection, manage unusual network occurrences, effectively allocate resources, and provide training in FL scenarios, among other tasks.

In the current scenario, the application of Federated Learning (FL) on Software-Defined Networking (SDN) is a developing interdisciplinary research area that has the potential to generate efficient communication methods. However, the full extent of this scheme is yet to be explored in recent research, and there may be obstacles preventing its further development. This article provides a comprehensive analysis of the existing literature on the combination of FL and SDN at multiple levels, discussing the

possibilities for integration to optimize various aspects. Additionally, the article presents various incentive mechanisms, privacy analysis, and aggregation methods in such systems.



**FIGURE 4.2** Federated learning model training.

## 4.2 FEDERATED LEARNING

In this section, we provide a basic introduction to federated learning, allied terminologies, and architectural paradigms.

Federated learning (FL) refers to a distributed machine learning paradigm where the main goal is to overcome the security and privacy concerns associated with conventional ML training and data silos. The usual FL architecture consists of a central server and various client devices, as shown in Figure 4.2. As the model training is collaborative, the final model is generated over a set of iterative training procedures. In any given FL training scenario, it can be divided into four phases.

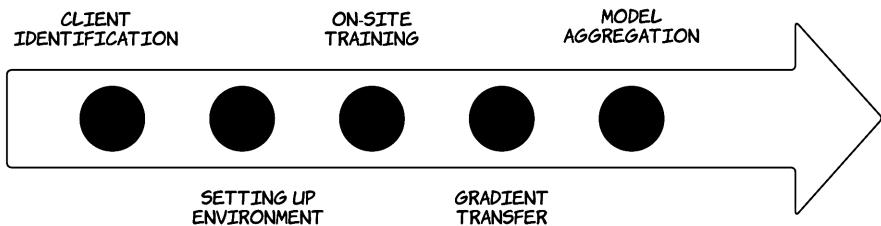
1. **Initialization of Training:** The initial step of Federated Learning (FL) is to transfer a model to the clients to commence the training. The model thus transferred is referred to as the global model, and the server device possesses the global model at the initial phase. It is the sole responsibility of the server to transfer the global model to every client participating in the training.
2. **On-Site Training:** Once the global model is received at the client's end, the devices perform training on the model using their in-house data. This training occurs for several iterations until a pre-set accuracy is achieved.
3. **Gradient Transfer:** After each iteration of local model training, the gradients are transferred to the global model for aggregation and the generation of a new global model.
4. **Model Aggregation:** Once gradients are received from the client devices, the initial process at the server is to perform aggregation of these gradients. For aggregation, techniques such as FedAvg are employed in usual scenarios. Post aggregation, an updated version of the global model is generated, which is then transferred back to the client devices for further training.

The various phases of the procedure are depicted in Figure 4.3. The aforementioned procedures are repeated by the central server and clients until the preset requirements are met. During training procedures, FL clients are required to transfer gradient updates only and keep their data locally, thus always ensuring data privacy. Since client devices typically own a smaller volume of data, which by itself is not feasible for any sort of training procedure, the training done on received global models makes it feasible for the device to gain access to quality models and performance results as well.

#### **4.2.1 VARIOUS CLASSIFICATIONS OF FEDERATED LEARNING**

Even though criteria such as network topology and ML models can be used to perform the categorization of FL, we aim to classify FL solely based on the data partitioning scheme employed during local model training. In the current scenario, data partitioning is approached in two ways: feature space and sample space. Sample space is typically used to refer to an entity that identifies the data sample, while feature space represents the set of all features offered by the sample. During FL training, participating clients utilize their local data for training, which may consist of different sample and vector

spaces. Based on these factors, FL is currently classified into three categories as described below:



**FIGURE 4.3** Various phases of federated learning model training.

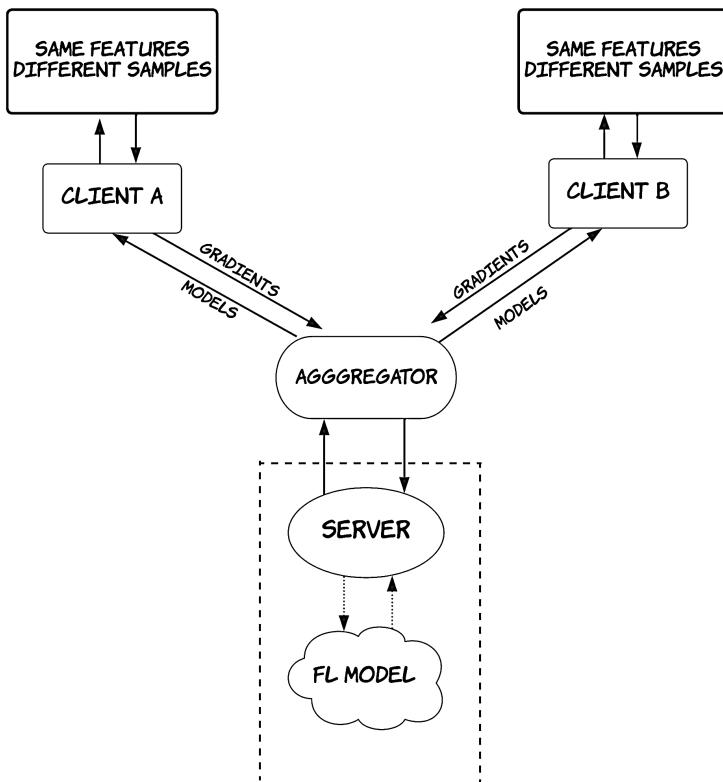
#### 4.2.1.1 HORIZONTAL FEDERATED LEARNING

Horizontal Federated Learning (FL) is used to describe an FL approach where the training dataset comprises highly overlapping features, but the individual users will have very limited similarity. Figure 4.4 depicts a sample of the Horizontal FL approach, and it is evident that the system uses identical features from various users from the initial dataset for performing local model training and global model aggregation as well. A classical use case of horizontal FL is the instance of banks. Most of the banking industries have user databases having highly similar or identical features, even though the individual users may not be the same in every case. Currently, some of the commonly used FL approaches come under the horizontal FL category only. The healthcare industry is also a wide employer of horizontal FL approaches.

#### 4.2.1.2 VERTICAL FEDERATED LEARNING

In contrast to horizontal Federated Learning (FL), vertical Federated Learning is applicable to scenarios where the dataset consists of identical users with highly dissimilar features. Figure 4.5 illustrates the workflow of a classical vertical FL system. In some literature, vertical FL has been presented with a third party, which is often claimed to provide better authentication and encryption features during model training. Vertical FL is commonly used in different industries that may share common factors, such as supermarkets and financial institutes operating in the same location. This is because although these firms may have highly identical user databases, the feature space they present will be completely different. Therefore, model generation in such

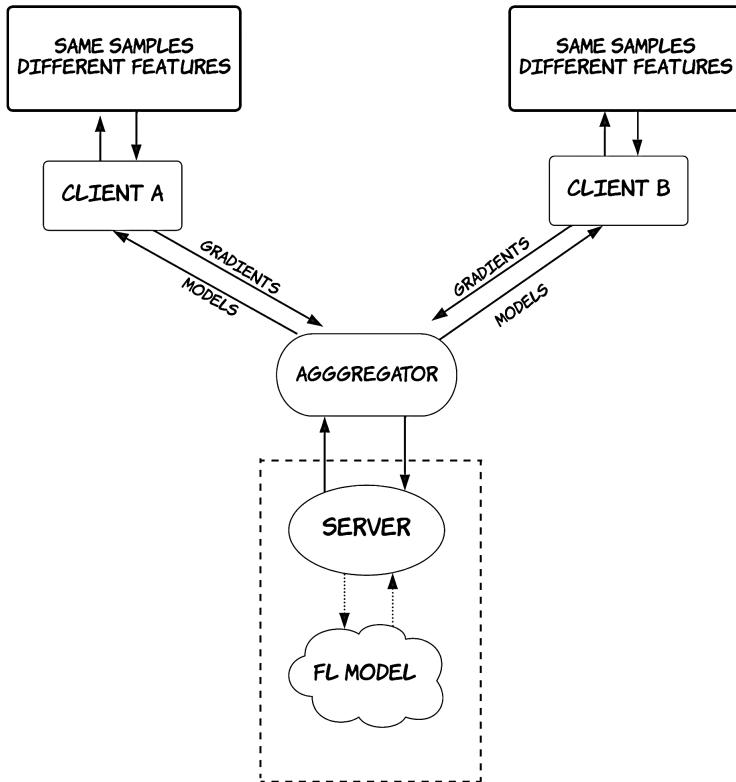
cases follows the vertical FL methodology, and any form of communication among the clients is prohibited to ensure the authenticity and privacy of the procedure.



**FIGURE 4.4** Horizontal federated learning architecture.

#### 4.2.1.3 FEDERATED TRANSFER LEARNING

The third category based on data distribution in FL is the federated transfer learning scheme. This scheme is employed when two different datasets have very little in common in terms of both the user space and the feature space, thus not fitting into both horizontal and vertical FL classification scenarios. Unlike horizontal and vertical FL, federated transfer learning is not limited to a few use cases, as the necessity of federated transfer learning can only be determined with respect to the data distribution at the moment.



**FIGURE 4.5** Vertical federated learning architecture.

### 4.3 SOFTWARE DEFINED NETWORKS

Three types of interfaces are defined in SDN architecture: the southbound, northbound, and east/westbound interfaces. The southbound interface is positioned between the control plane and the data plane, while the northbound interface lies between the control and application planes. In contrast to the other two, the east/westbound interface can be found inside the control plane only. The southbound interface is used by the control plane to manage the configuration and flow of data among switches in the data plane. Similarly, the northbound interface's main functionality is to retrieve the application's information from the global view maintained by controllers and manage the convergence of ideal configurations and behaviors to the data plane. The west/eastbound interface is solely responsible for ensuring compatibility

and interoperability among various controllers. Due to these features, which further ensure openness and flexibility, SDN has the potential to be applied to several other domains as well.

SDNs have the complete capability to successfully work around challenges faced by FL systems related to issues such as client devices and time management. In usual networking systems, integration of FL is a challenging task due to multiple factors such as insufficient participant count, lack of standardized working environment, and so on. In the context of FL in SDN, the SDN controller can usually take up the role of the server as they already have an inbuilt mechanism to gather data from multiple client devices and also possess higher-level computational capabilities and storage space, thus enabling them to function better when compared to client devices. In conventional networking architecture, security mechanisms to prevent data leakage or ensure privacy are void. Additionally, the lack of intrusion detection features is also a major issue in such networks. With SDN, the cooperative operations done by controllers and switches introduce standardization in multiple aspects, thus helping in overcoming such shortcomings on the security side and also introducing better defensive measures as well. Furthermore, in contrast to conventional networks, the central server in FL is not capable of managing several clients. Whereas in the proposed model, if more than one SDN controller is present in the control plane, it is possible for one SDN controller to oversee other SDN controllers utilizing open and standardized interfaces.

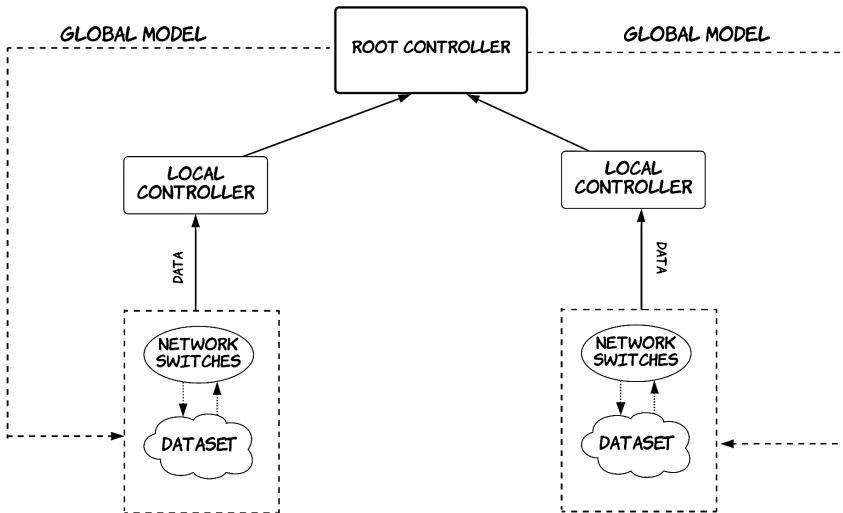
## **4.4 INTEGRATION OF FEDERATED LEARNING AND SDN**

This section discusses the existing literature on attempts to integrate FL and SDN. Some of the major challenges faced during the same have also been presented:

### **4.4.1 INTEGRATION OF FL AND SDN**

Usual SDN systems comprise control planes of both centralized and distributed schemas. The controller existing as a single entity is usually referred to as the centralized one, and its prime role is to monitor the functionalities of all other fellow devices in the control plane. On the other hand, the distributed version of the control plane usually comprises several controller devices, with each device expected to manage a smaller group

of controllers over the network. Figure 4.6 shows the common layer-based architecture followed when stacking multiple controllers in the distributed control plane.



**FIGURE 4.6** FL implementation on SDN.

A multi-layer distributed control plane usually consists of a root controller located at the top layer, and a set of local controllers situated at the lower layer. The layered architecture of the distributed control plane greatly enhances the computational capabilities and scalability of SDN systems. When Federated Learning (FL) is implemented on SDN systems, the local controllers act as client devices and handle most of the FL procedures. These devices are responsible for storing the training data, performing training, and transferring gradient updates. On the other hand, the root controller functions as a server and solely focuses on model aggregation. Furthermore, controllers possess greater computational and storage capabilities compared to switches and hosts, which improves the time efficiency for local model generation and global aggregation.

#### 4.4.2 CHALLENGES FACED

The integration of FL and SDN gives rise to multiple benefits for collaborating central devices with client systems. However, such methodologies

face intense problems and challenges. The rest of the section will discuss various challenges associated with the same.

#### ***4.4.2.1 DEVELOPMENT OF INCENTIVIZATION POLICIES***

Incentivization policies are usually employed to encourage certain client devices to participate and perform better in FL model training. In the context of FL training taking place in SDN environments, the SDN controller makes it mandatory for all client systems in the control plane to participate in the training. With the addition of such policies, rewards can be generated and distributed among clients, thus increasing their willingness to participate in model training. This indirectly creates opportunities to improve model quality through better performance from clients. In addition to this, incentivization policies also provide the central server and clients with a way to balance resource requirements that FL and other SDN controller-based tasks compete for.

#### ***4.4.2.2 SECURITY AND PRIVACY-RELATED ISSUES***

Enhanced security and privacy have been the building blocks and motivation for the development of FL. In FL, client devices, known as local controllers, are required to perform training and transfer the generated local gradients instead of local data. This approach inherently reduces security concerns. However, the integration of FL and SDN introduces some challenges. Features of SDN, such as openness and programmability, may compromise the standard security features of FL. Furthermore, in the SDN architecture, certain features of controllers and interfaces can disrupt the normal data flow in FL, affecting the aggregation procedure, generated models, and even the in-house data.

#### ***4.4.2.3 ROLE OF GLOBAL MODEL AGGREGATION METHODS***

In FL systems, the aggregation method employed usually plays a significant role in determining the efficiency and performance of the generated model and the system as a whole. Although SDN controllers can utilize in-built protocols and interfaces to address this issue, the speed and quality of such methods remain the main concerns when implementing FL in SDNs.

#### **4.4.3 MANAGING ORGANIZATION-BASED ISSUES IN FL AND SDN**

In a distributed control plane, controllers are usually stacked in an organized and proportionate manner. In the case of a data plane with a large number of SDN switches, the easiest method to efficiently manage it is to have a multi-layer control plane with several controllers, each responsible for managing individual partitions of the data plane. This forms an organized structure.

However, as the number and distribution of users increase, network partitions can become disorganized. This scenario is quite common. Such issues can lead to difficulties in selecting client devices, devising new incentive mechanisms, and generating aggregation algorithms for FL systems. The SDN system architecture allows controllers to collaborate and manage the performance of switches at various levels. By integrating FL in SDN, we can generate better alternatives to overcome these difficulties.

### **4.5 INCENTIVIZATION POLICIES**

#### **4.5.1 GAME THEORY**

Game theory is one such branch of economics that has found high levels of applicability in computer science as well. It mainly deals with developing strategies for groups or individuals to accomplish certain tasks in an optimized manner. Applications related to edge computing and mobile systems widely employ game theory for generating incentivization policies. In FL systems, it has also found its way for the same purpose. The following session will summarize some of the major models based on game theory currently being employed for generating incentivization in the FL context.

- The Stackelberg model is one of the well-known game theory models that operates through a leader-follower mechanism. In the context of Federated Learning (FL), the central server assumes the role of the leader, while the clients act as followers. The followers are responsible for providing regular updates on various system resource availability to the leader. Based on the budget allocated for achieving a specific goal, the leader determines the volume of returns to be shared with each follower. Although the system's functionality may be similar, the definition of leader and follower can vary across different methodologies. Sakariya et al. [1] present the standard methodology where the server system serves as the leader and the client system

as the followers. Similarly, Khan et al. describe the base stations in the network as the leaders and the participating client devices as the followers. In both cases, the objective of the system is to enhance the accuracy of the global model as a whole. In another article by Feng et al. [2], mobile devices assume the role of leaders, while the server device, which owns the global model, acts as the follower. The rationale behind this approach was to enable the server device to determine the training parameters, such as data size, for all devices in the network, regardless of their status, while performing Federated Learning (FL) in a community-based network. As mentioned earlier, Sun et al. [3] applied the classical Stackelberg model in FL, but they used it to adjust the total count and level of client devices participating in model training. While several articles have been built on the Stackelberg gaming model, some authors argue that the model is equally applicable to both followers and leaders. They propose dividing the Stackelberg game into two phases: the leader releases various stages of rewards, and then the followers attempt to maximize their benefit. Zhan et al. [4] attempt to incorporate deep reinforcement learning into the aforementioned two-level Stackelberg model to determine idealistic pricing and training approaches for the central server and client devices. However, despite the development of several approaches based on game theory, a group of researchers also contradicts the validity of game theory-based models, claiming that such strategies do not yield efficient results in increasingly complex network environments.

- There are various sub domains of game theory that have been employed for the same purpose in federated learning (FL) and similar domains. Auction theory is one such sub domain of game theory that operates based on the principles of auctioneers and bidders. In the context of auction theory, the typical scenario involves auctioneers releasing a task and bidders attempting to bid for the task by quoting their estimated cost for FL training. The final results of the auction are determined by selecting the winners based on the cost quoted by the bidders. Le et al. [5] have presented a couple of articles based on auction theory. In those articles, the base station assumes the role of the auctioneer, while the client devices act as bidders. The base station releases the initial task, and the client devices submit their bids by sending them to the base station for cost estimation. The base station evaluates the bids and ultimately determines the most

suitable client from the set of bidders, considering the relevance of the algorithm, and pays the reward. In a similar context, Yu et al. [6] attempt to motivate client devices to process good quality data by employing a dynamic pay-off sharing-based strategy and aiming to eliminate non-matching returns and contributions. However, a major issue with such approaches is that the bidding price is the only constraint evaluated in these auctions, whereas in reality, many other factors also play a crucial role in making such decisions. Therefore, various other dimensions should also be considered before making such decisions. One such article has been presented by Zeng et al. [7], where the authors introduce a multi-dimensional framework with an incentivization methodology. The approach aims to derive an optimal strategy for edge computing devices.

- The non-cooperative approach is another sub domain in game theory where each individual in the network is considered a player, and each player competes with one another. FL systems working on a non-cooperative gaming-based approach simply means that the FL server and clients are non-cooperative. An article based on the same is presented by Tang et al. [8], where the authors maximize the performance and balance the clients' situations. Pan et al. [9] have also proposed an article on similar grounds to determine the idealistic security strategy for clients and also items to attract participants for FL training.
- The Shapley value is another important term associated with cooperative game theory, where the primary focus is on the fair distribution of both gains and costs among various collaborating entities. As the name suggests, cooperative game theory refers to a system consisting of a group of players who work together. Several articles have discussed the application of the Shapley value in different contexts. In the article by Song et al. [10], the Shapley value is used to determine returns. Similarly, in article [11], the same measure is employed to calculate the contributions of various clients during training, in order to allocate rewards to them.

A fundamental game theory-based model known as Nash equilibrium is also applied in FL for incentivization. The methodology was proposed by John Nash, who stated that no players in the network can unilaterally alter their strategy to increase their payoff. As with the previously discussed models, the Nash equilibrium approach is also a cooperative model where

every single strategy is a mutual responsibility for fellow devices, and no client has any specific motivation to unilaterally alter the system strategy. The initial proposal of the discussed approach in the context of FL was made by Gupta et al. [12].

#### **4.5.2 CONTRACT THEORY**

In contract theory-based approaches, the main focus is on how economic actors construct contractual agreements. In the context of Federated Learning (FL), contract theory is primarily applied to create mechanisms that enable client devices to reach a consensus after evaluating themselves. Before implementing contract theory, several factors such as local model training, global model aggregation, and participant incentivization need to be considered. The initial step in employing the policy is carried out by the server, which generates a set of contracts with specific requirements. The client devices then choose a contract to follow while performing the training. Similar to other FL methodologies, locally trained models are sent to the server as gradients, which are then aggregated by the server and compensation is provided to the clients. The applications of contract theory are not limited to attracting devices for FL training, but also extend to encouraging participants to increase their contribution to training. However, a major challenge with contract theory models is the feasibility of generating an incentivization mechanism that can achieve high performance in complex networking systems. Therefore, other technologies are integrated with contract theory to overcome these limitations. An article by Lim et al. [14] presents the combination of contract theory with game theory for designing a hierarchical incentive mechanism in the FL scenario.

#### **4.5.3 FAIRNESS**

Fairness is a highly desirable factor in incentivization. In usual scenarios, gaming and contract theories mainly concentrate on distributing rewards to participants in FL training and the reward allocation may not be fair as each device may have different constraints. To rectify such unfairness in reward allocation and to optimize benefits for all participants, Qu et al. [15] proposed a fair incentive mechanism where the clients were able to truly report their contributions and losses as it is in FL training. A major issue with the initial proposal was that all clients may not be honest and some may have malicious

intentions. Thus, Wang et al. [16] proposed an updated version of the existing approach that calculated client contributions by deleting models. The values of the local models in training were better reflected in such an approach and improved the fairness of the FL system as a whole. Apart from such an approach, fairness is also evaluated in terms of reputation. Reputation-based incentive mechanics usually calculate the client's reputation value based on their attributes post-training. Such an incentive mechanism is proposed by Zhao et al. [17] for blockchain-based FL. Initially, the list of all participants was initialized with equal reputation, and reputation was updated after each iteration. The quality of data provided was also a criterion for increasing or decreasing the reputation. Rehman et al. [18] also proposed a similar scheme with the system comprising their components, edge devices, fog nodes, and cloud servers. Article [17] proposed a model with a randomly generated leader and client having the capability to rank one another.

## 4.6 SECURITY AND PRIVACY

The security and privacy preservation aspects of Federated Learning (FL) and Software-Defined Networking (SDN) have been extensively researched in recent years. While FL was initially designed with built-in privacy preservation, further research has revealed potential vulnerabilities in FL scenarios, such as inference attacks. Inference attacks exploit shared gradients and parameters to gather private user information, which can then lead to other attacks like reconstruction and GAN-based attacks. These attacks can compromise the quality of models and systems, undermining the initial claims of privacy preservation. In the following section, we will present various methodologies employed as privacy preservation measures in FL.

Based on the level of protection they offer, these methods are generally classified into three sub-categories: encryption-based, decentralization-based, and intrusion detection systems.

### 4.6.1 ENCRYPTION-BASED MECHANISMS

Encryption is a commonly employed methodology for improving data security and preserving privacy. Even though encryption has been employed in various contexts, in terms of FL, the major types of encryption methodologies used are differential privacy (DP), secure multi-party computation (SMC), and Homomorphic encryption (HE).

- **Differential Privacy:** It is one of the most commonly employed methods for enhancing privacy preservation. To successfully employ differential privacy, various types of noise are embedded into the data. These noise mechanisms make sensitive data appear perturbed, thus ensuring added privacy [19]. Wei et al. discussed one of the initial applications of differential privacy in the context of federated machine learning, using the Gaussian noise addition mechanism [20]. Geyer et al. also presented a similar article [21], where they employed a Gaussian mechanism to hide user contributions in FL training. Tri et al. presented a Bayesian differential privacy-based mechanism in [22], where they proposed adding noise to the privacy loss previously obtained from the data distribution.

Applications based on differential privacy can be classified into two types: global differential privacy (DP) and local DP. The successful implementation of certain differential privacy applications requires a third party to incorporate noise mechanisms into data in Federated Learning (FL). These differential privacy mechanisms are commonly known as global DP. Global DP typically requires a smaller amount of noise to be added to the system, thus ensuring higher levels of accuracy. Global DP is applied in various domains, such as healthcare and personal privacy protection scenarios. In contrast to global DP, local DP does not require a trusted third party, allowing users to directly encrypt the data before uploading it. Some researchers argue that local DP provides better results compared to global DP. Cao et al. [23] proposed a method for local DP based on Gaussian noise to enhance the security of power systems. In any DP mechanism, the amount of noise to be added is a crucial factor. Therefore, different researchers have presented different perspectives on this matter. Some researchers claim that the amount of noise should be directly proportional to the dimensionality of the data, as individual dimensions have varying impacts on the decision. Several articles have explored different policies regarding dimensions and their impact on noise addition. One such article, presented by Liu et al. [24], employs the Top-k concept to select the most significant dimensions based on their contributions in training.

- **Secure Multi-Party Computation:** Another sub domain of cryptography used to address privacy-related issues in collaborative computing systems is Secure Multi-Party Computation (SMC). SMC functions by forming a cluster of devices that follow specific protocols and operate in a controlled environment to protect private information. In

Federated Learning (FL), SMC has been widely adopted as a standard privacy processing mechanism for jointly performing training without the need for individual parties to share their data. The applicability of SMC in FL was initially investigated by Zhu et al. [25], and a proper design for integrating SMC in FL was presented by Bonawatz et al. [26]. Kanagavelu et al. [27] proposed a notable work on MPC-based FL, where individual participants can select a committee to offer SMC services, reducing overhead and maintaining system scalability.

- **Homomorphic Encryption:** Another cryptographic technique currently being applied in the context of federated learning is Homomorphic Encryption (HE). HE is used to solve highly complex mathematical problems. In the federated learning (FL) context, the implementation of HE requires clients to encrypt their local gradients before transferring them to the server. The clients also provide a scheme for the central server to work on the encrypted data. Once the central server performs aggregation on the received data using the proposed scheme, the data is sent back to the clients where it is decrypted to access the final model.

Zhang et al. [28] initially proposed the use of HE for privacy preservation in Bayesian models in vertical FL systems. The proposed methodology was later enhanced in article [29], where an additional component called the coordinator was introduced into the system. The coordinator's role is to specify certain security protocols, including keys and processing functions.

Even though it is claimed to be highly secure, Homomorphic Encryption (HE) comes with a certain amount of communication loss, which holds significant value. Therefore, extensive research has been conducted to optimize communication bottlenecks in HE. To reduce the communication bottleneck in HE, Zhang et al. [28] proposed partitioning the gradients into various batches and performing encryption on these batches instead of encrypting individual gradients. The proposed methodology proved to be highly efficient, taking only a fraction of the time compared to the primitive approach, while also reducing the total volume of cipher text and communication loss incurred.

- **Hybrid Encryption Techniques:** Apart from the techniques discussed previously, hybrid encryption techniques are also available, where the latter are combined to overcome the pitfalls of individual techniques and increase system functionality. Attempts to combine DP and SMC have been presented in articles [30, 31]. In general, such

methodologies allow the local client devices to perform DP-based encryption on their gradients before sending them to the central server. At the central server, the gradients are further encrypted to complete SMC-based secure aggregation. They also proposed that the amount of noise to be added to the gradient could be determined by the client devices themselves. Similar to the combination of SMC and DP, attempts have been made to combine DP with HC, which also provided outstanding encryption results. One such article is presented by Hao et al. [32], where the authors propose adding noise to individual gradients before encryption. To reduce computation cost, a lightweight additive HE was employed, and DP was mainly used to reduce threats of collusion-based attacks.

#### **4.6.2 DECENTRALIZATION-BASED MECHANISMS**

In classical learning systems, the necessity of a central server for performing model aggregation and synchronizing the training procedure is high. However, the reliability and trust factor offered by the central server is questionable. Additionally, even for genuine networks, FL servers tend to be honest but curious, meaning that they always tend to peek into client data without malicious intent. Thus, privacy concerns are always prevalent in such scenarios. To overcome these issues, decentralized FL-based approaches have been introduced as a measure for additional privacy preservation. Blockchain is one of the most widely employed technologies for implementing decentralized FL. With the employment of blockchain and distributed ledger systems, the requirement for a trusted third party can be easily mitigated from FL systems.

One of the initial works on Federated Learning (FL) that employed blockchain for privacy preservation of clients from unauthorized access was presented by Rahman et al. [33]. The article also proposed a framework that supports Homomorphic encryption and Secure Multi-Party Computation (SMC). Majeed et al. [34] proposed the proper integration of a blockchain-based ledger into the FL system to store the results of individual iterations, aiming to keep the system safe from poisoning attacks. Li et al. [35] also made a similar proposal, deploying a consensus-based mechanism to check updated models and aggregation procedures, thus avoiding the presence of adversarial models during training. Decentralized FL applications are not solely limited to blockchain. An article published by [36] proposed a lossless tree-boosting-based system for vertical FL. Wu et al. [37] also proposed a

tree-based mechanism with added Homomorphic Encryption (HE) and SMC features. Various other types of tree architecture could also be considered in the context of decentralized FL.

#### **4.6.3 INTRUSION DETECTION MECHANISMS**

Similar to privacy-preservation mechanisms, intrusion detection is also a highly effective method for preventing potential privacy breaches and threats, thereby improving system security. Traditional intrusion detection methodologies often rely on machine learning to train anomaly detection-based models. Like any machine learning model, intrusion detection systems require substantial amounts of training data to achieve accurate results. Federated Learning (FL) has emerged as a promising approach for working with large volumes of data without actually sharing them, making it suitable for developing intrusion detection systems.

Several articles have proposed the use of FL for intrusion detection. Liu et al. [38] proposed a collaborative intrusion detection system based on decentralized FL. Their model was built on blockchain technology to ensure efficiency and system security. Additionally, several decentralized intrusion detection systems based on blockchain have been proposed, specifically designed for lightweight devices. Mothukuri et al. [39] also proposed intrusion detection systems based on decentralized devices.

### **4.7 GLOBAL MODEL AGGREGATION**

Performing global model aggregation from the received gradients is one of the major phases of Federated Learning (FL) training. The level and efficiency of the aggregation being performed determine the overall efficiency of the system. In this section, we will discuss global model aggregation in relation to three different parameters: aggregation approaches, communication efficiency, and client selection.

#### **4.7.1 AGGREGATION APPROACH**

The methodology used for global model aggregation plays a crucial role in FL. The efficiency of the methodology applied directly influences the overall performance of the system and the generated model. In the current

scenario, multiple models and methodologies have been proposed for global model aggregation. This particular section will focus more on weight-based aggregation approaches rather than privacy-preserving approaches such as SMC-based aggregation, HE-based aggregation, and so on.

Federated averaging (FedAvg) is one of the initial aggregation protocols proposed with the introduction of FL. It is a federated version of the classical stochastic gradient descent algorithm, and the model was proposed by Google [40]. Despite the numerous articles presenting various aggregation technologies, FedAvg remains one of the benchmarks of FL aggregation. The FedAvg methodology works by initially distributing weights to client devices based on the local data they can provide for model training. After local model training, the updated weights are aggregated at the server to generate the global model. Computational time refers to the total time required for generating a model at the client end. Computation time efficiency is also associated with the generation of weights, rather than just the amount of data being processed. An initial approach considering both time efficiency and training data was proposed by Chen et al. [41]. The article proposed taking into consideration both factors in a synchronous manner for global model aggregation. Thus, local models with higher time efficiency could have larger stakes during global model aggregation.

Another methodology for distributing weights to the local client was proposed, which was based on the reputation of individuals. Various approaches were devised for calculating the reputation of a device in a given network. Similarly, a contribution-based weight distribution schema was initially proposed by Wang et al. [42]. They proposed that clients with higher levels of contribution for model generation should be given larger volumes of weights. Additionally, they also proposed that a particular reputation-based threshold value should be set, only surpassing which individual clients could participate in model aggregation. A similar proposal was made by Deng et al. [43] as well. They proposed letting the server device decide the aggregation for client devices based on the ability of the client devices to learn. The ability to perform learning was evaluated from the global model loss.

Even though FedAvg applied to IID data distribution, which was an idealistic scenario, real-life applications mainly worked on non-IID data. However, FedAvg failed to perform well in those instances. Thus, Ma et al. [44] proposed entirely different schemes for the weight distribution in non-IID use cases. One approach distributed weights based on the number of classes of data, and the second one calculated the accuracy of local models raised on validation data to determine the weights. Compared to the initial FedAvg, most of its successors had better generalization capacity.

#### **4.7.2 COMMUNICATION EFFICIENCY**

As federated learning aggregation works in a collaborative manner, co-ordinating the communications between the client devices and the server determines the performance of the system. In ideal situations, a communication-efficient FL system could perform better than another system with similar specifications, as more iterations could be completed in the same duration of time in the former compared to the latter. Several approaches have been employed to ensure communication efficiency in FL systems. Currently, quantization and sparsification are two of the major techniques employed for the same.

Basically, in gradient quantization-based approaches, gradients are compressed to a specific limit to reduce their volume. Kome et al. [45] proposed two different schemas for gradient quantization. The initial one was termed sketched updates, and the proposed model worked by compressing information through quantization, rotation, and sub sampling before gradient transfer. Gradient sparsification is also a highly employed technique for reducing data volume. The top-k sparsification approach is one of the most widely used methods for the same. Liu et al. [46] proposed a gradient compression technique based on top-K selection in their proposal on anomaly detection. In the proposed methodology, gradients with a certain level of importance could be transferred to the server for model aggregation. Similar work was proposed by Sattler et al. [47], where the authors employed sparse ternary compression based on top-k gradient sparsification technology. They worked on both uplink and downlink compression to further optimize communication. In addition to quantization and sparsification methods, certain other methodologies were also employed to reduce gradient transfer volume. Yang et al. [48] employed signal superimposition techniques for multi-user channels as a method for improving communication efficiency. Similarly, methods such as dropout, maximum mean discrepancy, and so on were also proposed as measures for developing communication efficiency techniques.

#### **4.7.3 CLIENT SELECTION**

Client selection usually implies the selection of clients to participate in FL training. Client selection is a crucial step in FL, as the number of clients participating in training, the nature of the clients, the quality of the data being provided, and other factors directly determine the efficiency of the generated

model and the required training time. Increasing the number of participants in FL does not necessarily result in higher performance efficiency. It is important to prioritize selecting a smaller number of clients who can contribute higher quality data and thereby improve trained models. Thus, client selection plays a significant role in determining the quality of the final model. In the current scenario, various approaches have been employed to determine the number of clients participating in training. One such approach is the probability-based mechanism, where the selection process relies on probability. This approach is simple and easy to implement for client selection. For example, Reisizadeh et al. [49] randomly selected clients from a set of total clients to minimize aggregation overheads. However, in reality, random selection does not offer an optimized system performance.

Another approach employed for client selection is the client capacity-based mechanism. In the proposal mechanism, clients are selected based on their resources and computational capabilities for generating federated learning models. The idea behind this approach is that clients with higher computational resources can generate better models in less time. Initial articles on this topic were proposed by Abdul Rahman et al. [50]. They suggested considering client devices' CPU and memory usage patterns as factors for determining their eligibility to participate in the training. Several other articles have also proposed alternative methodologies, such as efficiency-based approaches, due to the major issue of accurately estimating individual client devices' contributions before aggregation. Especially when estimating the contribution in a specific iteration, it is highly likely that the generated value will not be accurate. Thus, other methodologies were devised as alternatives to the existing ones.

## **4.8 FUTURE SCOPE**

In subsections, we will be discussing the challenges and future scope of integration of FL with SDN.

### **4.8.1 CLIENT SELECTION**

In collaborative training-based systems, it is always advisable to encourage a higher number of client devices to contribute their data for training and participate in training, as higher quality data will always yield better models with higher performance. In the current scenario, most of the client selection

methodologies are not as effective in encouraging client participation, and better technologies need to be devised for client selection.

When attempting to integrate FL with SDN, SDN controllers usually act as participants. When choosing an SDN controller as a client device, several factors need to be taken into consideration, such as their communication protocols, computation capability, data storage capacity, and so on. All these aforementioned parameters have a crucial role in determining the system performance and thus provide ample scope for future research.

#### **4.8.2 SCALABILITY ISSUES**

As FL and SDN are both distributed architecture-based applications, there are always issues related to system scalability. However, when it comes to FL, a major hurdle for scaling the system is the high communication costs. Thus, to minimize computation-based constraints, various technologies such as edge computing and fog computing could be introduced into FL systems. When it comes to scalability related to systems where FL and SDN are being integrated, it is always advised to introduce the distributed control plane with layered controller architecture. Data compression techniques currently being employed in FL also have scope in the new technology as in SDN the open nature of the communication protocols would help in the same.

### **4.9 CONCLUSION**

Through this article, we have presented some of the major technological advancements, challenges, and prospective solutions for the integration of Federated Learning (FL) into Software-Defined Networking (SDN) frameworks. We have also attempted to perform a robust analysis of the state-of-the-art literature available on the same. After presenting the fundamentals and major structural components of SDN and FL frameworks, we have stressed the necessity of integrating FL and SDN as an efficient method to manage client devices, data management, optimized training, and effective client collaborations. Driven by these factors, the study delved into summarizing various incentive mechanisms in such systems, along with privacy analysis and aggregation methods. As FL is a distributed learning paradigm that relies on data contribution from various clients to generate local models, followed by gradient sharing and aggregation to generate a final model, we also discussed the scope of a strong incentive mechanism. Other factors

determining the incentive distribution level, such as client reputation and their contribution in terms of data usefulness and other resources in training, were also discussed. The applicability of FL as a methodology for privacy preservation in the case of highly sensitive data was highlighted, along with a short comparison of centralized Machine Learning (ML) and FL on various malicious attack scenarios. Since SDN also faces certain security issues, designing anomaly detection methodologies based on FL-integrated SDN has emerged as a high scope research topic. As with any distributed computing framework, scalability is a serious issue in SDN as well. Thus, the study also presented the applicability of FL on SDN under various instances as an effective deterrent to scalability issues.

## KEYWORDS

- **data compression**
- **federated learning**
- **privacy preservation**
- **resource management**
- **scalability issues**
- **software-defined networks**
- **sparsification technology**

## REFERENCES

1. Sarikaya, Y., & Ozgur, E., (2019). Motivating workers in federated learning: A Stackelberg game perspective. *IEEE Networking Letters*, 2(1), 23–27.
2. Feng, S., Dusit, N., Ping, W., Dong In, K., & Ying-Chang, L., (2019). Joint service pricing and cooperative relay communication for federated learning. In: *2019 International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (Green Com) and IEEE Cyber, Physical and Social Computing (CPS Com) and IEEE Smart Data (Smart Data)* (pp. 815–820). IEEE.
3. Sun, W., Ning, X., Lu, W., Haibin, Z., & Yan, Z., (2020). Dynamic digital twin and federated learning with incentives for air-ground networks. *IEEE Transactions on Network Science and Engineering*.
4. Zhan, Y., Peng, L., Zhihao, Q., Deze, Z., & Song, G., (2020). A learning-based incentive mechanism for federated learning. *IEEE Internet of Things Journal*, 7(7), 6360–6368.

5. Le Tra, H. T., Nguyen, H. T., Yan, K. T., Minh, N. H. N., Shashi, R. P., Zhu, H., & Choong, S. H., (2021). An incentive mechanism for federated learning in wireless cellular networks: An auction approach. *IEEE Transactions on Wireless Communications*, 20(8), 4874–4887.
6. Yu, H., Zelei, L., Yang, L., Tianjian, C., Mingshu, C., Xi, W., Dusit, N., & Qiang, Y., (2020). A sustainable incentive scheme for federated learning. *IEEE Intelligent Systems*, 35(4), 58–69.
7. Zeng, R., Shixun, Z., Jiaqi, W., & Xiaowen, C., (2020). FMore: An incentive scheme of multi-dimensional auction for federated learning in MEC. In: (2020). *IEEE 40<sup>th</sup> International Conference on Distributed Computing Systems (ICDCS)*(pp. 278–288). IEEE.
8. Tang, M., & Wong, V. W., (2021). An incentive mechanism for cross-silo federated learning: A public goods perspective. In: *IEEE INFOCOM 2021—IEEE Conference on Computer Communications* (pp. 1–10). IEEE.
9. Pan, Q., Jun, W., Ali, K. B., Jianhua, L., Wu, Y., & Al-Otaibi, Y. D., (2021). Joint protection of energy security and information privacy for energy harvesting: An incentive federated learning approach. *IEEE Transactions on Industrial Informatics*, 18(5), 3473–3483.
10. Song, T., Yongxin, T., & Shuyue, W., (2019). Profit allocation for federated learning. In: *2019 IEEE International Conference on Big Data (Big Data)* (pp. 2577–2586). IEEE.
11. Sim, R. H. L., Yehong, Z., Mun, C. C., & Bryan, K. H. L., (2020). Collaborative machine learning with incentive-aware model rewards. In: *International Conference on Machine Learning* (pp. 8927–8936). PMLR.
12. Gupta, D., Kayode, O., Bhatt, S., Gupta, M., & Tosun, A. S., (2020). Learner’s dilemma: IoT devices training strategies in collaborative deep learning. In: *2020 IEEE 6<sup>th</sup> World Forum on Internet of Things (WF-IoT)* (pp. 1–6). IEEE.
13. Lim, W. Y. B., Zehui, X., Chunyan, M., Dusit, N., Qiang, Y., Cyril, L., & Vincent, P. H., (2020). Hierarchical incentive mechanism design for federated machine learning in mobile networks. *IEEE Internet of Things Journal*, 7(10), 9575–9588.
14. Cong, M., Han, Y., Xi, W., Jiabao, Q., Yang, L., & Siu, M. Y., (2020). *A VCG-Based Fair Incentive Mechanism for Federated Learning*. arXiv preprint arXiv:2008.06680.
15. Wang, G., Charlie, X. D., & Ziye, Z., (2019). Measure the contribution of participants in federated learning. In: *2019 IEEE International Conference on Big Data (Big Data)* (pp. 2597–2604). IEEE.
16. Zhao, Y., Jun, Z., Linshan, J., Rui, T., Dusit, N., Zengxiang, L., Lingjuan, L., & Yingbo, L., (2020). Privacy-preserving blockchain-based federated learning for IoT devices. *IEEE Internet of Things Journal*, 8(3), 1817–1829.
17. Ur Rehman, M. H., Khaled, S., Ernesto, D., & Davor, S., (2020). Towards blockchain-based reputation-aware federated learning. In: *IEEE INFOCOM 2020—IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)* (pp. 183–188). IEEE.
18. Xiong, P., Tian-Qing, Z., & Xiao-Feng, W., (2014). A survey on differential privacy and applications. *Jisuanji Xuebao/Chinese Journal of Computers*, 37(1), 101–122.
19. Wei, K., Jun, L., Ming, D., Chuan, M., Howard, H. Y., Farhad, F., Shi, J., et al., (2020). Federated learning with differential privacy: Algorithms and performance analysis. *IEEE Transactions on Information Forensics and Security*, 15, 3454–3469.
20. Geyer, R. C., Tassilo, K., & Moin, N., (2017). *Differentially Private Federated Learning: A Client Level Perspective*. arXiv preprint arXiv:1712.07557.

21. Triastcyn, A., & Boi, F., (2019). Federated learning with Bayesian differential privacy. In: *2019 IEEE International Conference on Big Data (Big Data)* (pp. 2587–2596). IEEE.
22. Cao, H., Shubo, L., Renfang, Z., & Xingxing, X., (2020). IFed: A novel federated learning framework for local differential privacy in power internet of things. *International Journal of Distributed Sensor Networks*, 16(5), 1550147720919698.
23. Liu, R., Yang, C., Masatoshi, Y., & Hong, C., (2020). FedSel: Federated SGD under local differential privacy with top-k dimension selection. In: *International Conference on Database Systems for Advanced Applications* (pp. 485–501). Springer, Cham.
24. Zhu, H., (2020). *On the Relationship Between (Secure) Multi-Party Computation and (Secure) Federated Learning*. arXiv preprint arXiv:2008.02609.
25. Bonawitz, K., Vladimir, I., Ben, K., Antonio, M., McMahan, H. B., Sarvar, P., Daniel, R., et al., (2017). Practical secure aggregation for privacy-preserving machine learning. In: *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security* (pp. 1175–1191).
26. Kanagavelu, R., Zengxiang, L., Juniarto, S., Yechao, Y., Feng, Y., Rick, S. M. G., Mervyn, C., et al., (2020). Two-phase multi-party computation enabled privacy-preserving federated learning. In: *2020 20<sup>th</sup> IEEE/ACM International Symposium on Cluster, Cloud and Internet Computing (CCGRID)* (pp. 410–419). IEEE.
27. Zhang, X., Anmin, F., Huaqun, W., Chunyi, Z., & Zhenzhu, C., (2020). A privacy-preserving and verifiable federated learning scheme. In: *ICC 2020–2020 IEEE International Conference on Communications (ICC)* (pp. 1–6). IEEE.
28. Zhang, J., Bing, C., Shui, Y., & Hai, D., (2019). PEFL: A privacy-enhanced federated learning scheme for big data analytics. In: *2019 IEEE Global Communications Conference (GLOBECOM)* (pp. 1–6). IEEE.
29. Truex, S., Nathalie, B., Ali, A., Thomas, S., Heiko, L., Rui, Z., & Yi, Z., (2019). A hybrid approach to privacy-preserving federated learning. In: *Proceedings of the 12<sup>th</sup> ACM Workshop on Artificial Intelligence and Security* (pp. 1–11).
30. Mou, W., Chunlei, F., Yan, L., & Chunqiang, H., (2021). A verifiable federated learning scheme based on secure multi-party computation. In: *International Conference on Wireless Algorithms, Systems, and Applications* (pp. 198–209). Springer, Cham.
31. Küpper, A., Younghlee, P., Peter, R., Stefan, S., & Jie, X., (2019). *Proceedings of the 2019 IEEE International Conference on Decentralized Applications and Infrastructures (DAPPCON 2019)*. IEEE.
32. Rahman, M. A., Shamim, H. M., Mohammad, S. I., Nabil, A. A., & Ghulam, M., (2020). Secure and provenance enhanced Internet of health things framework: A blockchain managed federated learning approach. *IEEE Access*, 8, 205071–205087.
33. Majeed, U., & Choong, S. H., (2019). FLchain: Federated learning via MEC-enabled blockchain network. In: *2019 20<sup>th</sup> Asia-Pacific Network Operations and Management Symposium (APNOMS)* (pp. 1–4). IEEE.
34. Li, Y., Chuan, C., Nan, L., Huawei, H., Zibin, Z., & Qiang, Y., (2020). A blockchain-based decentralized federated learning framework with committee consensus. *IEEE Network*, 35(1), 234–241.
35. Cheng, K., Tao, F., Yilun, J., Yang, L., Tianjian, C., Dimitrios, P., & Qiang, Y., (2021). Secure Boost: A lossless federated learning framework. *IEEE Intelligent Systems*, 36(6), 87–98.
36. Wu, Y., Shaofeng, C., Xiaokui, X., Gang, C., & Beng, C. O., (2020). *Privacy-Preserving Vertical Federated Learning for Tree-Based Models*. arXiv preprint arXiv:2008.06170.

37. Liu, H., Shuapeng, Z., Pengfei, Z., Xinjiang, Z., Xuebin, S., Geguang, P., & Yan, Z., (2021). Blockchain and federated learning for collaborative intrusion detection in vehicular edge computing. *IEEE Transactions on Vehicular Technology*, 70(6), 6073–6084.
38. Mothukuri, V., Prachi, K., Reza, M. P., Seyedamin, P., Ali, D., & Gautam, S., (2021). Federated-learning-based anomaly detection for IoT security attacks. *IEEE Internet of Things Journal*, 9(4), 2545–2554.
39. McMahan, B., Eider, M., Daniel, R., Seth, H., & Blaise, A. Y. A., (2017). Communication-efficient learning of deep networks from decentralized data. In: *Artificial Intelligence and Statistics* (pp. 1273–1282). PMLR.
40. Chen, Y., Xiaoyan, S., & Yaochu, J., (2019). Communication-efficient federated deep learning with layer-wise asynchronous model update and temporally weighted aggregation. *IEEE Transactions on Neural Networks and Learning Systems*, 31(10), 4229–4238.
41. Wang, Y., & Burak, K., (2021). Reputation-enabled federated learning model aggregation in mobile platforms. In: *ICC 2021-IEEE International Conference on Communications* (pp. 1–6). IEEE.
42. Deng, Y., Feng, L., Ju, R., Yi-Chao, C., Peng, Y., Yuezhi, Z., & Yaoxue, Z., (2021). Fair: Quality-aware federated learning with precise user incentive and model aggregation. In: *IEEE INFOCOM 2021-IEEE Conference on Computer Communications* (pp. 1–10). IEEE.
43. Ma, Z., Mengying, Z., Xiaojun, C., & Zhiping, J., (2021). Fast-convergent federated learning with class-weighted aggregation. *Journal of Systems Architecture*, 117, 102125.
44. Konečný, J., McMahan, H. B., Felix, X. Y., Peter, R., Ananda, T. S., & Dave, B., (2016). *Federated Learning: Strategies for Improving Communication Efficiency*. arXiv preprint arXiv:1610.05492.
45. Liu, Y., Sahil, G., Jiangtian, N., Yang, Z., Zehui, X., Jiawen, K., & Shamim, H. M., (2020). Deep anomaly detection for time-series data in industrial IoT: A communication-efficient on-device federated learning approach. *IEEE Internet of Things Journal*, 8(8), 6348–6358.
46. Sattler, F., Simon, W., Klaus-Robert, M., & Wojciech, S., (2019). Robust and communication-efficient federated learning from non-IID data. *IEEE Transactions on Neural Networks and Learning Systems*, 31(9), 3400–3413.
47. Yang, K., Tao, J., Yuanming, S., & Zhi, D., (2020). Federated learning via over-the-air computation. *IEEE Transactions on Wireless Communications*, 19(3), 2022–2035.
48. Reisizadeh, A., Mokhtari, A., Hassani, H., Jadbabaie, A., & Pedarsani, R., (2020). Fedpaq: A communication-efficient federated learning method with periodic averaging and quantization. In: *International Conference on Artificial Intelligence and Statistics* (pp. 2021–2031). PMLR.
49. AbdulRahman, S., Hanine, T., Azzam, M., & Chamseddine, T., (2020). FedMCCS: Multicriteria client selection model for optimal IoT federated learning. *IEEE Internet of Things Journal*, 8(6), 4723–4735.



Taylor & Francis

Taylor & Francis Group

<http://taylorandfrancis.com>

## CHAPTER 5

---

# Federated Learning in the Internet of Medical Things

S. SABAPATHI,<sup>1</sup> N. VIJAYALASKHMI,<sup>2</sup> and S. SINDHU<sup>2</sup>

<sup>1</sup>*Saveetha College of Liberal Arts and Science, Chennai, Tamil Nadu, India*

<sup>2</sup>*SRM Institute of Science and Technology, Chennai, Tamil Nadu, India*

---

### ABSTRACT

The Internet of Things (IoT) is bringing intelligent services and applications powered by artificial intelligence (AI) into many aspects of our daily lives. The high scalability of modern IoT networks and growing privacy concerns make AI techniques impractical in realistic application scenarios due to the requirements for centralized data collection and processing. The concept of federated learning (FL) is emerging as a method for enabling intelligent IoT applications with the ability to train AI in distributed devices without sharing data. From a comprehensive overview of the recent developments in FL and IoT to a discussion of their integration, we provide a comprehensive survey of FL's emerging applications in IoT networks. FL is especially explored and analyzed for its potential to enable a wide range of IoT services, such as sharing IoT data, offloading and caching data, detecting attacks, localizing them, and monitoring crowds using mobile phones. In our next section, we provide an extensive investigation of the application of FL to a variety of key IoT applications, including smart healthcare, smart transportation, Unmanned Aerial Vehicles (UAVs), smart cities, and smart manufacturing. In addition to highlighting FL-IoT services and applications, this review highlights important lessons learned. This survey concludes by highlighting challenges and possible future research directions in this rapidly growing field.

## 5.1 INTRODUCTION

A learning technique that enables the training of shared models using a collective wealth of data without central storage by users. This method, known as federated learning, involves a loose network of participating devices (referred to as clients) under the guidance of a central server. The increasing connectivity of the internet leads to a rise in data generation and the demand for intelligent applications to enhance user experiences. Additionally, the report predicts a nearly threefold increase in the number of IoT devices worldwide, from 9.7 billion in 2020 to over 29 billion by 2030.

A relatively new learning method called Federated Learning avoids centralized data collection and model training. Data is gathered from various sources, including mobile devices, and stored in a centralized location, such as a data center, in traditional machine learning pipelines. In centralized learning, a single machine learning model is trained using the entire set of data once it is available in the center. However, in federated learning, multiple machine learning models are trained on portable devices (referred to as clients), and the outcomes of these models are combined into a single model stored on the server. This way, the devices themselves train the model using the ground truth data, and only the trained model is shared with the server. This approach ensures that user data remains private while being utilized to create machine/deep learning (DL) models. One of the significant advantages of this decentralized concept is enhanced data security and protection. Fields like defense and healthcare are considered to benefit greatly from this concept. Some of the most commonly used applications of federated learning in our daily lives include the Google keyboard on Android devices and the Netflix recommendation engine [1].

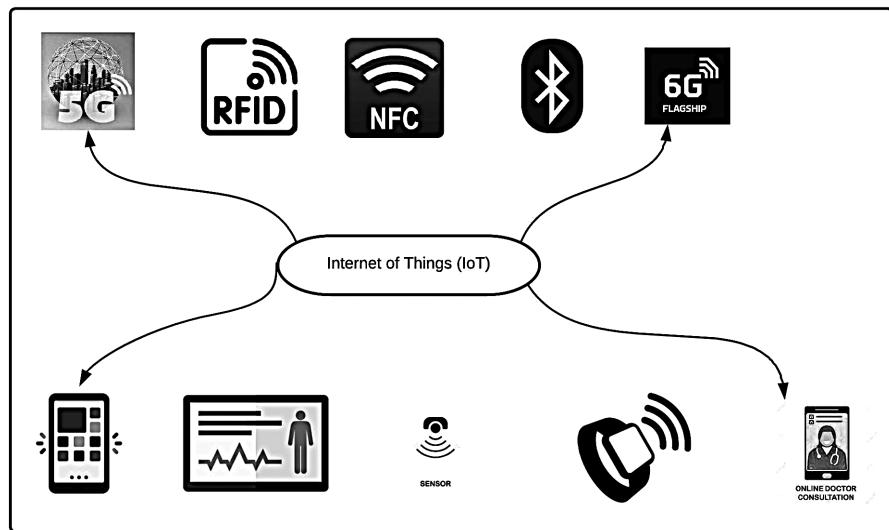
A network of medical devices that communicate with computer programmers and the internet primarily for the purpose of obtaining medical data is referred to as the “Internet of Medical Things,” or “IoMT.” IoMT devices are a subset of Internet of Things (IoT) devices, which include all web-enabled equipment such as smart automobiles and household appliances. These connected devices enable online activity tracking of their usage from any location. The IoMT, which allows machine-to-machine interactions and real-time intervention techniques, will significantly alter healthcare delivery, affordability, and dependability in the near future [2]. Additionally, greater patient input into decision-making will increase adherence to healthcare services. Future healthcare systems will be based on IoMTs, where every medical device will be connected to the Internet and supervised by medical professionals. As it develops, this offers faster and less expensive healthcare.

IoMT examples include tracking patient medication orders and the location of patients admitted to hospitals, monitoring patients with chronic or long-term conditions remotely, and patients' wearable mHealth devices, which can send information to caregivers [3]. Medical devices that can be converted to or used with IoMT technology include hospital beds equipped with sensors that monitor patients' vital signs and infusion pumps that connect to analytics dashboards. IoMT supports individualized healthcare and a high quality of life by offering devices that are tailored to physiological needs. Affordable medical devices and a connected health ecosystem will soon be a reality as a result of ongoing research in the sensor, network, cloud, mobility, and big data domains. A network of interconnected devices that continuously sense vital data is referred to as the IoMT, a healthcare application of the Internet of Things technology. It is now possible to continuously monitor a person's various health factors and foresee the onset of any diseased conditions thanks to advanced technological tools. Maintaining the caliber and standard of healthcare services offered would assist in reducing the patient load and burden on the currently available healthcare infrastructure facilities. Big data, artificial intelligence (AI), and machine learning tools can be used to accurately predict a patient's diseased condition even when the most skilled and experienced clinicians are unable to recognize and diagnose it. With the aid of various statistical techniques and scientific analysis using data accumulated in cloud servers from IoT devices, these techniques use test data to train and develop learning algorithms that can help to accurately predict any patient abnormality.

IoT systems are composed of sensors and devices that are interconnected through a network of cloud ecosystems using high-speed connectivity. The extensive storage space provided by cloud services receives the raw data collected from these devices/sensors. To gain further insights, this data is further processed and analyzed. This requires the use of additional programs, devices, and tools to enhance the visualization, analysis, processing, and management of the data [4]. Various wireless technologies, such as Radio Frequency Identification Tag (RFID Tag), Near Field Communications (NFC), Bluetooth, Long Term Evolution (LTE), and 5G/6G (and beyond), are interconnected with a range of devices including smartphones, monitoring devices, sensors, smart wearables, and other medical devices as depicted in Figure 5.1. Due to their advantages in high bandwidth and ultra-low latency, 5G/6G and beyond are currently widely utilized in IoMT.

The Internet of Things (IoT) will significantly transform not only the healthcare industry but also other sectors such as manufacturing, construction, and power distribution. In the field of healthcare, the Internet

of Medical Things (IoMT) connects various devices in real-time to collect vital data for healthcare applications. This advancement in technology has led to improved real-time health monitoring solutions and increased patient involvement in decision-making processes, resulting in a growing trend of human-machine interaction. With the help of IoT, real-time health monitoring, data recording, and health record maintenance are made possible, enabling data-driven decision-making. Patients can now manage their health in a more personalized manner.



**FIGURE 5.1** A general architecture for the internet of things.

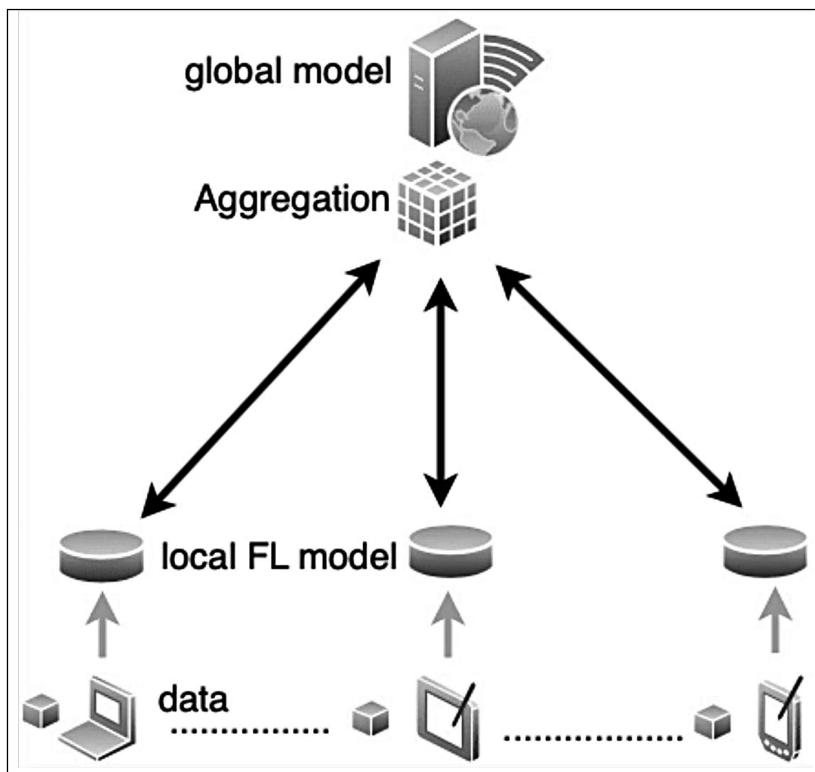
## 5.2 FEDERATED LEARNING APPLICATIONS

When training data is distributed at the edge for applications, Federated Learning methods are crucial for supporting privacy-conscious applications. Applications for federated learning include predicting health events like the risk of a heart attack from wearable technology, learning mood, semantic location, cell phone activity, and adapting to pedestrian behavior in self-driving cars [5]. Several well-known federated learning application types include:

1. **Smartphones:** Applications such as next-word prediction, facial recognition, and speech recognition are powered by statistical models that are used to collectively learn user behavior across a large pool of

mobile phones. Users may choose not to share data in order to protect their privacy, reduce phone bandwidth, or conserve battery consumption. Federated learning enables smartphones to have predictive capabilities without disclosing personal information or impacting the user experience.

2. **Organization:** In the context of federated learning, an entire organization or institution can also be referred to as a “device.” For applications in predictive medicine, hospitals are one organization with a lot of patient data. However, hospitals are required to adhere to strict privacy laws and may also be subject to administrative, legal, or ethical limitations that call for the local storage of data. Federated learning is a solution for these types of applications because it lightens the load on the network and makes private learning possible across various devices and organizations.



**FIGURE 5.2** A common architecture for federated learning applications.

Source: Reprinted from Ref. [15]

### **5.2.1 RATIONALE FOR CHOOSING FEDERATED LEARNING**

Shifting the classical learning procedures into a distributed or federated architecture depends on the system's requirements and the benefits they aim to achieve. In the context of this study, we will present the rationale for adopting the FL procedure in both IoT systems and IoMT systems.

#### *5.2.1.1 INTERNET OF THINGS*

Smart homes, self-driving cars, and wearable Internet of Things devices all use sensors to gather and respond to incoming data in real-time. For instance, for fleets of autonomous vehicles to operate safely, it may be necessary to use current models of traffic, construction sites, or pedestrian behavior [6]. However, due to privacy issues and the limited connectivity of each device, creating aggregate models in these scenarios can be difficult. We can train models that quickly adapt to changes in these systems while still protecting user privacy, thanks to federated learning techniques.

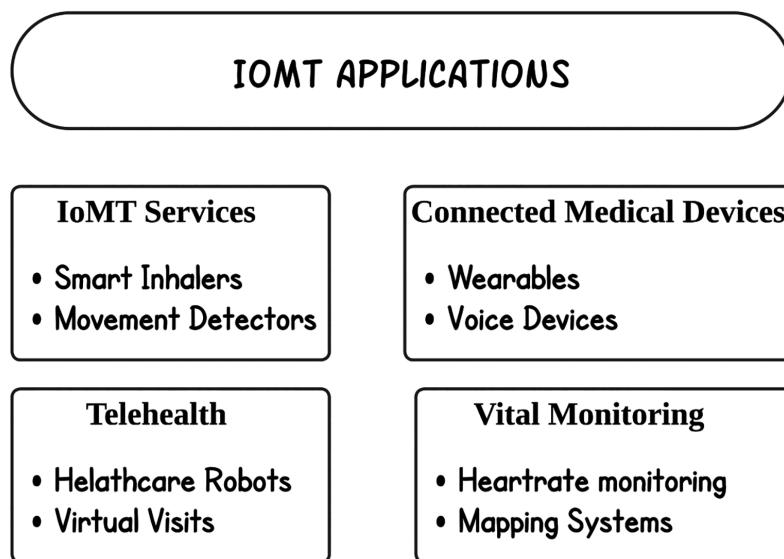
#### *5.2.1.2 INTERNET OF MEDICAL THINGS*

Even in the case of IoMT, integration federated learning offers several benefits, some of which include:

- **Data Island:** With the increase of IoMT and wearable devices, data growth is occurring at different individual units. Due to the difficulties in exchanging data, integrating enormous amounts of data is neither practical nor advised. Ailed data can thus result in the creation of disparate, isolated data that doesn't have real value for healthcare unless it's used to train AI models.
- **Privacy:** Data anonymization is achieved by removing patient identities from electronic medical records or medical images. However, it has been discovered that reconstruction techniques can still extract certain identifying information from anonymous data. Therefore, simply removing a person's name, ID number, or date of birth from medical records may not be enough to guarantee privacy.

To obtain real-time data for patient monitoring, enhancements such as sensors, signal conditioners, and communication modems can be incorporated into existing medical equipment. Examples of IoMT devices include

smart wearables, home medical equipment, point-of-care kits, and mobile health applications, all of which enable communication with remote medical professionals. These innovations and applications in the field of healthcare improve patient care, alleviate the workload of medical professionals, and allow for treatment to be administered at home or in non-hospital settings. The benefits of IoMT extend beyond disease prevention and health monitoring, promoting fitness and facilitating remote communication during emergencies. Figure 5.3 illustrates some of the typical use cases of IoMT systems, which have potential for integration with Federated Learning.



**FIGURE 5.3** Applications of IoMT.

### 5.3 FEDERATED LEARNING FOR IOMT DEVICES

IoMT devices with the ability to sense and transmit health updates of an individual are extensively used to gather healthcare data. Some of the common types of devices are as follows:

#### 5.3.1 CONNECTED HEALTHCARE

Connected healthcare is a term used to denote a system where various healthcare technologies and devices are connected, forming a stable healthcare management system that works on various aspects. It can be defined as a

convergence point for healthcare technology and telecommunication devices. In the current scenario, it is widely employed in hospitals, smart homes, and other settings, offering users a complete healthcare experience with remote care and access to vital health-related information [7]. Connected healthcare scenarios can be further classified into the following:

#### *5.3.1.1 WEARABLES*

In the past 10 years, fitness and activity trackers have become popular as lifestyle goods. We will observe a greater variety of wearables on the market as IoMT becomes more widely available. Patient monitors, a crucial piece of equipment for treating patients in remote locations, have been one of the most significant innovations. Other gadgets include chest straps with integrated ECG and heart rate sensors, as well as smart belts for elderly patients.

#### *5.3.1.2 VOCAL DEVICES*

Another trend that IoMT devices could benefit from is smart speakers. By developing applications for smart speakers, patients will have access to a personal “family doctor” who can perform basic diagnostic and assessment tasks. These voice-activated tools can assist medical professionals in reaching a larger number of patients and expanding their reach.

#### *5.3.1.3 TELEHEALTH*

Telehealth is typically defined as a system that utilizes virtual data, such as digital documents and information, along with various communication technologies to access healthcare services remotely and manage personalized healthcare routines. The communication technologies used can include computing devices, mobile devices, and more. Some common telehealth applications include chatbots and virtual visit systems.

- **Healthcare Chatbot:** Even though chatbots were initially developed as a mode of interaction in the late 1990s, in the current scenario, their use goes much further than that. When it comes to getting answers, chatbots have shown to be more popular than more traditional web forms—a benefit that the healthcare sector should take advantage of. A chatbot can be directed to automate the scheduling of

appointments, the recording of patient symptoms, and even the filling of prescriptions.

- **Virtual Visits:** Virtual appointments offer dedicated software with built-in capabilities to seamlessly relay information to doctors in real-time. They are the next best thing for patients who live remotely or are stranded, providing counseling knowledge.

#### 5.3.1.4 PATIENT VITAL MONITORING

Monitoring various vitals in the human body is also a major leap for IoMT-enabled healthcare. Some of the commonly monitored vitals are heart rate, oxygen levels, and so on. During the COVID-19 pandemic, IoT-enabled oxygen monitoring devices gained wide popularity. Similarly, the importance of cardiac monitoring awareness in patients has quickly increased due to the global rise in the prevalence of myocardial disease. The widespread use of certain smart watches that have an integrated ECG feature has made it familiar for a large group of people to understand the possibilities of identifying potential heart disease through such wearable devices. It is critical to democratize such technology that alerts people to potential heart issues before they develop into full-blown heart attacks, given this alarming trend. Only a few technically savvy people still use it, making it a niche phenomenon. By creating lightweight, wearable heart rate monitors that are affordable, accessible, and interoperable, the IoMT sector has the potential to disrupt this market. Such tools might become indispensable for upcoming medical professionals.

#### 5.3.1.5 NON-IMAGING DIAGNOSTICS

Even though image-based diagnostics take up a dominant portion of diagnostic tasks, IoT-based diagnostic procedures are equally effective. While image-based diagnostics occur during the clinical phase, signal-based diagnostics can be performed in any environment, making it highly effective for preventive or early disease detection [8]. These systems typically consist of IoMT devices that continuously monitor specific body functions for diagnostics. Some common types include:

- **Mapping Systems:** Electrophysiological (EP) mapping and imaging procedures have historically been carried out in specialized hospitals with advanced equipment. Without using invasive imaging methods,

it was impossible to see inside the body. You don't need to restrict this with IoMT. There are instances of doctors using medical imaging software for non-diagnostic imaging and borrowing techniques from neuroimaging.

- **Cloud-Connected ECG Solution:** The medical community has transitioned from on-site image storage and communication systems (PACS) to the cloud. This shift provides greater versatility to medical professionals, addressing the information overload that has affected radiology departments worldwide. Another advancing solution is the relocation of HIS-based PACS.

### **5.3.2 REMOTE ASSISTED LIVING**

Healthcare automation gathers and processes data specific to patients, and assesses patient care by comparing the most recent data to earlier records. The service provider can save money by transferring routing, monitoring, and field administration responsibilities to their IoMT machines rather than paying to add more resources and use more infrastructure. The productivity of medical resources has increased as a result of remote monitoring [9]. To ensure security, the commercialized “Body Guardian Remote Monitoring System” heart monitoring system separates identification information from observation data. To further ensure the dependability of our solution, we transmit and store sensitive data using cryptographic protocols.

Healthcare professionals can now treat more patients without in-person encounters thanks to telemedicine. And now that its worth has been established, it will remain. The next generation of telemedicine technology offers much more than the basic video conferencing that many healthcare providers have used to introduce the concept. Clinicians use natural language processing to record consultations automatically. Experts provide remote intervention for emergency measures. Additionally, patients receive advanced care wherever they are.

#### **5.3.2.1 BENEFITS OF TELEMEDICINE**

Through improved access to clinicians and specialists, improved patient communication, and the ability for both patients and providers to avoid high-risk environments, telemedicine is enabling a new standard of care.

- **Improved Access to Healthcare:** It is easier to schedule and keep appointments with telemedicine because it extends access to rural and underserved communities. It is also simpler for those who are unable to drive or have disabilities to access the care they require. Additionally, patients in remote medical facilities can access well-known specialists like neurologists.
- **High-Quality of Services Delivered:** Data collection and frequent patient monitoring are made possible by telemedicine technology. This could encourage better self-care habits and assist with diagnosis. Telemedicine systems with AI integration automatically analyze patient data to assist providers in responding to new developments.
- **Clinical Efficiencies:** More patients could be served by providers thanks to telemedicine technology. Between appointments, there is no need to clean the examination room. Clinics are smaller and visits are shorter. Emergency telemedicine accessibility can reduce ER visits. Integration with electronic health records (EHRs) enables healthcare professionals to conduct thorough assessments without having to flip through paper records while quickly comparing test results and reviewing patient history. Additionally, it lowers travel expenses for patients and professionals in high demand.
- **Safe Environment for Service Providers and Beneficiaries:** Telemedicine promotes a healthier environment for everyone. Doctors can consult with sick patients without risking the spread of illness. Immuno-compromised patients can receive routine examinations without having to leave their homes. Providers can give advice and monitor patient progress while better protecting themselves from infections.

#### 5.3.2.2 IOT AND AI IN TELEMEDICINE

Ultra-fast connectivity in the Internet of Things (IoT) era enables the connection of numerous medical devices as well as devices to servers and clouds. As a result, telemedicine technology can utilize real-time data to provide more effective remote care. Patients can monitor their blood pressure, temperature, and heart rate at home using wearables and other medical devices, and then send the results to their doctor for review. Healthcare providers can also add patients' notes, prescriptions, and other information for easy access by pharmacists and other professionals on-site.

The built-in wearable monitors the patient's vital signs throughout the day and uploads the information to the cloud for easy ongoing analysis by both the patient and the caregiver. With this level of monitoring, patients with chronic illnesses may be able to better manage their health and possibly avoid visits to the emergency room and urgent care centers. If you have any inquiries, you can make a telemedicine appointment [10]. Providers have access to ongoing metrics and can offer suggestions. Self-service kiosks in a variety of settings (clinics, pharmacies, or public places) provide clients with another way to communicate with their medical professionals. At kiosks, patients can also schedule appointments and make payments. EEG, ECG, and other readings can be taken by emergency personnel and transmitted to hospital staff on the go using telemedicine technology. The staff can be better prepared for the patient's arrival by receiving immediate treatment advice from experts. Emergencies often require quick action, especially in the case of heart attacks and strokes.

AI opens up new opportunities for telemedicine as well. For instance, AI can dynamically modify questions based on answers to provide prompts that ease the collection of patient history during telemedicine visits. AI algorithms are also particularly helpful for melanoma diagnosis. Based on personal monitoring data, other AI-based tools can suggest routine checkups or offer personalized medication reminders.

### **5.3.3 WELLNESS AND PREVENTIVE CARE (LIFESTYLE ASSESSMENT)**

IoMT-enabled gadgets have made health supervision simpler with monitoring systems for diet, physical activity, and quality of life. Wearable technology, implantable chips, and embedded systems in biomedical devices are examples of cutting-edge devices that continuously monitor patient activity and the corresponding essential changes. Users can locally examine and link a range of crucial events with health issues thanks to smart devices' powerful sensors, converters, and firmware. Due to their remote networking capabilities, these gadgets also provide expert support in emergencies at any remote location.

- **Improved Healthcare Quality:** With easy access to patient healthcare data through simultaneous reporting and monitoring via linked devices, healthcare providers will be better able to provide evidence-based therapies and customize treatments for patients. As a result, the treatment cycle will be shortened, and more patients will be served.

- **Optimized Medical Costs and Errors:** IoT makes it possible to monitor health parameters using diagnostic tools and mobile health applications, as well as report in real-time to doctors via linked equipment. Medical professionals can reduce medical errors by utilizing this accurate data collection to inform their decisions. As a result, there will be a decrease in unwarranted hospital admissions, stays, and visits, as well as a better use of resources and a decrease in healthcare expenses.
- **Healthcare Accessibility in Rural Areas:** The pervasive IoT infrastructure can improve access to high-quality healthcare in rural areas by enabling patients in remote communities to consult with doctors in urban specialty hospitals from the comfort of their homes and by enabling remote health monitoring. As a result, IoT has the potential to transform healthcare in India thanks to its patient-centric philosophy and wide-ranging advantages. The enabling infrastructure, including internet dependability and accessibility, the cost of IoT-enabled devices, a strong IT framework, privacy and data security, intelligent data utilization, and a smooth user experience, needs to be progressively reinforced for it to have the best effects.

The Indian healthcare landscape is evolving as a result of the ease of adoption and implementation of IoT. In addition to supporting population wellness and disease prevention programs, it can now provide tailored treatment anytime, anywhere, and address a range of health conditions, such as heart disease and smoking cessation, all while enhancing the patient experience.

#### **5.3.4 REMOTE INTERVENTION**

Doctors can administer medication and assess patient response in real time using data from sensors. Such prompt actions provide cutting-edge medical aid and lower hospital expenses. Through remote health state monitoring or telemedicine, it is possible to:

- Gather information on a patient's health before they see a doctor;
- Remotely consult with people on urgent issues;
- Compile a list of the patient's activities and chronic disease manifestations; and
- Offer quick assistance in life-threatening situations (such as a stroke).

This idea suggests replacing medical staff with intelligent apps on various devices. For the treatment of stroke, Shirley Ryan Ability Lab uses specialized flexible equipment. They are fastened to the neck to monitor swallowing and speech impairment (aphasia). IoMT makes it possible for medical devices like ECG sensors, glucose meters, and heart rate monitors to gather, process, and relay patient data. The most intriguing possibilities made possible by medical equipment's ability to share real-time data include:

- Improved care outcomes;
- Services for remote patient monitoring that are simplified;
- Lower healthcare expenses, especially in post-acute care settings (PAC); and
- Access to current patient health data, remote monitoring of linked device health and performance.

#### *5.3.4.1 INSIGHT OF CONNECTIVITY*

IoMT allows for both machine-to-machine and machine-to-human communication, enabling precise and dependable real-time intervention options such as monitoring a patient's vital signs. Medical gadgets can communicate with other internet devices as they have Wi-Fi connectivity, giving them the ability to transfer data to cloud storage via a secure network, expanding their capabilities even further.

When IoMT devices and EHR systems work together, tremendous potential is unlocked that has the potential to fundamentally alter healthcare delivery and systems [11].

IoMT primarily falls under the following categories in current clinical settings:

- **Remote Patient Observation:** Medical professionals and clinicians can remotely monitor a patient's heart rate, blood sugar levels, and other symptoms using medical gadgets. The ability to remotely track patient vitals is especially useful for monitoring patients with chronic conditions. Care providers can take preventative action in the event of crises such as sudden cardiac arrest (SCA) or epileptic seizures by receiving patient health notifications on their mobile devices. The future of patient health monitoring involves a combination of cutting-edge sensors, smart devices, and software suites to drive the sensors. These applications range from telehealth to chronic illness management and preventive treatment.

- **Drug Management:** Radio frequency identification (RFID) technology is used by IoMT devices to track and manage the movement of prescription medications throughout the supply chain. Pharmaceutical companies have started embedding RFID tags on the packaging of medicines to ensure uniform supply chain quality. The “smart pills” that help caretakers and patients’ loved ones keep track of medicine dosages represent the most ambitious deployment of IoMT so far.
- **Equipment Inspection:** The performance of connected medical devices can also be examined remotely. When maintenance is necessary, equipment such as MRIs, X-rays, CT scans, and other regularly used medical devices can notify hospital staff. Medical device providers can anticipate technical failures with the aid of predictive maintenance and remote diagnostics before they have an impact on patient care.
- **Biosensor Surveillance:** Analytical tools called biosensors can monitor biological substances such as blood, tissue, antibodies, and nucleic acids. They work with wearables and smartphones. Biosensors are among the most accurate configurations in the Internet of Medical Things (IoMT) and serve as the cornerstone for point-of-care (POC) and at-home diagnostics. There are many end-user applications where biosensors are used, but glucose monitoring is currently the most common one.

### **5.3.5 ENHANCING DRUG MANAGEMENT**

Pharmaceutical supply costs and availability issues can be addressed through the use of IoMT-based RFID tags. The FDA has proposed regulations for managing the pharmaceutical supply chain and implementing radio frequency identification (RFID) technology. This includes labeling pharmaceutical packaging with tags that manufacturers can utilize to ensure the integrity of the supply chain. Another option is to incorporate this technology directly into pharmaceuticals. WuXi PharmaTech and TruTag Technologies have developed an edible “smart” IoT tablet to monitor patient pharmacodynamic and drug dosage. Pharmaceutical companies benefit from these solutions as they help reduce risks and losses in their supply chains and provide better control. The infrastructure that encompasses intelligent hardware, software, medical systems, and intelligent services is known as the Internet of Medical Things (IoMT) or Health IoT. This technology advances healthcare, enables remote assistance, and gathers more patient data. All healthcare products

and services based on IoT can be broadly categorized into two groups: those intended for clinics and medical professionals, and those intended for users or patients.

Data on general health conditions, allergic reactions, tests, and other information are gathered and processed by smart devices (smart appliances, sensors, health meters, etc.). They facilitate the quick creation of patient statistics and medical records by medical staff [3]. The ability to receive data from connected systems helps doctors better manage patient care, offer timely assistance, and prevent the recurrence of chronic illnesses. Principal benefits of smart devices and medical solutions:

- Improved staff performance and mobility;
- Quicker processing of patient data;
- Reduced risk of error and human factor; and
- Cost-effective treatment.

According to the research, three-quarters of healthcare executives believe that IoT will disrupt the medical industry within the next two years. Three areas will initially see changes: data collection, wearable-assisted chronic disease prophylaxis, and remote patient monitoring.

## **5.4 IMPLEMENTATION OF IOMT-BASED APPLICATIONS AND USE CASES**

### ***5.4.1 ARCHITECTURAL OVERVIEW OF IOMT SYSTEM IMPLEMENTATION***

Three layers make up the macro-IoT architecture: local devices, connectivity, and data analytics and solutions [12]. Following is a discussion of each component's structural and functional aspects:

#### ***5.4.1.1 LOCAL SYSTEMS AND CONTROL***

One of the fundamental components of the IoT is distributed intelligence. The main objective of distributed intelligence is to build medical equipment with intelligent controls. This enables handling operational data both locally and on centralized servers. These devices often include sensors to track operating characteristics, converters to create digital inputs, controllers to make real-time decisions based on the converter inputs, and network

connections to share data with other machines/central servers. They consist of implants, portable diagnostics for doctors, and portable monitors, among other examples. Advanced electronics compatibility and integration are other factors that influence the adoption of device-level IoT solutions. These devices can securely communicate real-time qualitative biometric data from a patient's body to higher-level structures. The next layer of the ecosystem receives data transformed by encoders, actuators, and cryptographic devices. Analytical communication protocols such as NFC and over-the-air programming are examples of such protocols.

#### **5.4.1.2 CONNECTIVITY AND DATA LAYER FOR DEVICES**

The main goal of this layer is to gather data from interconnected devices and store it in predetermined data backups. The technology used at this point is not specific to any particular solution, such as patient monitoring. Secure medical data transmission technology enables large-scale data management and quality control. System integrators and end users can request advanced technology from network companies like Cisco and Oracle.

#### **5.4.1.3 ANALYTIC SOLUTIONS LAYER**

No matter what kind of healthcare solution is currently in use, central/remote servers gather data from various devices via the network and its essential parts. Real-time operational data analysis is done by servers with embedded algorithms to offer knowledge and insights. This data-driven due diligence aids with disease prediction, diagnostic capabilities, and the implementation of preventive measures. Healthcare solutions like remote patient intervention, chronic disease management, and monitoring are made possible by the collective and thorough analysis of data from different devices, including implants and smart devices.

### **5.4.2 IOMT APPLICATIONS**

The Internet of Things (IoT) and other supporting technologies have made it feasible to improve people's quality of life while also saving lives in the field of medicine. In addition to creating new application areas, it has changed how clinical decision-making, data collection, and patient record-keeping

are conducted. The term “IoMT” refers to the convergence of medical applications and equipment that can be connected to medical information technology systems. The IoMT’s ability to accelerate the transmission of medical data and ensure the connection between patients and doctors has reduced the number of unnecessary hospital stays and the burden on the healthcare system [13]. The Internet of Things (IoT) market is served by the production of smart devices such as wearable and medical monitoring equipment. These devices can provide telemedicine services or real-time location tracking in hospitals, communities, or homes.

#### ***5.4.2.1 IOMT NEEDS: THE HEALTHCARE PROFESSIONAL***

According to medical experts, the IoMT has the potential to help in disease diagnosis and significantly increase the efficacy of hospital medical services. The accuracy and usefulness of conclusions can be improved as clinical staff continuously monitor patient well-being data obtained through sensors using IoT innovation, helping to monitor and prevent persistent illnesses. As the status of patients in daily life may be determined, medical practitioners can concentrate on the care of patients who require in-person visits or treatments, enhancing productivity and efficiency. The Internet of Things (IoT) can also grow more swiftly in the medical sector if low-cost, lightweight sensors with wireless communication and integrated circuits (ICs) that use less power are developed. Improvements in hospital operations and clinical processes, simplification of medical information processing processes, enhancement of drug management procedures, prediction of maintenance issues, and improvement in healthcare cost-effectiveness can all be linked to a decrease in medical data errors in hospitals that provide IoMT services. It makes sense.

#### ***5.4.2.3 IOMT NEEDS: IN PERSON***

The use of IoMT technology reduces hospital stays from the patient’s perspective, making it more convenient for them. IoMT devices minimize unnecessary hospital visits, waiting times, and hospitalization durations by addressing the inconvenience of having to undergo testing during each visit. This is achieved by gathering, evaluating, and transferring patient data beforehand. As a result, costs and time requirements may decrease. IoMT technology enables the implementation of personal emergency response

systems (PERS), remote patient monitoring (RPM), and telemedicine in the home. PERS combines wearable and relay devices with real-time medical call center services to help the elderly and individuals with limited mobility become more independent at home. It facilitates quick communication between medical staff and patients, as well as the provision of emergency medical assistance. RPM supports long-term care and manages chronic diseases at home through the use of sensors and living room monitoring technology. Consumer-grade fitness and health devices, such as fitness bands, wristbands, sports watches, and smart clothing, are examples of consumer health wearables, in addition to activity trackers. However, the majority of wearable health devices for consumers have not yet been approved by regulatory agencies. Experts may approve them after cross-clinical validation or consumer inquiry for certain health applications. Patients are observed during acute home monitoring to expedite recovery and prevent readmission. Users are provided with medication awareness and dosage instructions to simplify medication management and enhance compliance. Telemedicine enables virtual counseling to assist patients in managing their conditions and obtaining prescriptions or courses of action. For example, visual perception or computer testing may be utilized to provide guidance and assess side effects and sores.

## 5.5 DISCUSSIONS AND FUTURE SCOPE

Despite being in their infancy, IoT-based clinical innovation applications have the potential to significantly increase the provision of medical services. The greatest benefit may be increased functional productivity as a result of increased use of related devices. More lives can be saved now than at any other time in recent memory thanks to the straightforward information transfer from low-level physical devices to the cloud (and related information analysis). With the help of information-driven dynamics, caregivers can carefully monitor a patient's overall health, take precautions, and react quickly to emergencies.

Intelligent devices that can reduce patient expenses, increase patient consistency, and provide quick clinical response are anticipated to be used within the connected framework. While health monitoring machinery is designed to be functionally effective, it can still pose serious risks during use, such as information theft, shaky information transmission, and weak organizational ties [14]. The development of IoT-based organizational and

information arrangements is intended to be fueled by these challenges, as well as administrative barriers. Government initiatives, such as the Patient Security and Reasonable Consideration Act, which focuses on validated EHR (Electronic Health Records), may be able to improve the standard of care, as well as its efficacy and quality. Far-reaching technology and international information norms that support dependable information handling still have room for improvement.

The integration of artificial intelligence into various organizations, both in the local research community and in the field of Federated Learning (FL), has significantly improved due to recent advancements in medical technology. This study thoroughly examines a wide range of topics including FL, AI, Explainable AI (XAI), and e-Medical services. Additionally, we regularly discuss the latest advancements in FL and AI applications in cutting-edge healthcare. FL-Medical services, FL-Artificial Intelligence, and other related areas are closely interconnected. The aforementioned concepts also address important aspects such as security, privacy, reliability, flexibility, and confidentiality. It is important to note that medical services extend beyond just FL and AI methodologies, although we have incorporated FL and AI XAI approaches into various frameworks for healthcare. However, there are still challenges and adjustments that need to be addressed. The implementation of the Internet of Medical Things (IoMT) in healthcare settings faces several difficulties, with the most significant ones being costly infrastructure and data security. While there are other contributing factors, these are the primary issues that require immediate attention to ensure the widespread adoption of this technology.

When it comes to allied technologies in IoT systems, NFC and RFID are two connectivity technologies that are present in the majority of mobile consumer devices today. This enables seamless communication between data generators and processing devices. Without having to buy pricey monitors that connect to medical equipment, doctors, patients, and families can benefit from IoMT. Chronic disease rates are significantly rising, driving demand for more effective therapies and more affordable medical care. Due to the steadily rising computational power of mobile devices and the need for innovation in healthcare to improve care outcomes, IoMT can create paradigm-shifting business trends. Hospitals and Independent Software Vendors should therefore invest in this technology because it is expected to become increasingly important over the next periods. Table 5.1 provides a basic comparative analysis of application scenarios with and without IoMT systems.

**TABLE 5.1** Comparison of Impact Delivered by IoMT Devices on Healthcare Applications

<b>Without IoMT</b>	<b>With the Use of IoMT</b>
Long-term hospital stays to ensure that patients' health will recover after being discharged.	By using home health monitoring systems, hospitalization expenditures were reduced and the standard of care was raised.
Risk of a false diagnosis when diagnostic test results are interpreted arbitrarily.	Care decisions are made by providers that are more precise and adhere to industry best practices.
When several doctors work together to provide care for a single patient, this is an instance of care coordination.	The IoMT devices' high-quality data collection on patient health enables the delivery of accurate care even when healthcare professionals switch.
There are very few care options available in post-acute care (PAC) and home health settings.	Expand the scope of health conditions that can be monitored in PAC and home health settings.

## 5.6 CONCLUSION

With the increasing requirements and awareness about Quality of Service (QoS) in healthcare services, IoT-based applications have found widespread applicability in healthcare scenarios. The Internet of Medical Things (IoMT) has the potential to be a promising solution for advancing well-being observation and patient outcomes, considering the associated benefits and challenges. This innovation caters to a future of personalized medical services and higher expectations for daily comforts worldwide, providing individualized, information-driven treatment regimens and improved devices based on physiological needs. Recent technological advancements in sensors, organization, distributed storage and processing, versatility, and large-scale data analysis have sufficiently progressed to support the development of practical, intelligent clinical devices and related environments for medical services. By integrating Federated Learning (FL) into existing IoMT systems, the applications have the potential to perform even better than before. Additionally, enhanced privacy and security aspects can be achieved in such systems, which is highly desirable in healthcare scenarios. Thus, our study concludes that IoMT and FL are two highly useful technologies that can mutually benefit from each other when properly integrated.

## KEYWORDS

- **artificial intelligence**
- **federated learning**
- **healthcare scenarios**
- **internet of medical things**
- **internet of things**
- **medical diagnostics**
- **medical services**

## REFERENCES

1. Alamleh, A., Albahri, O. S., Zaidan, A. A., Albahri, A. S., Alamoodi, A. H., Zaidan, B. B., Sarah, Q., et al., (2022). Federated learning for IoMT applications: A standardization and benchmarking framework of intrusion detection systems. *IEEE Journal of Biomedical and Health Informatics*.
2. Nguyen, D. C., Quoc-Viet, P., Pubudu, N. P., Ming, D., Aruna, S., Zihuai, L., Octavia, D., & Won-Joo, H., (2022). Federated learning for smart healthcare: A survey. *ACM Computing Surveys (CSUR)*, 55(3), 1–37.
3. Nair, A. K., & Sahoo, J., (2023). Internet of things in smart and intelligent healthcare systems. In: *Intelligent Internet of Things for Smart Healthcare Systems* (pp. 1–19). CRC Press.
4. Nair, A. K., Chinju, J., & Jayakrushna, S., (2022). Implementation of intelligent IoT. In: *AI and IoT for Sustainable Development in Emerging Countries: Challenges and Opportunities* (pp. 27–50). Cham: Springer International Publishing.
5. Farhad, A., Woolley, S., & Andras, P., (2021). Federated learning for AI to improve patient care using wearable and IoMT sensors. In: *2021 IEEE 9<sup>th</sup> International Conference on Healthcare Informatics (ICHI)* (pp. 434). IEEE.
6. Ahmed, J., Tu Ngoc, N., Bakhtiar, A., Awais, J., & Jawad, M., (2022). On the physical layer security of federated learning-based IoMT networks. *IEEE Journal of Biomedical and Health Informatics*.
7. Nair, A. K., EbinDeni, R., & Jayakrushna, S., (2022). An overview of artificial intelligence for advanced healthcare systems. *Digital Health Transformation with Blockchain and Artificial Intelligence*, 141–160.
8. Binson, V. A., Subramoniam, M., Ragesh, G. K., & Ajay, K., (2021). Early detection of lung cancer through breath analysis using AdaBoost ensemble learning method. In: *2021 2<sup>nd</sup> International Conference on Advances in Computing, Communication, Embedded and Secure Systems (ACCESS)* (pp. 183–187). IEEE.
9. Ragesh, G. K., & Baskaran, K., (2012). A survey on futuristic health care system: WBANs. *Procedia Engineering*, 30, 889–896.

10. Razdan, S., & Sachin, S., (2022). Internet of medical things (IoMT): Overview, emerging technologies, and case studies. *IETE Technical Review*, 39(4), 775–788.
11. Ragesh, G. K., & Baskaran, K., (2016). Cryptographically enforced data access control in personal health record systems. *Procedia Technology*, 25, 473–480.
12. Nair, A. K., Jayakrushna, S., & Ebin, D. R., (2023). Privacy-preserving federated learning framework for IoMT-based big data analysis using edge computing. *Computer Standards & Interfaces*, 103720.
13. Otoom, Y., Yue, W., & Amiya, N., (2021). Federated transfer learning-based IDs for the internet of medical things (IoMT). In: *2021 IEEE Globecom Workshops (GC Wkshps)* (pp. 1–6). IEEE.
14. Ahmed, S. T., Vinoth, K. V., Krishna, K. S., Akansha, S., Muthukumaran, V., & Deepa, G., (2022). 6G enabled federated learning for secure IoMT resource recommendation and propagation analysis. *Computers and Electrical Engineering*, 102, 108210.
15. Jiang, Jichu & Kantarci, Burak & Oktug, Sema & Soyata, Tolga. (2020). Federated Learning in Smart City Sensing: Challenges and Opportunities. *Sensors*. 20. 6230. 10.3390/s20216230.



Taylor & Francis

Taylor & Francis Group

<http://taylorandfrancis.com>

## CHAPTER 6

---

# Federated Learning Approaches for Intrusion Detection Systems: An Overview

AKARSH K. NAIR,<sup>1</sup> JAYAKRUSHNA SAHOO,<sup>1</sup> and GAURAV JASWAL<sup>2</sup>

<sup>1</sup>*Department of Computer Science and Engineering,  
Indian Institute of Information Technology, Kottayam, Kerala, India*

<sup>2</sup>*iHUB and HCI Foundation, Indian Institute of Technology, Mandi,  
Himachal Pradesh, India*

---

## ABSTRACT

The rapid growth of the internet and similar networking technologies, along with smart devices, has triggered a sudden boost in communication networks. Most of the smart devices we use, such as mobile phones, wearable devices, and autonomous driving systems, are part of distributed networking paradigms that produce high volumes of data in a short duration of time. The need for real-time computation in such systems has also resulted in the development of strong communication channel transmission technologies and allied devices. The classical centralized architecture is gradually shifting to edge-based devices due to these reasons. Thus, due to the security concerns faced by these devices, intrusion detection systems have started gaining popularity as an effective deterrent to impending privacy breaches and adversarial attacks. Even though intrusion detection systems based on classical learning approaches such as ML and DL perform well, they are not ideally suited for distributed computing-based systems. Therefore, federated learning needs to be integrated with existing IDS approaches to develop a system with better performance efficiency. To provide a solid reason behind the rationale for

choosing FL-based IDS, we evaluate basic ML approaches and present their shortcomings as well. We also present some of the existing FL-based IDS approaches. Finally, we discuss most of the existing challenges on various aspects alongside presenting the future research directions on the same.

## **6.1 INTRODUCTION**

The internet has undergone significant development in terms of connectivity and volume in recent years. Alongside this, various types of interconnected devices and systems have emerged. Consequently, network systems are constantly vulnerable to various security threats and vulnerabilities, including intrusions and malicious attacks. Intrusions are typically classified as attempts to breach systems, compromising their confidentiality and integrity. To counter such adversarial situations, efficient intrusion detection systems (IDS) must be developed [1]. In the context of network security, IDS plays a crucial role in providing a fundamental level of security. Depending on the anomalies to be detected, IDS also employ various techniques. Traditional IDS approaches, such as signature-based anomaly detection, have become less effective over time as newer attacking methodologies are devised.

Over the last few years, AI-based technologies such as machine learning (ML) and deep learning (DL) have undergone extensive changes. They are currently being employed for various use cases related to healthcare, computer vision, communication networks, and even security-based applications. AI reduces the need for human intervention in security-based use cases. For example, AI-based IDS systems for anomaly detection in surveillance cameras are a real-life example. Although several ML and DL models have already been successfully applied for IDS applications, the current centralized model training architecture requires users to collect data at a single point, which raises security concerns. Motivated by such issues, Google proposed a distributed learning paradigm called federated learning (FL), which focuses on performing on-site model training and offers features such as data locality and privacy preservation. FL is a collaborative learning methodology, meaning that devices learn cooperatively. A single model is trained over multiple devices using decentralized data sources, and the final model is generated through some sort of averaging procedure. Generally, federated learning (FL) systems consist of two different procedures. The first procedure involves performing local model generation, while the second procedure involves performing global model generation using gradients obtained during local model training. In FL systems, apart from transferring

gradients, no other data transfer occurs between the server and the local devices. This ensures enhanced privacy preservation for the entire system, along with the added benefits of cost reduction compared to centralized machine learning (ML) systems. Generating the global model in federated learning systems is an iterative process. The training begins with the central server sending a global model to the client devices, and the client devices send back the gradients after training on local data. This process continues until a pre-set accuracy is achieved or the global model converges. Therefore, it can be inferred that even though individual client devices may not possess large amounts of data for deep learning (DL) or machine learning training, federated learning enables them to access high-quality models and achieve better system performance through collaborative functioning.

In general, with respect to the data distribution of individual clients, the Federated Learning (FL) architecture can be divided into three main subsections: horizontal FL, vertical FL, and Federated Transfer Learning.

Horizontal Federated Learning refers to FL data distributions where the datasets processed by individual clients have similar feature spaces but varying observations. Similarly, with vertical Federated Learning, individual datasets will contain data from different domains, thus having varying feature spaces. In the case of Federated Transfer Learning, the datasets on client devices will be entirely different, meaning that they differ not only in terms of instances but also feature spaces.

Currently, Federated Learning (FL) is being applied in various domains related to healthcare, finance, and others. FL has also been integrated into intrusion detection systems (IDS) in recent years, primarily for anomaly detection based on client devices. The combination of FL with IDS also enables the development of customized defensive mechanisms for light-weight devices. Additionally, it helps overcome computational complexity-related issues, ensures data privacy, optimizes bandwidth utilization, and improves system performance in heterogeneous environments. Although there are already articles discussing the combination of FL with IDS, there is still a need for a comprehensive survey. Therefore, we will attempt to provide a comprehensive survey of articles discussing the integration of FL with IDS.

## 6.2 INTRUSION DETECTION SYSTEMS

An intrusion detection system is a highly relevant component of any information security system aiming to identify and isolate instances of malicious

attacks and anomalies in such systems. With respect to the input data being given, IDS can be mainly classified into host-based IDS and network-based IDS. The major difference between the two is that while the former attempts to gather and evaluate logs of operating systems and other applications, network-based IDS monitors and evaluates network traffic only. The classical IDS architecture mainly comprises the following components: components for collecting data and its associated processing, components mainly aiming for evaluating and adversarial scenario detection, and finally response generation [2]. Most of the IDS architecture also comprises a data repository that is entitled to collect and store raw data and system triggers along with a knowledge base where the information regarding triggering rules, security signatures, and adversarial attack patterns are stored.

When it comes to anomaly detection information systems, two main approaches have been employed: the signature-based approach and approaches built on artificial intelligence (AI). Signature-based approaches have been prevalent for a very long time and they mainly employ pattern-matching techniques to identify specific adversarial scenarios. In such systems, the attack knowledge base needs to be frequently updated to keep up with the changing attack patterns and maintain high levels of detection accuracy [3]. Signature-based IDS systems are ideally suited for identifying existing attack types, but they fail miserably in case of newly developed attacks and zero-day attacks. AI-based IDS systems find their applicability in such use cases. Currently, a wide range of AI techniques has been employed as part of anomaly detection in IDS systems, such as genetic algorithms, clustering, deep neural networks, and so on. Further classification of IDS is performed based on the connection and coordination of various components present in the system. They are monolithic, distributed hierarchical, and agent-based systems. The various types of IDS classifications are presented in the following section.

### ***6.2.1 CLASSIFICATIONS OF INTRUSION DETECTION SYSTEMS***

When it comes to the classification of IDS, various categories are generated based on different aspects. Some of the common aspects include the region of applicability, model of detection, methods employed, system architecture, and so on. We will be discussing some of the common classification types in detail in the following section.

### ***6.2.1.1 CLASSIFICATION BASED ON REGION OF WORK***

As discussed previously, host intrusion detection systems and network intrusion detection systems (NIDS) are the two major classifications of IDS.

Host Intrusion Detection Systems (HIDS) are responsible for monitoring internal factors of a system, such as user logs and user actions. HIDS systems maintain a record of system events, detecting any occurrences of anomalous events to keep track of operations and memory regions, thus ensuring the system's status is updated. However, HIDS are vulnerable to tampering attacks, and there is a need for tamper-proof or highly relevant methodologies in such systems [4]. Another major disadvantage is the resource-intensive nature of the system, which affects the overall performance of the host system. These methodologies also fail to identify attacks unless they extensively affect the system. Therefore, the host-based idea system is not a perfect solution for a wide range of adversarial conditions, and better alternatives need to be devised.

The NIDS primarily evaluates system traffic, both incoming and outgoing. Therefore, NIDS must be deployed at specific locations throughout the network to evaluate all connections. The network IDS system assesses all traffic passing through each sub-network and examines it for anomalies based on existing knowledge. One notable feature of NIDS is that it remains undetected by intruders during evaluation and traffic monitoring [5]. However, the system may face bottleneck issues due to the high volume of packets during peak hours. Additionally, challenges related to VPN authentication and the lack of methodologies for interpreting encrypted data are drawbacks of such systems.

### ***6.2.1.2 CLASSIFICATION BASED ON MODE OF DETECTION***

As with HIDS and NIDS, another classification based on the mode of detection is also performed. In this classification, two subcategories are identified: signature-based and anomaly-based detection systems.

The signature-based systems are one of the primitive models in network IDS. As discussed earlier, these systems rely on pattern recognition methodology for anomaly detection, and benign patterns are inferred from a knowledge base available in the system. Thus, they are also referred to as knowledge-based detection approaches as they also keep a record of various

patterns obtained from previous attacks [6]. These systems aim to generate a library with a wide range of past adversarial signatures to improve the future efficiency of detection procedures. However, as these systems are dependent upon existing signatures in the library, new attacks render these systems futile.

Anomaly-based intrusion detection systems were devised to overcome the challenges faced by signature-based intrusion detection systems. In these systems, the user's behavior is the sole criterion for classifying them as malicious or benign. These systems already possess knowledge of behavioral patterns generated from normal and benign users, using statistical and other approaches. If a user deviates from the established behavioral pattern, which is considered benign, they will be classified as having adversarial intent. This deviation between the expected and observed behavior is referred to as the extent of anomaly [7]. Similar to their predecessors, anomaly-based intrusion detection systems also have several limitations. One major drawback is the inability to evaluate encrypted data packets. Additionally, the generation of normal data patterns puts a significant burden on the system when the data volume increases.

#### *6.2.1.3 CLASSIFICATION BASED ON SYSTEM ARCHITECTURE*

Another classification is performed based on the architecture of the IDS system. They are mainly categorized into two categories, with further subcategories formed as a result of an effective combination of the primitive categories. The categories are as follows:

In distributed IDS systems, the assumption is that the IDS is entirely distributed, meaning that each individual device in the information system has its own intrusion detection system. It is also estimated that such distributed IDS systems can communicate with one another and a centralized dedicated server system [8]. The central IDS also performs the final data analysis and decision-generating procedures.

When it comes to hierarchical IDS, the same assumption of the locally distributed IDS system applies. Additionally, the local systems are initialized into clusters, and each cluster will have a dedicated cluster head that can perform inter-cluster communications [9].

#### *6.2.1.4 AI-BASED INTRUSION DETECTION SYSTEMS*

AI-based methodologies have found strong applicability in developing intrusion detection systems for real-life application scenarios. In centralized

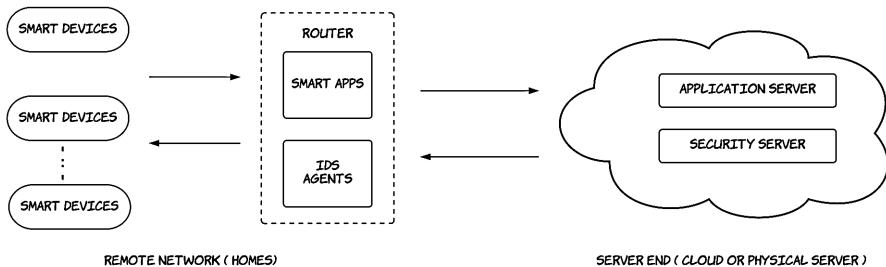
scenarios, ML and DL approaches have been widely employed for anomaly detection. Article [10] presents an IDS system built on random forests and decision trees for safeguarding training ground-based communication systems. In the subsequent layers of anomaly detection, a state observer is employed. The effective combination of both these layers generates an efficient intrusion detection system. Such systems only make use of ML techniques and still have scope for further enhancements in efficiency and effectiveness. An SVM-based two-stage approach is proposed in [11]. The IDS aims to protect advanced metering infrastructure from cyber-attacks. The functionality of SVM is to identify adversarial behavior inside the meter. For generating attack routes and identifying possible attacks, the Temporal Failure Propagation Graph is employed. It analyzes the similarity between recent anomalous events and existing events and does basic pattern matching to classify it as malicious or not. The model was designed for a single data type, making the model highly selective and decreasing its applicability as well. A similar article based on a gated recurrent unit-inspired recurrent auto-encoder is presented in [12]. The article focuses on identifying anomalies in controlled area network-based automatic embedded systems under varying adversarial conditions. An Intrusion Detection System (IDS) is proposed in [13] to prevent adversarial attacks in smart grid-based Supervisory Control and Data Acquisition (SCADA) systems. The proposed approach utilizes gradient boosting as a feature selection methodology and decision tree-based approaches for classifying malicious and benign cases. While machine learning-based approaches offer accuracy and effectiveness, there is always a need for better-performing systems. Therefore, Deep Learning (DL) based intrusion detection systems have been proposed as an effective alternative to existing methods.

Tian et al. [14] proposed a benchmark planning-based intrusion detection system for web-based anomaly detection. The system was developed to safeguard multiple web-based applications across a vast geographical area. Although the system performed well in real-life scenarios, there is a need for better DL methodologies for real-time applications. Farivar et al. proposed a methodology in the article [15] that employs intelligent variable structure control for evaluating attacks in nonlinear cyber-physical systems. This methodology focuses solely on Cyber-Physical Systems (CPS) and utilizes a combination of nonlinear control and basic artificial neural networks for efficient functioning. Compared to existing literature, the proposed methodology demonstrates high levels of merit and efficiency. However, like previous models, the proposed methodology also has issues related to applicability to varying domains.

### 6.2.2 SELECTION OF IDS ARCHITECTURE

The usual criterion for selecting the architecture of an IDS system depends on factors such as the category of the information system, computational power, energy requirements, network bandwidth, and so on. For example, IDS systems associated with cloud computing are typically distributed. Similarly, when it comes to IoT-based systems like smart homes and video surveillance systems, it is recommended to opt for a hybrid architecture such as distributed hierarchical agent-based systems. The recent reason for choosing such an architecture is that, when combined with traditional IDS systems, the typical surveillance systems have a cloud storage backend, which requires the system to continuously transfer data for storage and backup purposes. In such a hybrid architecture, software-based IDS will also be present inside the hardware component of the system. These software-based components synchronize the entire functioning of the hardware system and various other functionalities related to anomaly detection [2].

The architecture of a conventional smart home IDS system is shown in Figure 6.1. The system comprises several smart devices at the local end (in homes or institutions) and an associated smart router and other software components for initial analysis of the generated data. Then, the data is forwarded to the server where extended analysis can be performed. A major issue associated with such systems is the lack of privacy for the users involved, as their daily activities are constantly being monitored and analyzed.



**FIGURE 6.1** Smart home IDS architecture.

The hierarchical distributed variant of IDS architecture is highly popular and widely employed for adversarial attack detection and anomaly detection in smart grids and similar cyber-physical systems [9]. Smart grids are typically higher-level applications of home energy management systems,

encompassing a wide range of components such as power stations, end users, various electrical components, and more.

The hierarchy of local IDS systems can depend on the characteristics of collaborating devices or the intrusion limits, which are boundaries set to control the extent of damage to systems even in successful adversarial scenarios. As a result, FL systems and current-day IDS systems share multiple common factors. For example, the system architecture of server-based FL systems closely resembles the architecture of distributed IDS systems. When evaluating cloud-based IDS environments, they bear a strong resemblance to the characteristics of cross-silo FL systems in terms of computational resource management and data storage facilities. Cross-device FL systems employ a similar methodology as IDS systems built for IoT environments, as both architectures consider factors such as power expenditure, communication bandwidth, and resource management. While an exact match for data partitioning in FL cannot be identified in IDS, network-based IDS systems closely resemble horizontal FL systems in their use of horizontal data partitioning methodologies. In terms of host-based IDS, the data typically consist of logs from different use cases with a similar format, which closely resembles the vertical data partitioning scheme of vertical FL systems. The main motivation for applying Federated Learning (FL) on intrusion detection systems is its ability to significantly reduce privacy concerns when dealing with sensitive and personalized data. Additionally, in the case of FL systems handling sensitive data from highly complex organizations in the finance and healthcare industry, the presence of a set of collaborating devices enhances the overall trust level of the system. Features such as data locality and redundant privacy-preserving methods also contribute to the trust factor. Moreover, in IoT-based environments, FL is capable of reducing the total volume of network traffic, indirectly aiding in the development of an optimized network in terms of energy consumption, which further supports heterogeneous systems. All these optimization issues have served as strong motivations for conducting these studies, and the following session will discuss them in detail.

### 6.3 FL ARCHITECTURE FOR DEPLOYMENT OF IDS

When deploying an intrusion detection system, the performance efficiency depends on factors such as the system architecture and the type of IDS being deployed. In the current scenario, three major classifications of IDS frameworks are being used: the centralized framework, decentralized framework,

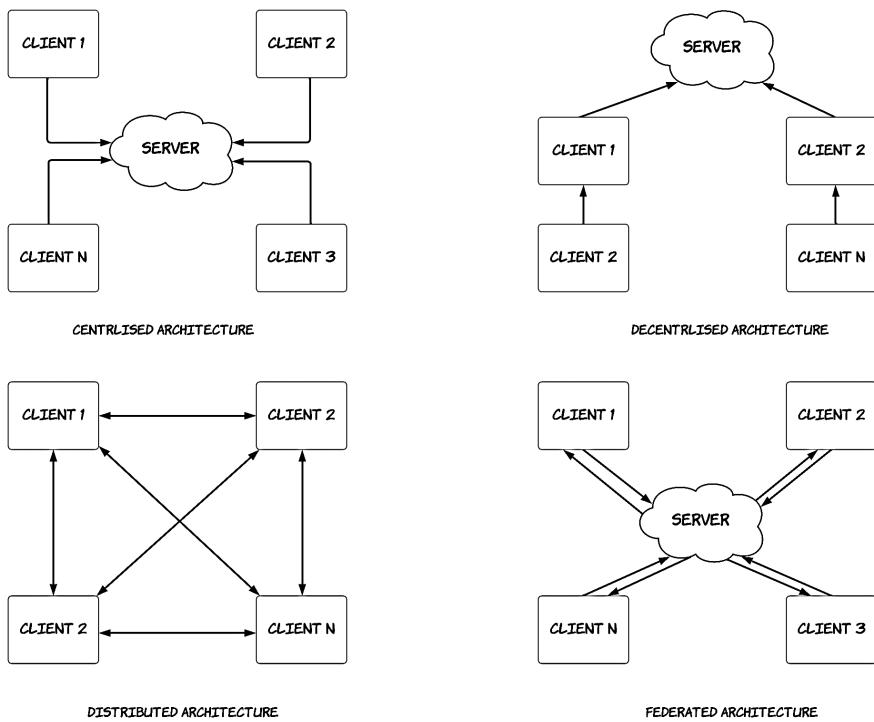
and distributed framework. In the centralized framework, the system consists of two main phases. The first phase is the data collection phase, where data is collected from central storage entities such as the cloud or central server. The second phase involves using the retrieved data to generate ML or DL models. The centralized framework is typically deployed for IDS running in small-scale networks with limited scalability. This choice is made due to the constant need for data transfer from server to client in such IDS systems. Even when the bandwidth of the communication channels is increased in relation to the number of devices, the server still faces a significant bottleneck in such networks. In contrast to the centralized framework, the decentralized and distributed frameworks make use of multiple IDS for attack detection. These architectures often utilize edge computing principles to reduce overall traffic in communication channels and decrease processing latency.

In any system following an edge computing-based architecture, data localization is an inbuilt feature. In the case of IDS, it introduces another set of concerns. Thus, to overcome issues related to data localization, a distributed generative adversarial network-based IDS was proposed in the article [16]. The proposed model attempts to monitor IDS scenarios in the host system as well as the neighboring systems in the network. Even though the model improved the performance efficiency of distributed IDS, issues related to latency started to affect the system. Extensive research has proposed the introduction of federated learning into such IDS as a solution for overcoming most of the existing issues related to data localization as well as latency. Various types of IDS architectures can be seen in Figure 6.2.

The traditional FL architecture is based on a client-server system, where model aggregation occurs at the server end and basic model training at the client end. This method of process distribution prevents bottlenecks from being generated at the client end while still benefiting from employing edge computing-based principles. An article by Rey et al. [17] proposed the integration of FL with IDS for both multilayer perceptrons and auto encoders. The authors experimentally combined the models on different types of IDS architecture with FL and showcased the superiority of the FL approach. A similar article by Rahman et al. [18] presented and evaluated a FL-based IDS on the NSL KKD dataset. The proposed model also demonstrated that the FL counterpart outperforms classical centralized intrusion detection systems. FL-based approaches not only offer increased efficiency and data utilization but also a high degree of scalability. The increasing complexity and size of the federated network only enhance the capacity of the intrusion detection module.

Depending on the use case and user requirements, IDS can be deployed in networks of varying sizes or as stand-alone systems. Each type of deployment

has its own applicability and may not be suitable for other applications. For example, implementing a stand-alone IDS on a highly complex network can disrupt the normal transmission of packet data. Similarly, stand-alone NIDS may not be capable of handling adversarial scenarios in edge computing systems, just as HIDS may be incapable of dealing with indirect adversaries. To overcome these limitations without introducing additional challenges such as increased latency, it is highly recommended to effectively integrate federated learning with IDS. The increasing complexity of networks only enhances the performance of federated intrusion detection systems, making them an ideal choice for such use cases.



**FIGURE 6.2** Types of IDS architecture.

## 6.4 FL-BASED INTRUSION DETECTION SYSTEMS

As we all know, networks have been classified with respect to various criteria such as geographical area, user density, the distance between nodes, network traffic, and so on. For instance, if we consider geographical area-based

classification, networks can be easily classified into VAN, PAN, LAN, MAN, and so on. Therefore, when attempting to integrate FL-based IDS into any network, various network factors such as the number of devices, data being transferred, geographical area of the network, the volume of data being transferred, and similar other features need to be taken into consideration. These factors are also the main components that determine the susceptibility of a network to a particular type of attack, its intensity, and thus the ideal IDS for that system. In recent times, communication channels and methods have advanced to a very high extent, and huge networks connected over continents and even extraterrestrial nodes such as satellites are coming into existence. Even though this has been a significant advancement in communication, it has also made it easier for attackers to access systems regardless of their location and communication channels used. This highlights the importance of implementing IDS in all types of networks, including satellite-terrestrial integrated networks (STIN). As mentioned earlier, the requirements for FL-based IDS can vary greatly depending on the specific network. For example, in STIN networks, factors such as low packet loss rate, decreased CPU utilization, and energy optimization need to be considered [19]. In local area networks, data distribution can be a major challenge due to their fragmented nature. Therefore, in such scenarios, a segmented FL-based IDS is preferable over a centralized one. The rationale behind this choice is that the segmented FL architecture performs better for non-IID data, as it utilizes a higher number of global models compared to a single global model in a centralized architecture.

#### **6.4.1 FL FOR HETEROGENEOUS ANOMALY DETECTION**

Even though theoretically, anomaly-based IDS is a prominent type of IDS, in reality, such systems produce high levels of misclassified outputs, thus being least preferred in most instances. The reason behind the pitfall of such a model is due to insufficient and low-quality data, along with inefficient models, thus producing high levels of false alarms. Usually, attackers modify their methodology to avoid anomaly-based IDS systems. Thus, high levels of resources are required for developing IDS capable of detecting such attempts and precisely classifying anomalies. For heterogeneous systems, the training dataset should also have high levels of heterogeneity and a huge data volume, as intrusion attempts in such systems are also very difficult to detect. Even though distributed IDS is an optimal choice for such systems, data scarcity issues pertain to such systems as well. The integration of FL can

act as a solution for such data scarcity issues in IDS. It allows for effective collaboration with data present at client devices, thus increasing the total volume of available data and effectively improving IDS performance.

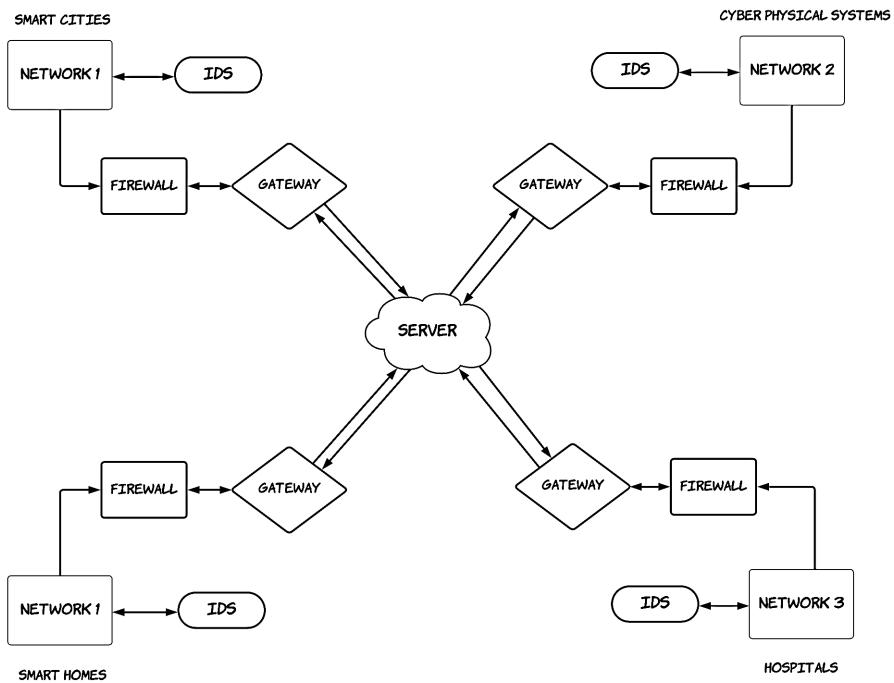
A collaborative IDS based on a disagreement policy-inspired semi-supervised learning methodology is proposed by Li et al. [20]. The proposed algorithm utilizes three trained classifiers to label all the unlabeled data through a voting mechanism. In this framework, the models are continuously updated whenever new data is labeled. The main objective of incorporating FL in such a system is to address issues related to data scarcity and handle unlabeled data in a semi-supervised learning environment. The proposed model demonstrates efficient performance compared to supervised ML approaches, offering improved accuracy and reduced false alarms. When it comes to large organizations with significant monetary assets that require protection by an IDS, ensuring security for individual nodes becomes a challenging task. These networks are often heterogeneous, meaning that each component possesses different properties in terms of data, communication channels, communication protocols, and even communication bandwidth, resulting in high levels of dissimilarity in system properties. This pattern persists even within sub-networks. Figure 6.3 illustrates a typical FL-based IDS system architecture designed for heterogeneous environments.

Accumulating such heterogeneous data into a single centralized point is a tedious task. Such levels of heterogeneity always generate confusion in the anomaly detection module, leading to false alarms and failed detections. Thus, as a solution for tackling this heterogeneity, the federated framework can be effectively employed. A comprehensive experimental analysis of various articles citing the usefulness of heterogeneous and non-IID data in Federated Learning (FL) is presented in the article [21]. Even though classical FL is superior to most of the centralized frameworks, dealing with non-IID data is still an issue in FL systems. Various other articles are also available where the non-IID and heterogeneous nature of data are harnessed to work on federated frameworks for both machine learning and Deep Learning (DL) models [22].

#### **6.4.2 FL FOR DISTRIBUTED ATTACK DETECTION**

Distributed attacks, such as distributed denial of service attacks (DDoS), are one of the most prominent threats to distributed systems and networks. The classical DDoS attack works by overloading the victim's device and exhausting the system by utilizing different adversarial devices to launch

collaborative DoS attacks [23]. In networking scenarios, DDoS attacks are one of the major reasons for fluctuating QoS for the end user. Thus, early detection and isolation of such attack scenarios is a factor that determines better QoS for the user as a whole. Due to this reason, researchers have started exploring the scope of FL-based IDS as a preventive measure against such DDoS attacks. Currently, literature is available in this context where FL-based IDS is being employed in various types of IoT-based networks for DDoS attack detection [24].



**FIGURE 6.3** FL-based IDS for heterogeneous environments.

#### 6.4.3 FL-BASED IDS FOR IOT SYSTEMS

Lightweight IoT devices with low specifications, such as edge devices, are always easier targets for adversaries. Every device in an IoT system has its significance, irrespective of its specifications, thus making the data valuable. As these IoT devices have low specifications, the power allocation is also very low. Lesser power availability is followed by decreased security features as well. Federated Learning (FL) acts as an effective solution for such problems

as well. As it is built on a distributed framework, it is capable of exploiting the maximum computational capability and also providing a better environment for Intrusion Detection Systems (IDS) to perform against adversaries [25]. As FL works on an iterative data transfer mechanism, interpreting the transferred packets can help the system identify any abnormal behaviors and thus adversarial attacks affecting the network. Thus, an automated IDS can help the system achieve such a feature where anomalous packets can be immediately dropped and the system safeguarded from suspicious users.

In federated frameworks, the anomaly detection modules are present at every communication node, and as usual, model aggregation is performed at the server. The anomaly detection model functions by first scanning the packets entering the system and then extracting the various features of the evaluated packets [26]. The extracted features are converted to symbols which later undergo probability computation on a GRU. As with input, the output parameters are also checked and packets are categorized into adversarial or benign.

## 6.5 CHALLENGES FACED AND FUTURE SCOPE

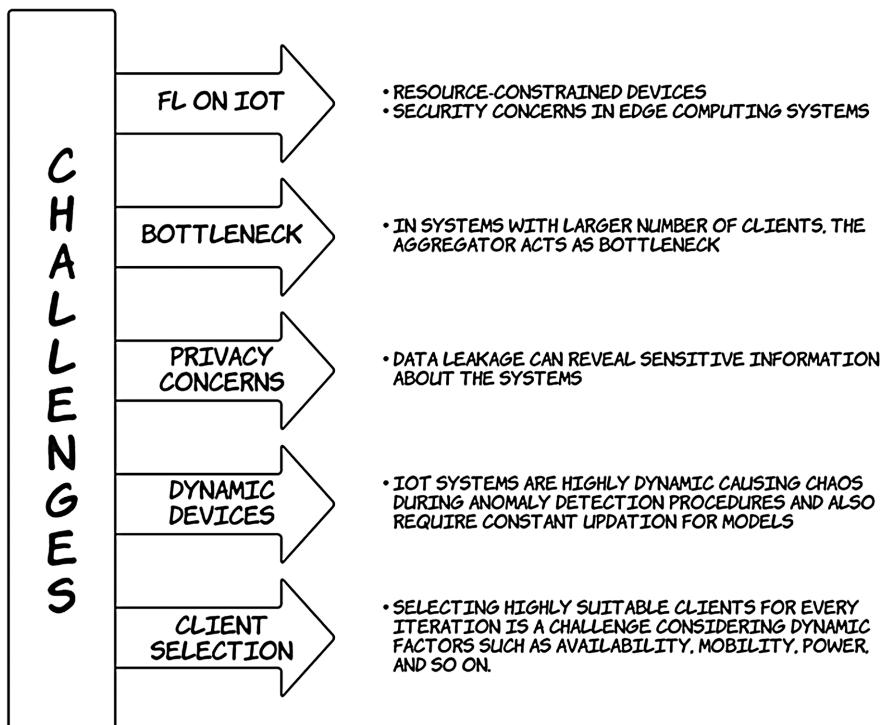
Based on the studies performed and analysis of the currently available literature, we can infer certain challenges and future directions for the effective integration of FL on intrusion detection systems. Figure 6.4 depicts some of the common challenges faced during the integration of FL with IDS. A detailed explanation of some of the challenges is described below:

### 6.5.1 LACK OF STANDARD IDS DATA SETS FOR FL

Most of the existing literature studies on FL-based IDS systems are conducted using general network datasets, which are not ideally suited for IoT-based and lightweight environments. Although new datasets specific to IoT scenarios have been released, they fail to provide a stable environment for testing various aspects of FL and IDS. Additionally, existing IDS datasets only offer skewed distributions, with either benign or malicious scenarios outweighing the other, resulting in futile results during experimentation. Therefore, it is necessary to develop better datasets that can simulate various adversarial scenarios, taking into account different system configurations of IoT and other lightweight networks, as well as the potential for integrating FL and IDS for improved experimentation and future research.

### 6.5.2 BOTTLENECK-RELATED ISSUES

Even though FL functions are based on a collaborative training mechanism, centralized FL systems rely on a single server device for performing model aggregation. Thus, this entity and its communication channels act as areas of communication bottlenecks and also as single points of failure. An effective solution for such problems is shifting from centralized FL to decentralized FL or incorporating other distributed technologies such as blockchain. The introduction of blockchain into such systems can guarantee better trust levels alongside maintaining a ledger of the procedures happening inside the network, which can be used during system failures [27]. Even though articles on the above-mentioned technology are available, it is currently not suited for IDS-based applications as they do not discuss aspects related to training round frequencies and the number of devices in case of larger networks, which are mandatory factors when working on IDS.



**FIGURE 6.4** Challenges faced during FL-IDS integration.

### **6.5.3 COMMUNICATION REQUIREMENTS**

From the initial stages of Federated Learning (FL) itself, developing a communication channel with significant bandwidth for gradient exchange was a genuine research problem [28]. In IoT-based scenarios, these issues escalate due to the communication and computational constraints of devices. Attempts to integrate FL on Intrusion Detection Systems (IDS) in such situations will only deteriorate the system performance even more. In the context of FL, two major factors define the communication channel bandwidth size between clients and server devices. The primary factor is the volume of data to be transmitted, and the second one is the total number of rounds to be performed for the model to converge. The second factor itself is determined by several other factors such as data distribution, number of clients, use cases, and so on. Several other techniques, such as gradient compression and sparsification, have been proposed in an attempt to reduce the volume of total data being transferred, thus indirectly reducing the need for higher bandwidth in communication channels. Extensive research is needed to generate a tradeoff between these techniques and the volume of bandwidth required to attain ideal system performance.

### **6.5.4 SELECTING CLIENT DEVICES**

In Federated Learning (FL) training, clients play a significant role as the training takes place on-site, and the availability of clients is a crucial factor in determining the quality of the model generator. Along with availability, other factors such as device status, power level, computational capacity, communication capabilities, and data set quality are associated with client devices, which also impact the performance of the final generated model.

Client selection not only affects the generation of the FL model but also plays a major role in determining the detection efficiency of the Intrusion Detection System (IDS). In a typical FL training scenario, some devices may not be present at certain points during training due to factors like device mobility or power shortage. Such device unavailability automatically leads to delays in FL model convergence. When integrating FL with IDS, it also affects the performance of the IDS by increasing the time required to isolate anomalies and perform efficient intrusion detection. Therefore, better strategies and protocols need to be devised for client selection and improving

system performance, even when the number of uncooperative clients increases in such systems.

#### **6.5.5 DYNAMIC NATURE OF HOST DEVICES**

IoT devices are generally highly dynamic and undergo high levels of changes in various aspects throughout their lifetime. Such changes pose serious issues to FL-based IDS systems as the systems need to be constantly updated to keep up with the changes in devices. In such systems, a major issue is that similar changes can occur due to certain adversarial attacks as well. So correctly classifying each scenario as adversarial or benign is a highly relevant task. Additionally, in heterogeneous environments, changes occurring in a single device cannot be isolated and can affect other devices of the system as well, impacting the total performance of the system. Researchers state that the federated counterpart of reinforcement learning can be applied as an effective solution for the problems associated with dynamic environments. However, such an approach is yet to be introduced in FL-based IDS, thus still making this a possible research direction.

#### **6.5.6 ADVERSARIAL ATTACKS**

As in centralized learning approaches, federated learning is also prone to certain adversarial attacks that can hinder the learning procedures. According to current literature [29], some of the major security issues associated with FL are poisoning attacks, including data poisoning and model poisoning. In data poisoning-based attacks, corrupted data is added to systems, which can open up back doors or facilitate other successful attacks such as label flipping attacks. Model poisoning attacks involve altering the models to perform tasks according to the adversary's intent. These attacks often confuse IDS, leading to high levels of misclassification [25]. Certain security measures have been proposed to mitigate such attack scenarios. In a similar context, article [12] presents a detailed comparison of changing behavioral patterns in aggregation functions during adversarial attacks for systems with FL-based IDS. The article also identifies the applicability of specific aggregation protocols as ideal alternatives for particular adversarial attack scenarios. Additionally, the applicability of trust and reputation-based mechanisms can

be considered as active deterrents for such security threats, in addition to lightweight cryptographic measures.

#### **6.5.7 PRIVACY ISSUES**

Even though FL was initially proposed as a methodology to overcome the privacy concerns associated with traditional learning systems, the methodology has yet to be called fully secure as it is still prone to certain levels of privacy breaches and adversarial attacks. One of the common security issues associated with FL is the presence of a malicious server that has the complete capability of performing intended data leakage attacks [29]. Privacy-preserving techniques such as differential privacy, secure multiparty computation, homomorphic encryption, and so on can limit the extent of privacy breaches, but they also have certain limitations. All these limitations of FL negatively affect the efficiency of IDS systems as well. While current literature has proposed methodologies to overcome these shortcomings [30], further studies are required to develop a trade-off between the system performance and accuracy when it comes to the effective development of FL-based IDS.

### **6.6 CONCLUSION**

Federated learning has been under the limelight of research for the past few years, citing its advantages when compared to traditional learning methodologies. Through this article, we aim to provide a detailed overview of various literature presenting the integration of FL with intrusion detection systems. The utilization of federated learning in intrusion detection systems produces optimal solutions with high-quality models, ensuring enhanced user privacy. The article considers various aspects of FL and IDS, including different classifications, data distribution patterns, architectural patterns, and more. A comprehensive analysis of several articles attempting to implement FL-based IDS is thoroughly evaluated. Additionally, various use cases related to different types of attacks and implementation scenarios, such as IoT networks, are also presented in detail. The study discusses some of the challenges and future prospects for implementing IDS based on FL. We also address issues related to system latency, false alarms, various

adversarial attacks, and more. The study concludes by presenting possible research directions for addressing the aforementioned challenges in IDS.

## KEYWORDS

- **adversarial attacks**
- **distributed denial of service attacks**
- **federated learning**
- **internet of things**
- **intrusion detection systems**
- **privacy issues**
- **traditional learning**

## REFERENCES

1. Hassan, M. M., Abdu, G., Ahmed, A., Majed, A., & Giancarlo, F., (2020). A hybrid deep learning model for efficient intrusion detection in the big data environment. *Information Sciences*, 513, 386–396.
2. Khraisat, A., Iqbal, G., Peter, V., & Joarder, K., (2019). Survey of intrusion detection systems: Techniques, datasets, and challenges. *Cyber security*, 2(1), 1–22.
3. Khraisat, A., & Ammar, A., (2021). A critical review of intrusion detection systems in the internet of things: Techniques, deployment strategy, validation strategy, attacks, public datasets, and challenges. *Cyber security*, 4(1), 1–27.
4. Liu, M., Zhi, X., Xianghua, X., Changmin, Z., & Jinjun, C., (2018). Host-based intrusion detection system with system calls: Review and future trends. *ACM Computing Surveys (CSUR)*, 51(5), 1–36.
5. Pawlicki, M., Michał, C., & Rafał, K., (2020). Defending network intrusion detection systems against adversarial evasion attacks. *Future Generation Computer Systems*, 110, 148–154.
6. Masdari, M., & Hemn, K., (2020). A survey and taxonomy of the fuzzy signature-based intrusion detection systems. *Applied Soft Computing*, 92, 106301.
7. Aldweesh, A., Abdelouahid, D., & Ahmed, Z. E., (2020). Deep learning approaches for anomaly-based intrusion detection systems: A survey, taxonomy, and open issues. *Knowledge-Based Systems*, 189, 105124.
8. Wang, Y., Weizhi, M., Wenjuan, L., Zhe, L., Yang, L., & Hanxiao, X., (2019). Adaptive machine learning-based alarm reduction via edge computing for distributed intrusion detection systems. *Concurrency and Computation: Practice and Experience*, 31(19), e5101.

9. Ahmim, A., Leandros, M., Mohamed, A. F., Makhlouf, D., & Helge, J., (2019). A novel hierarchical intrusion detection system based on decision tree and rules-based models. In: *2019 15<sup>th</sup> International Conference on Distributed Computing in Sensor Systems (DCOSS)* (pp. 228–233). IEEE.
10. Gao, B., Bing, B., Wei, Z., & Xiang, L., (2021). An intrusion detection method based on machine learning and state observer for train-ground communication systems. *IEEE Transactions on Intelligent Transportation Systems*.
11. Chih-Che, S., Jonathan, S. C. D., Adam, H., & Chen-Ching, L., (2020). Intrusion detection for cyber security of smart meters. *IEEE Transactions on Smart Grid*, *12*(1), 612–622.
12. Kukkala, V. K., Sooryaa, V. T., & Sudeep, P., (2020). Indra: Intrusion detection using recurrent auto encoders in automotive embedded systems. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, *39*(11), 3698–3710.
13. Upadhyay, D., Jaume, M., Marzia, Z., & Srinivas, S., (2020). Gradient boosting feature selection with machine learning classifiers for intrusion detection on power grids. *IEEE Transactions on Network and Service Management*, *18*(1), 1104–1116.
14. Tian, Z., Chaochao, L., Jing, Q., Xiaojiang, D., & Mohsen, G., (2019). A distributed deep learning system for web attack detection on edge devices. *IEEE Transactions on Industrial Informatics*, *16*(3), 1963–1971.
15. Farivar, F., Mohammad, S. H., Alireza, J., & Mamoun, A., (2019). Artificial intelligence for detection, estimation, and compensation of malicious attacks in nonlinear cyber-physical systems and industrial IoT. *IEEE Transactions on Industrial Informatics*, *16*(4), 2716–2725.
16. Ferdowsi, A., & Walid, S., (2019). Generative adversarial networks for distributed intrusion detection in the internet of things. In: *2019 IEEE Global Communications Conference (GLOBECOM)* (pp. 1–6). IEEE.
17. Rey, V., Pedro, M. S. S., Alberto, H. C., & Gérôme, B., (2022). Federated learning for malware detection in IoT devices. *Computer Networks*, *204*, 108693.
18. Rahman, S. A., Hanine, T., Chamseddine, T., & Azzam, M., (2020). Internet of things intrusion detection: Centralized, on-device, or federated learning? *IEEE Network*, *34*(6), 310–317.
19. Li, H., & Han, T., (2019). *An End-to-End Encrypted Neural Network for Gradient Updates Transmission in Federated Learning*. arXiv preprint arXiv:1908.08340.
20. Li, W., Weizhi, M., & Man, H. A., (2020). Enhancing collaborative intrusion detection via disagreement-based semi-supervised learning in IoT environments. *Journal of Network and Computer Applications*, *161*, 102631.
21. Briggs, C., Zhong, F., & Peter, A., (2020). Federated learning with hierarchical clustering of local updates to improve training on non-IID data. In: *2020 International Joint Conference on Neural Networks (IJCNN)* (pp. 1–9). IEEE.
22. Zhang, C., Paul, P., & Hamed, H., (2019). Deep learning in mobile and wireless networking: A survey. *IEEE Communications Surveys & Tutorials*, *21*(3), 2224–2287.
23. Doriguzzi-Corin, R., & Domenico, S., (2022). *FLAD: Adaptive Federated Learning for DDoS Attack Detection*. arXiv preprint arXiv:2205.06661.
24. Li, J., Lingjuan, L., Ximeng, L., Xuyun, Z., & Xixiang, L., (2021). FLEAM: A federated learning empowered architecture to mitigate DDoS in industrial IoT. *IEEE Transactions on Industrial Informatics*, *18*(6), 4059–4068.

25. Nguyen, T. D., Phillip, R., Markus, M., & Ahmad-Reza, S., (2020). Poisoning attacks on federated learning-based IoT intrusion detection system. In: *Proc. Workshop Decentralized IoT Syst. Secur. (DISS)* (pp. 1–7).
26. Anastasakis, Z., Konstantinos, P., Terpsi, V., Stavroula, B., Artemis, V., Dimitrios, S., Antonis, G., & Theodore, Z., (2022). Enhancing cyber security in IoT systems using FL-based IDS with differential privacy. In: *2022 Global Information Infrastructure and Networking Symposium (GIIS)* (pp. 30–34). IEEE.
27. Zhao, R., Yijun, W., Zhi, X., Tomoaki, O., Bamidele, A., & Guan, G., (2022). Semi-supervised federated learning-based intrusion detection method for the internet of things. *IEEE Internet of Things Journal*.
28. Chen, M., Nir, S., Vincent, P. H., Yonina, C. E., & Shuguang, C., (2021). Communication-efficient federated learning. *Proceedings of the National Academy of Sciences*, 118(17), e2024789118.
29. Nair, A. K., Ebin, D. R., & Jayakrushna, S., (2023). A robust analysis of adversarial attacks on federated learning environments. *Computer Standards & Interfaces*, 103723.
30. Agrawal, S., Sagnik, S., Ons, A., Gokul, Y., Kandaraj, P., Mamoun, A., Sweta, B., et al., (2022). Federated learning for intrusion detection system: Concepts, challenges, and future directions. *Computer Communications*.

## CHAPTER 7

---

# Exploring Communication Efficient Strategies in Federated Learning Systems

AKARSH K. NAIR, JAYAKRUSHNA SAHOO, and EBIN DENI RAJ

*Department of Computer Science and Engineering, Indian Institute of Information Technology, Kottayam, Kerala, India*

---

### ABSTRACT

Federated learning is a distributed learning paradigm that enables various lightweight devices in distributed networks, such as edge devices, to collaboratively generate a machine learning model over a certain number of iterations using their in-house data. Due to the iterative nature of model training, the FL system requires an excessive number of gradient transfer operations among client devices and the server system for the successful generation of ML models. In certain cases, the FL system requires the transfer of a large number of parameters based on the complexity of the model. In reality, model training occurs on lightweight devices with limited communication bandwidth capabilities. When multiple client devices attempt to transfer multiple parameters simultaneously, it puts a significant load on communication channels, leading to communication bottlenecks. This common scenario in FL systems greatly degrades the model generation capabilities and affects the system's performance. To address the issue of communication delays, various methodologies have been proposed in this study. We will be presenting some common strategies in FL systems to achieve communication efficiency, such as data compression, distillation, sparsification, and more. The study will also discuss various aspects of individual methodologies, including their merits and demerits.

## 7.1 INTRODUCTION

The rapid evolution in deep learning (DL) based techniques has been a highly beneficial technological venture for humans in several domains, such as smart healthcare, autonomous systems, personalized systems, and more. In the majority of use cases, the quality of the DL models being generated is determined by the availability of excessive and usable data. As a result, most DL-based applications rely solely on centralized data silos for storing training data. However, certain data, such as healthcare and medical data, tend to be private due to their origin. With the growing awareness of privacy preservation, several regulations, such as GDPR, have been implemented in various AI approaches to ensure data privacy. Search factors have posed a major challenge in collecting and aggregating data from various sources into a single destination for performing AI-based tasks.

Federated learning is an emerging approach where models are learned from private data, which has found huge applicability in several applications involving privacy-sensitive data such as healthcare and finance. Federated learning offers a secure and private learning platform where raw data transfer is avoided, and clients collaboratively generate global models via local gradient transfer and aggregation procedures only. Instead of collecting and aggregating data at a centralized point, Federated learning systems have their private data stored over multiple client devices. Each individual client keeps a version of their local data for computing model updates during system training. In each iteration, a certain number of clients initially generate their local gradients and transfer them to the central server, where they are aggregated and the global model is formed. The server then broadcasts the global model to clients again for performing another iteration, and the cycle continues for multiple rounds until the system converges. As the gradient data contains very little information regarding the actual data, it offers higher levels of privacy compared to standard centralized learning approaches.

Nonetheless, implementing federated learning for real-life applications is still a major challenge, mainly due to the difficulties of its distributed operational nature. The existing technologies and systems are accustomed to the traditional approach, and such rapid changes make it highly difficult for existing systems to adapt to it. Some of the common challenges include communication overhead, device heterogeneity, data heterogeneity, and so on. When it comes to device heterogeneity, identical devices have varying specifications, resulting in different computation environments,

thus causing disparity in their time and methods of model training. Similar to data heterogeneity, various systems comprise different datasets, thus causing heterogeneity in terms of data type as well as distribution. Among these aforementioned challenges, communication overhead is a major issue acting as a major hurdle for the development of the technology. Since Federated Learning (FL) models are generated over multiple iterations through continuous gradient transfer operations, the server and clients must engage in intense communication throughout the training process. As a result, the communication cost increases exponentially with the complexity of the model being trained. When dealing with deep learning (DL) models, the current state of the model tends to be highly complex, with millions of parameters. Consequently, transmitting such complex models among devices becomes impractical in federated environments, particularly due to communication bandwidth limitations imposed by lightweight clients. This limitation restricts the applicability of large models in conventional FL environments. Additionally, the range of participating devices contributes to the communication bottleneck. Real-life FL systems often involve a large number of devices with varying degrees of heterogeneity, resulting in differences in training time, training data, and communication capabilities. This discrepancy causes transmission delays during gradient transfer operations, further exacerbating the overall communication issues in FL systems.

Current research interest in FL has been centered on optimizing communication channels and achieving communication efficiency. One of the potential suggestions is to employ gradient compression for local model gradients. However, such a method leads to data loss during high levels of compression, thus requiring sacrificing system performance. Additionally, compressed gradient updates also result in less degraded global models, as models tend to lose their ability to work with heterogeneous data with compression. Another strategy employed for achieving communication efficiency is referred to as distillation. In distillation-based approaches, rather than transferring model updates, the system only transfers local model predictions on a public dataset which will already be distributed between participating client devices. Such a method can reduce the overall communication costs for instances with model size overtaking the public dataset size. Such an approach has certain limitations due to the privacy aspect. Most of the real-world scenarios with FL applications, such as personalized systems and healthcare systems, have data with high privacy requirements, which makes them unsuitable. As a result, a strategy offering both communication efficiency as well as system privacy is yet to be devised.

The rest of the study is divided into six sections. Section two presents the aspect of communication efficiency in FL systems, followed by communication scenarios in section three. In section four, we discuss the various resource allocation strategies, and section five comprises the overall discussion of the articles and future scope for further research. The study is concluded in section six with a conclusion.

## **7.2 COMMUNICATION EFFICIENCY**

In the current scenario, three different methodologies have been applied to develop a communication efficiency framework for FL systems. These methodologies are quantization-based methods, sparsification-based methods, and distillation-based methods. In this section, we will discuss each of these approaches in detail, presenting their respective subcategories as well.

### **7.2.1 QUANTIZATION-BASED APPROACHES**

By using the term quantization, it simply means to reduce the total size of data points. It is usually achieved by converting a set of points from a higher size to a lower size, sacrificing precision but not compromising system performance. Especially in the context of Federated Learning (FL), most quantization approaches aim to compress the volume of gradients transferred from individual clients to the server. In typical FL systems, these gradients are transferred in a contiguous manner. However, for quantization purposes, they are broken down into smaller fragments through discretization of each client gradient, thereby reducing the size of the representation. In the following sections, we will present some widely employed quantization approaches in the context of FL.

#### **7.2.1.1 STOCHASTIC QUANTIZATION**

Stochastic quantization (SQ) is one of the basic quantization methodologies initially proposed by Alistrah et al. [1]. It is built on the classical stochastic gradient approach (SGD) and performs gradient quantization via a novel method termed Quantized stochastic gradient descent (QSGD). The purpose of the proposed model is to optimize the data transmission problem in parallel SGD computing and attempts to develop a trade-off between communication

bandwidth and system convergence time. In QSGD, hyper parameter tuning is employed for performing optimization alongside ensuring minimal variance in the system. Keeping QSGD as a reference, several updated versions were developed [27-32]. For instance, a quantization approach based on partial client selection was proposed by Reisizadeh et al. [2] referred to as FedPAQ. The idea behind the proposed model was that a limited number of clients considerably reduced the total volume of data being transferred, thus achieving quantization during model aggregation. An enhanced version of FedPAQ was proposed, termed as FedCOM [3], where the authors focused on making the model applicable for both heterogeneous and homogeneous settings. Article [4] introduced a quantization-based FL framework called FedGLOMO, with the main goal of reducing the variance among updates generated by various clients through the use of momentum-based global aggregation techniques. A similar article was presented by Dai et al. [5], proposing a new quantization approach referred to as hyper-sphere quantization (HSQ). The objective was to generate an efficient global model by employing the SQ quantizer to reduce communication costs during local gradient transfer.

Focusing on the factor that the quantization level could not be managed dynamically, an enhanced version was proposed by Jhunjhunwala et al. [6], termed AdaQuantFL, which operates on an adaptive quantization strategy. It considers a trade-off between error and communication bits and also allows clients to automatically adjust the quantization levels of QSGS during training time. A lossy function-based FL framework with a similar intention is proposed in the article [7] to quantize the global model as well as local gradients simultaneously, aiming to reduce communication costs. The article differentiates itself from its predecessors as it does not rely on streamlined global model transfer methods and also addresses faults at the server end.

#### 7.2.1.2 ROTATION-BASED QUANTIZATION

Even though the stochastic approach was an easy and effective mode of quantization, it was ideally suited for frameworks where individual vector coordinates need to be assigned for a finite set of possibilities. Extensive research has shown the inability of SQ models to perform well in applications where the distribution of vectors is usually very wide, with the largest and smallest values having two extreme positions away from one another. Thus, a high volume of research has been conducted attempting to solve such problems referred to as the Distributed Mean Estimation (DEM) problem. An

initial article on the same was proposed by Suresh et al. [8], where a biased and deterministic quantization framework was developed by performing random rotation before employing the standard quantization techniques. Inspired by the same, a biased and unbiased compression technique was proposed by Vargaftik et al. [9]. The proposed model was termed DRIVE, and it was performed by introducing random rotation into the DEM framework for quantizing the initial vector into a one-bit quantization level. The authors further extended the work in the article [10] by proposing a newer framework called EDEN. In contrast to [11], where the framework was based on discrete quantization, the proposed model was built on an interval-based quantization approach, where individual coordinates were quantized to their nearest center. As a result, the model significantly reduces the entropy of the quantized vectors and achieves efficient estimation within a preset communication budget. However, in earlier articles, each client had an individual rotation matrix independent of other devices, which increased the processing time for the server. This was because the server needed to rotate the vector back to its normal state before working on it in each iteration. Taking this issue as motivation, Basat et al. [12] proposed a framework where the same matrix could be used, thus speeding up the model aggregation process. The major advantage of the proposed model was that only a single inversion operation was required at the server end before aggregation.

#### *7.2.1.3 LATTICE-BASED QUANTIZATION*

The initial article on lattice-based quantization was proposed by Zamir and Feder in [13]. The authors propose optimizing every single point to the closest lattice, which is a better approach compared to optimal vector quantizers. The introduction of lattice-based quantization into the Federated learning context was done by Shlezinger et al. [14]. The authors later enhanced their work by also proposing a universal vector quantization approach specifically built for the FL system referred to as UVeQFed. Major factors taken into consideration for the framework were rate-constrained channels and also generating a solution for the throughput-constrained uplink problem via the application of subtractive dithered lattice quantization with minimum guaranteed distortion. As part of further optimization, Chen et al. [15] also proposed an article where lattice quantization was achieved via client selection. The selection criterion was based on their chances of connecting to the server and also fine-tuning the bandwidth for minimizing transmission delay and optimizing channel usage as well.

#### 7.2.1.4 SINGLE BIT QUANTIZATION (SIGN BASED)

In existing literature, the transferred gradients were compressed to very high limits, usually up to 4 bits, which reduced their precision considerably. A different approach, built on single-bit quantization methods, was proposed by Bernstein et al. [16]. The proposed model, referred to as signSGD, represents the initial gradients via sign while maintaining a high level of system convergence rate and testing accuracy. Another similar framework, sto-signSGD, is presented in article [17], where the authors attempt to represent the quantization value of the gradient in a random variable form, which itself is a dynamic value changing with respect to the sign of the quantization result. The system also incorporates a voting mechanism to ensure the robustness of the proposed framework under highly heterogeneous environments. An application of the signSGD model on an edge computing-based system is presented by Zhu et al. [18]. The proposed framework also operates on a majority voting-based system, along with a single-bit gradient quantization approach, to achieve communication efficiency.

#### 7.2.1.5 COMPRESSION SENSING-BASED QUANTIZATION

In certain signals, there persists a scarcity of data which can be utilized to effectively implement quantization. Compression sensing (CS) is a combination of classical sampling and compression techniques employed in conventional signal processing approaches, leveraging the scarcity of signals to compress and reduce the total volume of data samples required for effective signal recovery. This methodology employs matrix transformations to obtain a sparse representation of the initial signal without gaps, followed by reconstruction to generate the initial signal from the gathered data information by solving an optimization problem. Abdi and Fekri proposed initial work on CS as a quantization approach for distributed learning systems in [19]. The article focuses on achieving high levels of compression for problems related to gradient variance and optimizing convergence rate during quantization. Li et al. [20] and Fan et al. [21] presented other articles on single-bit CS-based quantization, both utilizing iterative hard thresholding for gradient reconstruction. Similar to previous cases of single-bit quantization, the issue of quantization error persisted in these systems. Oh et al. [22] proposed a multi-tier quantization method called Q-EM-GAMP as a solution to these issues. The method utilizes expectation maximization principles to perform efficient reconstruction, significantly reducing error.

### 7.2.1.6 OTHER METHODOLOGIES

Other than the previously mentioned approaches, several methodologies are applied for performing quantization. One such approach was presented by He et al. [49], where they proposed a methodology called CosSGD, which performs quantization on the angle vector based on the gradients using cosine functions to limit the range. Another method, proposed in Article [23], is called FedZip, which utilizes classical K-means clustering, top-Z sparsification, and Huffman encoding for compression. In the context of non-iid data, the Artemis framework is proposed in [24], which compresses local gradients and values using the s-quantization approach. The article also mentions the need to optimize the inbuilt memory mechanism for improving model convergence in non-IID data scenarios. Chen et al. [25] proposed a quantized aggregation strategy called FedHQ, where the weight distribution strategy is altered based on the upper bound of individual clients. Cui et al. [26] presented a novel algorithm named MUCSC, which applies classical soft computing methods as a compression strategy. This method can also be implemented as a solution for networks with resource scarcity problems.

### 7.2.2 SPARSIFICATION-BASED APPROACHES

Sparsification approaches convert complete gradient data into sparse content, consisting of parameters with higher significance only, while another subset of coordinates with lesser significance is initialized to zero. In the context of federated learning, top-k and rand-k are two of the most widely employed sparsification approaches. In top-k sparsification, the subset of elements generated comprises k-percent of the sparsification target, possessing the highest absolute values. On the other hand, in the rand-k method, the values are selected randomly. Although rand-k is referred to as an unbiased sparsification approach, the results often have a higher number of compression errors, making it less applicable for real-life applications compared to the top-k sparsification approach. In general, it can be said that the application of sparsification techniques functions more radically and offers better communication load reduction when compared to quantization-based approaches. For instance, the top-k sparsification approach offers very high accuracy and convergence rate, even when more than 90% of the gradients can be eliminated.

In comparison with sparsification in the context of traditional learning, sparsification in FL takes into consideration various factors such as

optimizing computation, memory management, achieving communication efficiency in federated training scenarios, and so on. Thus, current research mainly concentrates on optimizing existing sparsification approaches such as top-k methods for providing better service delivery for tasks associated with efficient communication. Although several sparsification approaches have been implemented in the context of traditional learning, the scope of this article limits us to select a few methods that have found applicability in the context of FL.

#### 7.2.2.1 ADAPTIVE SPARSIFICATION

Even though the top-k sparsification approach has proven to be an ideal technique for achieving optimality in terms of data compression and communication for a given instance with  $k$  elements, it operates on an iterative basis. Therefore, Sahu et al. [27] proposed a different sparsification approach based on a hard threshold criterion, aiming to achieve optimality for training in a global manner rather than an iterative one. In the proposed model, the sparsity degree is determined adaptively with respect to a hard threshold value, and the model optimizes the compression loss, demonstrating its applicability as an efficient alternative to the existing sparsification method [27]. A similar article has been proposed in [28], where the adaptive nature of the model in determining the sparsity degree  $k$  is utilized to further minimize the total training time. The optimality is achieved by finding a balance point for the sparsity degree where both communication and computational optimality can be attained.

Most of the literature on adaptive sparsification mainly focuses on adjusting the level of system sparsity and adapting the system requirements based on other FL system parameters such as iteration counts and partial client participation. Sattler et al. [29] presented a study analyzing communication delays and various error accumulation techniques in FL systems. The rationale behind combining these techniques is that both methods compress data by introducing a delay in update transfer and accumulating gradient information before the actual transfer. Based on this observation, the proposed model, the sparse binary compression approach, efficiently balances temporal and gradient sparsity. Another approach that considers both local updates and sparsity values was proposed by Nori et al. [30], optimizing these factors to enhance system performance and creating a framework known as Fast FL. Article [31] also introduces an adaptive compression-based approach that achieves a better trade-off between training speed and system accuracy by

considering two major factors: network delays and compression control. The proposed methods easily outperform both of the well-known sparsification methods, top-k and rand-k. Li et al. [32] proposed an approach referred to as the general gradient sparsification approach, which aims to mitigate issues related to the delay of updates caused by communication channel problems. The approach mainly accumulates gradient data of lower significance and transfers them to the adaptive optimizers. After that, the client updates are updated via the batch normalization (BN) layer as well. The proposed model overcomes issues caused by delays in the delivery of model gradients and thus optimizes the system performance while achieving gradient sparsity. Another approach referred to as linear FetchSGD is presented by Rothchild et al. [33], where the author attempts to accumulate the errors that occurred at the client end during training and shift them to the central server, thus eliminating the requirement for local client states. The prominent issues associated with partial clear participation in FL training can be easily mitigated by employing the FetchSGD approach in FL systems.

#### *7.2.2.2 BIDIRECTIONAL SPARSIFICATION*

The proposed methods easily outperform both of the well-known sparsification methods, top-k and rand-k. Li et al. [32] proposed an approach referred to as the general gradient sparsification approach, which aims to mitigate issues related to the delay of updates caused by communication channel problems. The approach mainly accumulates gradient data of lower significance and transfers them to the adaptive optimizers. After that, the client updates are updated via the batch normalization (BN) layer as well. The proposed model overcomes issues caused by delays in the delivery of model gradients and thus optimizes the system performance while achieving gradient sparsity. Another approach referred to as linear FetchSGD is presented by Rothchild et al. [33], where the author attempts to accumulate the errors that occurred at the client end during training and shift them to the central server, thus eliminating the requirement for local client states. The prominent issues associated with partial clear participation in FL training can be easily mitigated by employing the FetchSGD approach in FL systems. The proposed compressor, known as the boom filter-based index compressor, is capable of reducing up to 50% of the total transmission data when compared to existing raw data sparse representation methods. Additionally, it is recommended to cap the transmission of new non-zero positions for communication optimization. The proposed method utilizes a novel approach called time-correlation sparsification (TCS), where

the correlation between nearby sparse representations is used to reduce the transmission of nonzero data index values during FL training. The proposed model provides better compression for downstream data transfer compared to upstream data.

#### 7.2.2.3 LAYER-WISE SPARSIFICATION WITH PIPELINE

In an attempt to achieve better communication efficiency in FL systems by accelerating the training of the entire system, certain methods have been proposed to utilize the model structure for achieving parallelism in terms of communication with computations. This approach is referred to as pipeline. Shi et al. [37] proposed a layered adaptive gradient specification approach that combines the conventional gradient pacification methodology with the concept of a pipeline. In the proposed model, the subset of values of each individual layer is considered independently based on a given ratio. However, a significant issue with this approach is the requirement for high levels of communication, as each individual layer has its own sparsified gradients that require individual communication setup, which adds to the cumulative total [37]. Thus, one alternative for alleviating such overloads is by merging the gradients of individual layers for communication purposes. However, this approach introduces computation overload, thus increasing the total processing time. Therefore, in an attempt to generate a trade-off between gradient computations and layer-wise sparsification, the optimal merged gradient specification approach was proposed in [38]. The proposed method considers trade-off as an optimization problem, thus minimizing the training time by maximizing the overlap between sparsified gradient computation and communication during model generation.

#### 7.2.3 KNOWLEDGE DISTILLATION

In the context of FL systems, knowledge distillation (KD) is mainly employed as a solution for bottleneck issues associated with communication channels. To reduce the size of models, KD employs a transfer mechanism-based approach where information from a large model, usually referred to as a mentor, is transferred to a small model which acts as a mentee. This transfer ensures that the model's performance in terms of precision or convergence is not affected by any means. Thus, the superior network is comparatively a larger network with a huge volume of parameters and a sophisticated

network structure, possessing high levels of performance and generalization capability. On the other hand, the mentee network is less complex, having a lesser number of parameters. As many of the student models possess identical shape sizes, making the output also identical, it can be inferred that the output data is irrelevant to the structure of the mentee model. This makes it possible for KD to address issues associated with model heterogeneity by just uploading the final result of individual models rather than the complete model itself, and such a procedure is referred to as federated distillation (FD). Through this study, a short discussion on the various classifications of FD concerning the dataset uploaded by the client is discussed. Based on that criterion, FD approaches are divided into data-additional FD approaches and data-free FD approaches.

#### ***7.2.3.1 DATA ADDITIONAL FD APPROACHES***

The initial proposal of FedMD was made by Li and Wang [39], where the proposed model initially trained the local model based on the public dataset available to the devices. It then uploaded the generated class scores only and refrained from transferring the model parameters. On the server side, it integrates the obtained data received from all the client devices. As the model training under the FedMD approach happens simultaneously on both types of data, i.e., labeled and private, there is a necessity for a considerable amount of communicational resources, making it least suitable for resource-constrained environments. Additionally, generating public datasets is a tedious task as it requires certain procedures and thus lacks generalization ability. Another article based on FedDF was also proposed by Lin et al. [40], mainly for generating global models via unlabeled datasets using an ensemble approach. It also proved the robustness of FedDF towards the training dataset, irrespective of the selection and combination patterns. The liberty to choose the auxiliary dataset is entrusted to the server, considerably reducing the computational load of the client devices as well. As the publicly available dataset is usually unlabeled, the privacy compromised in FedDF via leakage is also much less than in the FedMD approach. Another FD-based approach referred to as compressed federated distillation (CFD) is proposed by Sattler et al. [41], where features such as “entropy” and “margin” are taken into consideration as standard measures for determining the volume of the public dataset for distillation. They also employ a uniform quantization approach to reducing communication costs as well. The initial works on Federated Learning (FL) mainly focused on optimizing communication

costs by altering the data offset size or the logistic vectors, while completely ignoring updates to the aggregation approach. In article [42], an approach called DSFL is proposed, which aims to perform aggregation on local models through an entropy reduction approach. As data heterogeneity is a challenge in FL systems, the entropy obtained for global logits is also high, indicating difficulties in generating an efficient model using a non-IID data distribution scenario. The proposed model reduces the entropy of the global logits by adding a temperature factor to the SoftMax function, making the system more stable and fast. A similar article is proposed by Sattler et al. [43], where the proposed model is referred to as FEDAUX. The idea of the model is to calculate the probability of the assured occurrence of each client's logits through contrastive logistic scoring and use the value to determine the aggregation weights required to generate efficient results for data with high differentiation.

Another adaptive aggregation strategy, termed pFEdSD, was proposed by Li et al. [44] specifically for achieving dynamicity in modifying individual participants' training weight and also for improvement in global models' performance. The Jensen-Shannon divergence approach is used to identify the measure of divergence between the outputs generated from the local models at two adjacent rounds and thus the overall similarity of the models. Sturluson et al. [45] also presented a similar adaptive aggregator-based approach named FedRAD. The proposed model employs a median score to reinforce the global model. Liu et al. [46] proposed an approach to address concerns over the combination of public dataset selection strategies along with non-trivial aggregation methods. The proposed model is specifically designed to solve non-IID data distribution problems by employing weighted logistic-based dissimilarity of clients' data and then adding differential privacy to the aggregation algorithm for privacy preservation of local data.

#### 7.2.3.2 DATA FREE FD

Even though the additional FD approach in the data can considerably reduce communication overheads by transferring the value of the logits function, it requires client devices to transfer a portion of their local data for constructing the public dataset. Such an approach is not acceptable for FL systems as these data transfer operations compromise the privacy of users. Therefore, data-free FD approaches have been proposed as an alternative to the existing approaches, eliminating the need for data transfer and preserving privacy and ensuring security for client data. The initial work on data-free FD was

proposed by Jeong et al. [47] to handle non-IID data distribution problems. In the proposed model, logits are periodically sent to the server for averaging and the mentee receives the aggregated logits of the global models as the mentor's output to assist in model training. However, the article does not discuss the aspect of model heterogeneity. To address this, Jiang et al. [48] presented FedDistill, an approach that forcefully makes the client share their global model with an identical structure while training on another customized local model. In such systems, a client device possesses two different models: a customized complex model, usually acting as a mentor model, generated through training on local data, and a less complex model received directly from the server device. In each iteration, mentee models are transferred to the server and aggregated, which still allows for the possibility of inferring individual client information by interpreting the model structures. Inspired by FedDistil, Wu et al. [49] proposed another framework referred to as FedKD, which includes three separate loss functions to optimize global model training. They also introduced matrix factorization approaches to reduce communication overheads. In this approach, the KL divergence is used to adaptively infer details about mentor and mentee models. However, directly aggregating the global model still has performance issues due to system and data heterogeneity. Therefore, better alternatives are required to ensure system performance. FedGEN is proposed in [50] to address this issue by training a generator on the server side based on the classifier data received from client devices. In each iteration, the label count of the local device is notified to the server device, and the local training is regularized once the global lightweight generator is received. Various models built on these concepts are presented, such as FedGKD [51] and FedFTG [52]. FedGKD uses the average parameters of the global model for ensembling, which helps mitigate the client drift problem caused by non-IID data distribution. FedFTG initially generates synthetic data to train the model, and then real data samples are used to train the global model simultaneously.

#### 7.2.3.3 APPLICATION OF KNOWLEDGE DISTILLATION

Due to the mentor-mentee relationship-based architecture, KD is widely employed in applications that have client heterogeneity-related issues, making it applicable for personalized FL systems. Another KD-based FL framework was proposed by Cho et al. [53], which was built on a weighted consensus-based distillation approach along with diversity regularization for extracting model information from heterogeneous environments. Zhu et al.

[54] also proposed a personalized FL framework called Fed Rescue, which generates several sub-models by performing pruning on the global model based on the pruning sequence. Thus, each client device individually selects the pruning weight, closing its local model and receiving a simplified version of the model instead of the complex global model. A KD-based knowledge agnostic framework, FedZKT, is proposed by Zhang et al. [55], where the proposed model functions based on a customized generator built on the local model structure of individual clients and strives to achieve communication efficiency. Ozkara also presented a personalized KD approach, QuPeD, that applies a soft quantization approach to solve optimization problems and utilizes KD to tackle issues associated with resource heterogeneity. Due to its highly desirable features related to feasibility and compression ability, KD has found wide employability in most FL systems. For example, in FL systems employing differential privacy and secure multiparty computation, KD can be employed as a compression mode, as both of these privacy-preserving approaches are renowned for causing high levels of communication and computational overheads. Thus, several articles are being presented where KD is proposed as an effective compression approach for systems with secure computation requirements. Moreover, KD is also highly applicable in edge-based and IoT environments for bringing in communication cost-reduction features.

#### **7.2.4 OTHER METHODOLOGIES**

Other than the three technologies discussed, other nonconventional methods have also been employed in FL to ensure communication efficiency during model training and functioning. Some of the methods are as follows:

##### **7.2.4.1 FASTER MODEL CONVERGENCE**

The initial article on federated learning employed FedAvg as a global model aggregation, where multiple gradients from individual clients were received and an aggregation was performed on them using an averaging approach to generate the global model. In this approach, the aggregation was performed after several iterations, thus reducing the frequency of communication in such systems. An updated version of the FedAvg algorithm was proposed by Li et al. [56], termed as q-FedAvg, which dynamically selected a subset of clients and achieved a faster convergence rate in terms of the total number

of communication rounds. Several other articles are available proposing various approaches, such as low-rank Hadamard product parameterization and employing compressed model updates for heterogeneous systems, to achieve faster convergence.

#### ***7.2.4.2 SYSTEM ARCHITECTURE***

Unlike the classical client-server architecture employed in centralized FL systems, various other network topologies can be used, especially in networks based on cross-silo architecture. Marfoq et al. [57] claimed the efficiency of high-speed channels for information exchange in cross-silo-based systems and also introduced a network topology for optimizing cyclic time problems and obtaining the largest value of throughput. Article [58] proposed a client clustering-based approach through HL-SGD, which utilized the inter-device communication capabilities inside the FL system to generate clusters and then speed up the model convergence without compromising system performance by utilizing the high-speed inter-device communication channels for data transfer.

#### ***7.2.4.3 COMPARISON AND DISCUSSIONS***

In the above-mentioned sections, we present three main communication efficient strategies employed in FL systems. Each approach presents a set of advantages along with a set of disadvantages as well. Regarding the quantization approaches, time-based quantization is the most explained approach, making convergence analysis a straightforward task. However, sparsification employs an empirical approach and sometimes offers better communicational cost reduction compared to quantization approaches. The major issue with such an approach is the lack of theoretical validity for the methodology. When it comes to KD, the method offers optimal system performance for heterogeneous FL systems. The method also faces similar issues as faced in sparsification, i.e., lack of theoretical validation.

### **7.3 COMMUNICATION SCENARIO**

Most of the existing articles operate with the assumption that the communication channels of the systems are in ideal condition and are independent

of factors such as resource availability of clients, such as system power, bandwidth, and so on. Additionally, even though the works concentrate on proposing methodologies for model compression and communication round reduction, thus reducing the overall communication time, they overlook the time-related aspect of the technology. In this section, we attempt to present the impacts of various communication scenarios on system performance and convergence rate in FL systems.

### **7.3.1 UNRELIABLE NETWORK CONDITIONS**

Usually, federated learning (FL) systems consist of multiple devices, such as IoT systems, computer systems, and mobile devices, which may operate in completely different network environments. Consequently, the unavailability of clients during model training is a common issue in FL. This issue is not limited solely to mobility-related problems but can also be caused by other factors, such as power outages or network outages. These factors can ultimately result in a reduced model convergence rate and deteriorating system performance. In light of this, Yu et al. [59] considered the situation and conducted a theoretical analysis, inferring that every communication channel has an inherent packet loss rate. However, assuming that every communication channel has an identical packet loss rate is a flawed inference, as it depends on factors such as time duration and packet size. Inspired by this, Zhang et al. [60] proposed a novel algorithm called ACFL, which is designed to adaptively compress the information transferred from the model based on factors such as the physical condition of the network, drop rate, and total volume of packets. Another approach is proposed in the article [61] for calculating the probability of each communication link succeeding. They make use of stochastic geometric tools to calculate the loss rate and then apply a weight factor based on the scheduling policy and the probability rate of individual clients based on successful gradient transfer during global model aggregation for optimizing the system performance. To mitigate issues associated with the straggler effect of client devices and outdated client models in heterogeneous networks, Wu et al. [62] proposed incorporating SAFA features into such systems. Especially in certain systems, some clients make use of the most recently updated version of global models as their base model for local training, while others are left with old models to perform training, causing a significant discrepancy in the final model generated by both devices.

Estimating the physical condition of a network is a challenging task. Wu et al. [63] presented a framework called Hybrid FL, which analyzed the reliability claims of systems developed with strong privacy-preserving mechanisms. They proposed embedding regional slack factors and altering client selection strategy on a regional basis to address these issues. Previous articles mathematically present the unreliability of networks in the physical layer, while certain articles focus on the transport-level perspective. For example, some articles only consider primitive communication protocols like UDP, rather than reliable protocols such as TCP or IP. The gossip-based averaging protocol is widely used as a fault-tolerant large-scale framework. However, the complexity of this approach increases linearly with the number of clients, which inversely affects the convergence of the FL model and increases communication overhead. To address this, Ryabinin et al. [64] proposed an approach that allows clients to dynamically align with a group where they feel they belong. Even during dropout, the influence of the client only affects the individual group.

### **7.3.2 NOISY FADING CHANNELS**

In recent times, extensive research has been conducted on implementing FL-based systems in real-life scenarios, especially with over-the-air communication capabilities where devices in the network communicate with each other through the medium access channel (MAC) and perform collaborations over wireless communication channels. Such systems, usually referred to as over-the-air FL or OTA-FL, are capable of simultaneously utilizing both the entire spectral as well as the temporal resources, thus reducing the overall communication overhead on the system. Nonetheless, such systems suffer from issues related to “noisy fading channel” problems, which are a genuine concern concerning communication channels as they lead to issues such as packet dropping or mismatch in packet information, and so on. The majority of the existing research concentrates on optimizing the uplink noisy fading in MAC. One such article is presented by Zhu et al. [65], where the author assumes the channel to follow IID Rayleigh fading and then designs an aggregation approach termed broadband analog aggregation (BAA). The proposed approach utilizes the superposition property of waves offered by the MAC to efficiently perform aggregation and resistance of such fading channels. An extension of the work is presented in article [66], where two complex algorithms are proposed for tackling the issue of fading channels. The authors propose a digitally distributed SGD algorithm for selecting

an individual device for transmitting distributed data in every single iteration. However, a major shortcoming of the proposed model was the lack of system robustness. Thus, an analog counterpart of the model referred to as the compressed analog distributed SGD (CADSGD) approach was proposed for accelerating computation through data sparsity and allocating power alignment of individual client devices to the server for achieving performance efficiency and system robustness. Nonetheless, the CADSGD power alignment methodology did not apply to any heterogeneous systems as it was controlled by individual clients. Later on, Yang et al. [67] considered the impact of noise on model convergence in OTAFL systems for non-IID distribution and heterogeneous scenarios. They proposed a highly flexible model framework, APC-OTALFL, which allows individual client devices to adaptively calculate the total transmission power level required and the total count of iterations required to maximize resource utilization in both communication and computation terms. Additionally, article [68] discussed dynamically adjusting the transmission power of individual devices based on the aggregated gradients collected in previous iterations to achieve better performance for over-the-air computations (Air Comp) in fading channels. Another FL aggregation approach, termed analog aggregation (AGA), adapts itself to changing receiver gradients depending on the training data and channel state information in fading channel systems. Article [69] also proposed a multi-access framework referred to as the Gradient-Based Multiple Access (GBMA) framework, which is claimed to function without any power control to nullify the effects of fading channels for various models. Instead, the GBMA approach directly affects noise distortion gradients. In addition to optimizing noisy fading channel uplink, article [70] proposed the usage of retransmissions even after communication failure to reduce estimation errors and improve convergence rate. The straggler problem in fading channels was investigated by Lin et al. [71], where they proposed a solution by assigning relays to allow clients to upload their models to the relay server. This helps mitigate issues with straggling clients. While most existing articles focus on uplink data transfer as the main cause of communication bottlenecks and strive to make it error-free, downlink data transfer also significantly affects system performance in FL systems. Thus, Xia et al. [72] proposed a framework called Fed Split to address issues related to noisy fading channels. They also introduced “Air Comp” at the client devices to recover lost data during the aggregation of local updates caused by noise. Amiri et al. [73] also tackled issues in download fading in broadcast channels by adopting a digital approach to quantize gradient updates. They provided a theoretical convergence analysis for analog downlink transmission. Some extensive

studies investigate both noise-fading downlinks and uplink channels, as seen in article [74]. This article performs an extensive convergence analysis on FedAvg for non-IID data distribution, limited client selection strategy, and noise fading bidirectional channel conditions. However, previous articles did not discuss the stochastic delay in time-varying fading channels. To address this issue, Li et al. [75] examined the delay distribution of FL systems with wireless channels for both uplink and downlink transmissions.

## **7.4 RESOURCE ALLOCATION**

Because of the challenges presented by the extensive heterogeneity of individual client devices and the resources they offer, developing an efficient communication strategy to transfer information in FL systems has always been a hot research topic. In this section, we attempt to present the existing literature on the applicability of various optimization approaches to solve these issues.

### **7.4.1 LOSS FUNCTION OF GLOBAL MODEL**

When optimizing the global loss function, multiple factors need to be considered and an optimal trade-off must be generated to achieve the desired result. Si et al. [144] attempted to establish a client scheduling approach and a bandwidth allocation strategy for individual clients by developing a trade-off between system latency and training iteration number. Article [76] considered different data partitioning approaches, namely vertical and horizontal, and optimized communication and computation power usage for both cases. The scheduling algorithm is converted into an optimization problem, aiming to minimize the global loss function. Article [77] also presents an iteration-termination-based approach to FedCau for achieving optimized computational and communication procedures. The authors also attempt to combine FedCau with Top-q and LAQ compression approaches to further optimize the communication side of the systems. However, most of the existing articles fail to achieve generalization capabilities as they mainly focus on features such as bandwidth and computational power management. Inspired by this issue, Wang et al. [78] proposed a generalized model for dynamically minimizing the loss function following a resource budget. The proposed model does not specify any single resource type and analyzes the convergence bound of the FL system as a woe for non-IID data distributions.

#### **7.4.2 TIME CONSUMPTION**

When considering time estimation in FL systems, both communication and computational time must be taken into account. Article [79] has examined both communication and computational timing to develop a strategy for client selection and power allocation. However, the proposed approach only considers the transmission time constraint. Yang et al. [80] focused on computation time and developed a bisection search algorithm to find an optimal solution for the problem. Additionally, article [81] also suggests considering the cost of the block validation procedure and introducing digital twin-based wireless systems to address resource deficiency issues during FL tasks in real-world applications.

#### **7.4.3 SELECTION OF CLIENT DEVICES**

Various strategies have been employed in FL systems for performing client selection. An approach based on greedy strategy is proposed in article [82], where the main focus is on scheduling a higher number of devices within a given duration of time in each iteration. The deadlines for the time frame are chosen experimentally, which makes it challenging to adapt the framework to dynamic channels and changing computing environments. To address these issues, Chen et al. [83] proposed the CEFL framework, which utilizes outdated model parameters received from client devices for scheduling and includes a theoretical analysis of its model convergence and communication patterns. However, existing literature has not considered the details of channel state information (CSI). Therefore, Zhao et al. [84] presented an approach where two bandwidth allocation approaches were utilized based on CSI and particle swarm optimization. Subsequently, the global loss function is formulated as an optimization problem, aiming to minimize it while maximizing the number of client devices.

#### **7.4.4 ENERGY USAGE**

An initial proposal for developing an energy-efficient approach was made by Zeng et al. [85]. The authors proposed a joint bandwidth allocation scheme along with a scheduling approach aiming to minimize the overall energy usage of the system. The framework worked on the assumption that every

client device had an identical model structure and thus was ignorant about the computational energy aspect. Yang et al. [86] addressed the issue associated with computational energy in their article, where a closed-form solution was proposed for managing the computational and transmission of resources in the case of lightweight algorithms.

#### **7.4.5 HYBRID OBJECTIVE**

Some of the existing research focuses on solving multiple objectives simultaneously, often referred to as multi-objective optimization approaches. For instance, certain articles have presented an optimization approach where multiple client resources, such as storage capacity, communication bandwidth, and power consumption, are attempted to be optimized in malicious and straggler systems. However, the authors fail to validate the functioning of the approach in a theoretical manner. Similarly, Zaw et al. [87] attempted to simultaneously optimize the loss function of the global model along with the computation and communication time, formulating the problem into a generalized Nash equilibrium problem and then solving it via a convergence analysis.

### **7.5 DISCUSSIONS AND FUTURE SCOPE**

Even though the current research focuses on paths of continuous updates, certain research gaps have been identified, leaving room for future research in specific aspects, as discussed below.

#### **7.5.1 COMPUTATIONAL REQUIREMENTS**

As discussed earlier, several data compression algorithms have been employed to optimize communication on the client's side of the system. Even though limited literature is available comprehensively discussing computation time and computing power consumption in FL systems, studies specifically concentrating on evaluating the power consumption of compression-based algorithms are lacking. Future research on various application scenarios such as mobile devices and other IoT systems is required to advance the research.

### **7.5.2 NEED FOR AN EXTENSIVE COMMUNICATION SYSTEM**

The majority of the existing literature considers a single compression method, and in some cases, a combination of two methods. However, these combinatorial approaches may not be sufficient for certain use cases. Therefore, an effective combination of multiple approaches is necessary to address these issues and create an optimized FL system with compression models. Additionally, it is crucial to identify the right balance of individual methods when developing hybrid approaches to ensure performance efficiency.

### **7.5.3 ASYNCHRONOUS AGGREGATION**

Through this article, we have focused solely on synchronous aggregation protocols in FL. However, the time required for client devices to transmit the gradient values of local models in each round may vary depending on the availability of communication and computational resources, as well as the volume of training data. This can lead to the server occasionally entering a waiting state due to a lack of gradients, which in turn slows down the entire training process, commonly known as the straggler problem. Additionally, since the local data distributed across client nodes often does not adhere to an identical probability distribution, the models trained on this dataset also exhibit variations. This poses challenges in model convergence, as these models may converge in different directions, necessitating additional iterations for the server to achieve better performance and increase training time. Therefore, the synchronous aggregation algorithm should be considered when addressing communication overhead to expedite model convergence and achieve improved system performance.

### **7.5.4 INTEGRATION WITH 5G AND BEYOND**

The rapid advancements in wireless communication technologies have facilitated the growth of Federated Learning (FL) as well. The typical bottleneck issues experienced in FL, such as bandwidth and processing speed, can be easily addressed with the introduction of 5G technology. The integration of the 5G network into any technology is expected to provide a significant boost in bandwidth and a considerable reduction in system latency. This feature enables various systems, such as the Internet of Things (IoT), the

Internet of Vehicular Things (IoVT), and other use cases, to perform better with FL models, as their real-time processing requirements can be met more effectively. For example, a real-time video analysis system with a focus on privacy preservation has been utilizing FL in conjunction with 5G to achieve its objectives. In a global industrial context, specific standards for the operation of FL on 5G networks have already been established. The introduction of 6G will also facilitate better ultra-low latency communications with higher reliability and efficiency in such systems. As 5G and beyond are becoming hot topics, enabling the establishment of huge volumes of heterogeneous networks with high levels of intelligence obtained from the availability of immense volumes of data, achieving communication efficiency becomes a primary consideration for FL systems. System heterogeneity, network connectivity, power levels, and so on will always lead to complicated networking scenarios and generate more vulnerabilities and occurrences of outages. Thus, it is essential to implement real-life simulations and effective training routines to accelerate convergence and identify bottlenecks in such scenarios related to AP/VR, autonomous transportation mechanisms, and so on, in order to address these issues.

## **7.6 CONCLUSION**

As a result of sophisticated wireless networking scenarios and the heterogeneity of devices, performing model training on edge devices and network edge is becoming a highly complex task in the current situation. This article attempts to present some prospective solutions to these issues from three different perspectives: communication efficiency, communication environment, and resource allocation. Firstly, we discuss various efficient communication strategies for federated learning systems. The study is divided into three main subcategories: Quantization-based approaches, sparsification-based approaches, and KD-based approaches. For each approach, we present their advantages, demerits, and possible aspects for future developments to provide a comprehensive study. Next, the study presents the influence of adverse effects on communication environments. We also discuss the effect of unfavorable communication environments on federated learning systems. We classify the scenarios into two main types: unreliable channels and noise-fading channels. In unreliable channels, the connection between the server and the client does not guarantee any sort of performance, and there is a high chance of the client becoming unavailable during training. When it comes to noisy fading channels, these channels introduce inherent

noise and a fading tendency that increases with distance. Finally, we discuss various resource allocation algorithms from a communication perspective, considering communication efficiency as an optimization problem subject to certain constraints. Thus, these approaches are classified based on their objective function for optimization. We identify the loss function of the global model, time consumption, selection of client devices, energy usage, and other commonly employed objectives for the same.

## KEYWORDS

- **bottlenecks**
- **communication efficiency**
- **federated learning**
- **integration**
- **internet of things**
- **optimization**
- **sparsification**

## REFERENCES

1. Alistarh, D., Demjan, G., Jerry, L., Ryota, T., & Milan, V., (2017). QSGD: Communication-efficient SGD via gradient quantization and encoding. *Advances in Neural Information Processing Systems*, 30.
2. Reisizadeh, A., Aryan, M., Hamed, H., Ali, J., & Ramtin, P., (2020). FedPAQ: A communication-efficient federated learning method with periodic averaging and quantization. In: *International Conference on Artificial Intelligence and Statistics* (pp. 2021–2031). PMLR.
3. Haddadpour, F., Mohammad, M. K., Aryan, M., & Mehrdad, M., (2021). Federated learning with compression: Unified analysis and sharp guarantees. In: *International Conference on Artificial Intelligence and Statistics* (pp. 2350–2358). PMLR.
4. Das, R., Anish, A., Abolfazl, H., Sujay, S., Inderjit, S. D., & Ufuk, T., (2022). Faster non-convex federated learning via global and local momentum. In: *Uncertainty in Artificial Intelligence* (pp. 496–506). PMLR.
5. Dai, X., Xiao, Y., Kaiwen, Z., Han, Y., Kelvin, K. W. N., James, C., & Yu, F., (2019). *Hyper-Sphere Quantization: Communication-Efficient SGD for Federated Learning*. arXiv preprint arXiv:1911.04655.
6. Jhunjhunwala, D., Advait, G., Gauri, J., & Yonina, C. E., (2021). Adaptive quantization of model updates for communication-efficient federated learning. In: *ICASSP 2021–2021*

- IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)* (pp. 3110–3114). IEEE.
- 7. Amiri, M. M., Deniz, G., Sanjeev, R. K., & Vincent, P. H., (2020). *Federated Learning with Quantized Global Model Updates*. arXiv preprint arXiv:2006.10672.
  - 8. Theertha, S. A., Felix, X. Yu., Sanjiv, K., & McMahan, B. H., (2016). *Distributed Mean Estimation with Limited Communication*. arXiv e-prints: arXiv-1611.
  - 9. Vargaftik, S., Ben-Basat, R., Amit, P., Gal, M., Ben-Itzhak, Y., & Michael, M., (2021). DRIVE: One-bit distributed mean estimation. *Advances in Neural Information Processing Systems*, 34, 362–377.
  - 10. Vargaftik, S., Ran, B. B., Amit, P., Gal, M., Yaniv, B. I., & Michael, M., (2022). Eden: Communication-efficient and robust distributed mean estimation for federated learning. In: *International Conference on Machine Learning* (pp. 21984–22014). PMLR.
  - 11. Basat, R. B., Shay, V., Amit, P., Gil, E., Ben-Itzhak, Y., & Michael, M., (2022). *QUICK-FL: Quick Unbiased Compression for Federated Learning*. arXiv preprint arXiv:2205.13341.
  - 12. Zamir, R., & Meir, F., (1992). On universal quantization by randomized uniform/lattice quantizers. *IEEE Transactions on Information Theory*, 38(2), 428–436.
  - 13. Shlezinger, N., Mingzhe, C., Yonina, C. E., Vincent, P. H., & Shuguang, C., (2020). Federated learning with quantization constraints. In: *ICASSP 2020–2020 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)* (pp. 8851–8855). IEEE.
  - 14. Chen, M., Nir, S., Vincent, P. H., Yonina, C. E., & Shuguang, C., (2021). Communication-efficient federated learning. *Proceedings of the National Academy of Sciences*, 118(17), e2024789118.
  - 15. Bernstein, J., Yu-Xiang, W., Kamyar, A., & Animashree, A., (2018). signSGD: Compressed optimization for non-convex problems. In: *International Conference on Machine Learning* (pp. 560–569). PMLR.
  - 16. Jin, R., Yufan, H., Xiaofan, H., Huaiyu, D., & Tianfu, W., (2020). *Stochastic-Sign SGD for Federated Learning with Theoretical Guarantees*. arXiv preprint arXiv:2002.10940.
  - 17. Zhu, G., Yuqing, D., Deniz, G., & Kaibin, H., (2020). One-bit over-the-air aggregation for communication-efficient federated edge learning: Design and convergence analysis. *IEEE Transactions on Wireless Communications*, 20(3), 2120–2135.
  - 18. Abdi, A., & Faramarz, F., (2020). Quantized compressive sampling of stochastic gradients for efficient communication in distributed deep learning. In: *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 34, No. 4, pp. 3105–3112).
  - 19. Li, C., Gang, L., & Pramod, K. V., (2021). Communication-efficient federated learning based on compressed sensing. *IEEE Internet of Things Journal*, 8(20), 15531–15541.
  - 20. Fan, X., Yue, W., Yan, H., & Zhi, T., (2021). Communication-efficient federated learning through 1-bit compressive sensing and analog aggregation. In: *2021 IEEE International Conference on Communications Workshops (ICC Workshops)* (pp. 1–6). IEEE.
  - 21. Oh, Y., Namyoong, L., & Yo-Seb, J., (2021). Quantized compressed sensing for communication-efficient federated learning. In: *2021 IEEE GLOBECOM Workshops (GC Wkshps)* (pp. 1–6). IEEE.
  - 22. He, Y., Maximilian, Z., & Mario, F., (2020). *CosSGD: Nonlinear Quantization for Communication-Efficient Federated Learning*. arXiv preprint arXiv:2012.08241.
  - 23. Malekjoo, A., Mohammad, J. F., Hanieh, M., Morteza, H., Alizadeh-Shabdz, F., & Reza, R., (2021). *Fed zip: A Compression Framework for Communication-Efficient Federated Learning*. arXiv preprint arXiv:2102.01593.

24. Philippenko, C., & Aymeric, D., (2020). *Bidirectional Compression in Heterogeneous Settings for Distributed or Federated Learning with Partial Participation: Tight Convergence Guarantees*. arXiv preprint arXiv:2006.14591.
25. Chen, S., Cong, S., Lanxue, Z., & Yuanmin, T., (2021). Dynamic aggregation for heterogeneous quantization in federated learning. *IEEE Transactions on Wireless Communications*, 20(10), 6804–6819.
26. Cui, L., Xiaoxin, S., Yipeng, Z., & Yi, P., (2021). Slashing communication traffic in federated learning by transmitting clustered model updates. *IEEE Journal on Selected Areas in Communications*, 39(8), 2572–2589.
27. Sahu, A., Aritra, D., Ahmed, M. A., Trambak, B., Marco, C., & Panos, K., (2021). Rethinking gradient sparsification as total error minimization. *Advances in Neural Information Processing Systems*, 34, 8133–8146.
28. Han, P., Shiqiang, W., & Kin, K. L., (2020). Adaptive gradient sparsification for efficient federated learning: An online learning approach. In: *2020 IEEE 40th International Conference on Distributed Computing Systems (ICDCS)* (pp. 300–310). IEEE.
29. Sattler, F., Simon, W., Klaus-Robert, M., & Wojciech, S., (2019). Sparse binary compression: Towards distributed deep learning with minimal communication. In: *2019 International Joint Conference on Neural Networks (IJCNN)* (pp. 1–8). IEEE.
30. Nori, M. K., Sangseok, Y., & Il-Min, K., (2021). Fast federated learning by balancing communication trade-offs. *IEEE Transactions on Communications*, 69(8), 5168–5182.
31. Abdelmoniem, A. M., & Marco, C., (2021). DC2: Delay-aware compression control for distributed machine learning. In: *IEEE INFOCOM 2021-IEEE Conference on Computer Communications* (pp. 1–10). IEEE.
32. Li, S., Qi, Q., Jingyu, W., Haifeng, S., Yujian, L., & Richard, Y. F., (2020). GGS: General gradient sparsification for federated learning in edge computing. In: *ICC 2020–2020 IEEE International Conference on Communications (ICC)* (pp. 1–7). IEEE.
33. Rothchild, D., Ashwinee, P., Enayat, U., Nikita, I., Ion, S., Vladimir, B., Joseph, G., & Raman, A., (2020). FetchSGD: Communication-efficient federated learning with sketching. In: *International Conference on Machine Learning* (pp. 8253–8265). PMLR.
34. Sattler, F., Simon, W., Klaus-Robert, M., & Wojciech, S., (2019). Robust and communication-efficient federated learning from non-IID data. *IEEE Transactions on Neural Networks and Learning Systems*, 31(9), 3400–3413.
35. Shi, S., Qiang, W., Kaiyong, Z., Zhenheng, T., Yuxin, W., Xiang, H., & Xiaowen, C., (2019). A distributed synchronous SGD algorithm with global top-k sparsification for low bandwidth networks. In: *2019 IEEE 39<sup>th</sup> International Conference on Distributed Computing Systems (ICDCS)* (pp. 2238–2247). IEEE.
36. Xu, H., Kelly, K., Aritra, D., Xin, L., Alexandros, N., & Panos, K., (2021). Deep Reduce: A sparse-tensor communication framework for federated deep learning. *Advances in Neural Information Processing Systems*, 34, 21150–21163.
37. Shi, S., Zhenheng, T., Qiang, W., Kaiyong, Z., & Xiaowen, C., (2019). *Layer-Wise Adaptive Gradient Sparsification for Distributed Deep Learning with Convergence Guarantees*. arXiv preprint arXiv:1911.08727.
38. Shi, S., Qiang, W., Xiaowen, C., Bo, L., Yang, Q., Ruihao, L., & Xinxiao, Z., (2020). Communication-efficient distributed deep learning with merged gradient sparsification on GPUs. In: *IEEE INFOCOM 2020-IEEE Conference on Computer Communications* (pp. 406–415). IEEE.
39. Li, D., & Junpu, W., (2019). *FedMD: Heterogenous Federated Learning Via Model Distillation*. arXiv preprint arXiv:1910.03581.

40. Lin, T., Lingjing, K., Sebastian, U. S., & Martin, J., (2020). Ensemble distillation for robust model fusion in federated learning. *Advances in Neural Information Processing Systems*, 33, 2351–2363.
41. Sattler, F., Arturo, M., Roman, R., & Wojciech, S., (2021). CFD: Communication-efficient federated distillation via soft-label quantization and delta coding. *IEEE Transactions on Network Science and Engineering*, 9(4), 2025–2038.
42. Itahara, S., Takayuki, N., Yusuke, K., Masahiro, M., & Koji, Y., (2021). Distillation-based semi-supervised federated learning for communication-efficient collaborative training with non-IID private data. *IEEE Transactions on Mobile Computing*, 22(1), 191–205.
43. Sattler, F., Tim, K., Roman, R., & Wojciech, S., (2021). FedAUX: Leveraging unlabeled auxiliary data in federated learning. *IEEE Transactions on Neural Networks and Learning Systems*.
44. Li, X., Yanxia, G., Yuan, L., & Li-E, W., (2021). Personalized federated learning with semisupervised distillation. *Security and Communication Networks*, 2021.
45. Sturluson, S. P., Samuel, T., Muñoz-González, L., Matei, G., Passerat-Palmbach, J., Daniel, R., & Amir, A., (2021). *FedRAD: Federated Robust Adaptive Distillation*. arXiv preprint arXiv:2112.01405.
46. Liu, L., Jun, Z., Song, S. H., & Khaled, B. L., (2022). *Communication-Efficient Federated Distillation with Active Data Sampling*. arXiv preprint arXiv:2203.06900.
47. Jeong, E., Seungeun, O., Hyesung, K., Jihong, P., Mehdi, B., & Seong-Lyun, K., (2018). *Communication-Efficient on-Device Machine Learning: Federated Distillation and Augmentation Under Non-IID Private Data*. arXiv preprint arXiv:1811.11479.
48. Jiang, D., Chen, S., & Zhihui, Z., (2020). Federated learning algorithm based on knowledge distillation. In: *2020 International Conference on Artificial Intelligence and Computer Engineering (ICAICE)* (pp. 163–167). IEEE.
49. Wu, C., Fangzhao, W., Lingjuan, L., Yongfeng, H., & Xing, X., (2022). Communication-efficient federated learning via knowledge distillation. *Nature Communications*, 13(1), 1–8.
50. Zhu, Z., Junyuan, H., & Jiayu, Z., (2021). Data-free knowledge distillation for heterogeneous federated learning. In: *International Conference on Machine Learning* (pp. 12878–12889). PMLR.
51. Yao, D., Wanning, P., Yutong, D., Yao, W., Xiaofeng, D., Hai, J., Zheng, X., & Lichao, S., (2021). *Local-Global Knowledge Distillation in Heterogeneous Federated Learning with Non-IID Data*. arXiv preprint arXiv:2107.00051.
52. Zhang, L., Li, S., Liang, D., Dacheng, T., & Ling-Yu, D., (2022). Fine-tuning global model via data-free knowledge distillation for non-IID federated learning. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 10174–10183).
53. Cho, Y. J., Andre, M., Gauri, J., Robert, S., & Dimitrios, D., (2022). *Heterogeneous Ensemble Knowledge Transfer for Training Large Models in Federated Learning*. arXiv preprint arXiv:2204.12703.
54. Zhu, Z., Junyuan, H., Steve, D., & Jiayu, Z., (2022). Resilient and communication efficient learning for heterogeneous federated systems. In: *Proceedings of Thirty-ninth International Conference on Machine Learning (ICML 2022)*.
55. Zhang, L., Dapeng, W., & Xiaoyong, Y., (2022). FedZKT: Zero-shot knowledge transfer towards resource-constrained federated learning with heterogeneous on-device models. In: *2022 IEEE 42<sup>nd</sup> International Conference on Distributed Computing Systems (ICDCS)* (pp. 928–938). IEEE.

56. Li, T., Maziar, S., Ahmad, B., & Virginia, S., (2019). *Fair Resource Allocation in Federated Learning*. arXiv preprint arXiv:1905.10497.
57. Marfoq, O., Chuan, X., Giovanni, N., & Richard, V., (2020). Throughput-optimal topology design for cross-silo federated learning. *Advances in Neural Information Processing Systems*, 33, 19478–19487.
58. Guo, Y., Ying, S., Rui, H., & Yanmin, G., (2021). Hybrid local SGD for federated learning with heterogeneous communications. In: *International Conference on Learning Representations*.
59. Amiri, M. M., & Deniz, G., (2020). Federated learning over wireless fading channels. *IEEE Transactions on Wireless Communications*, 19(5), 3546–3557.
60. Wu, W., Ligang, H., Weiwei, L., Rui, M., Carsten, M., & Stephen, J., (2020). SAFA: A semi-asynchronous protocol for fast-federated learning with low overhead. *IEEE Transactions on Computers*, 70(5), 655–668.
61. Wu, W., Ligang, H., Weiwei, L., & Rui, M., (2020). Accelerating federated learning over reliability-agnostic clients in mobile edge computing systems. *IEEE Transactions on Parallel and Distributed Systems*, 32(7), 1539–1551.
62. Yu, C., Hanlin, T., Cedric, R., Simon, K., Ankit, S., Dan, A., Ce, Z., & Ji, L., (2019). Distributed learning over unreliable networks. In: *International Conference on Machine Learning* (pp. 7202–7212). PMLR.
63. Zhang, X., Xiaomin, Z., Ji, W., Hui, Y., Huangke, C., & Weidong, B., (2020). Federated learning with adaptive communication compression under dynamic bandwidth and unreliable networks. *Information Sciences*, 540, 242–262.
64. Zhu, G., Yong, W., & Kaibin, H., (2019). Broadband analog aggregation for low-latency federated edge learning. *IEEE Transactions on Wireless Communications*, 19(1), 491–506.
65. Yang, H., Peiwen, Q., Jia, L., & Aylin, Y., (2022). *Over-the-air federated Learning with Joint Adaptive Computation and Power Control*. arXiv preprint arXiv:2205.05867.
66. Zhang, N., & Meixia, T., (2021). Gradient statistics aware power control for over-the-air federated learning. *IEEE Transactions on Wireless Communications*, 20(8), 5115–5128.
67. Sery, T., & Kobi, C., (2020). On analog gradient descent learning over multiple access fading channels. *IEEE Transactions on Signal Processing*, 68, 2897–2911.
68. Hellström, H., Viktoria, F., & Carlo, F., (2021). Over-the-air federated learning with retransmissions. In: *2021 IEEE 22<sup>nd</sup> International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)* (pp. 291–295). IEEE.
69. Lin, Z., Hang, L., & Ying-Jun, A. Z., (2022). Relay-assisted cooperative federated learning. *IEEE Transactions on Wireless Communications*.
70. Xia, S., Jingyang, Z., Yuhan, Y., Yong, Z., Yuanming, S., & Wei, C., (2021). Fast convergence algorithm for analog federated learning. In: *ICC 2021-IEEE International Conference on Communications* (pp. 1–6). IEEE.
71. Amiri, M. M., Deniz, G., Sanjeev, R. K., & Vincent, P. H., (2021). Convergence of federated learning over a noisy downlink. *IEEE Transactions on Wireless Communications*, 21(3), 1422–1437.
72. Wei, X., & Cong, S., (2022). Federated learning over noisy channels: Convergence analysis and design examples. *IEEE Transactions on Cognitive Communications and Networking*.
73. Li, L., Longwei, Y., Xin, G., Yuanming, S., Haiming, W., Wei, C., & Khaled, B. L., (2021). Delay analysis of wireless federated learning based on saddle point approximation and large deviation theory. *IEEE Journal on Selected Areas in Communications*, 39(12), 3772–3789.

74. Ryabinin, M., Eduard, G., Vsevolod, P., & Gennady, P., (2021). Moshpit SGD: Communication-efficient decentralized training on heterogeneous unreliable devices. *Advances in Neural Information Processing Systems*, 34, 18195–18211.
75. Shi, W., Sheng, Z., Zhisheng, N., Miao, J., & Lu, G., (2020). Joint device scheduling and resource allocation for latency-constrained wireless federated learning. *IEEE Transactions on Wireless Communications*, 20(1), 453–467.
76. Yu, C., Shuaiqi, S., Kuan, Z., Hai, Z., & Yeyin, S., (2022). Energy-aware device scheduling for joint federated learning in edge-assisted internet of agriculture things. In: *2022 IEEE Wireless Communications and Networking Conference (WCNC)* (pp. 1140–1145). IEEE.
77. Mahmoudi, A., Hossein, S. G., José, M. B. Da. S. Jr., & Carlo, F., (2022). *FedCau: A Proactive Stop Policy for Communication and Computation Efficient Federated Learning*. arXiv preprint arXiv:2204.07773.
78. Wang, S., Tiffany, T., Theodoros, S., Kin, K. L., Christian, M., Ting, H., & Kevin, C., (2019). Adaptive federated learning in resource-constrained edge computing systems. *IEEE Journal on Selected Areas in Communications*, 37(6), 1205–1221.
79. Chen, M., Vincent, P. H., Walid, S., & Shuguang, C., (2020). Convergence time optimization for federated learning over wireless networks. *IEEE Transactions on Wireless Communications*, 20(4), 2457–2471.
80. Yang, Z., Mingzhe, C., Walid, S., Choong, S. H., Shikh-Bahaei, M., Vincent, P. H., & Shuguang, C., (2020). *Delay Minimization for Federated Learning Over Wireless Communication Networks*. arXiv preprint arXiv:2007.03462.
81. Lu, Y., Xiaohong, H., Ke, Z., Sabita, M., & Yan, Z., (2020). Low-latency federated learning and blockchain for edge association in digital twin-empowered 6G networks. *IEEE Transactions on Industrial Informatics*, 17(7), 5098–5107.
82. Nishio, T., & Ryo, Y., (2019). Client selection for federated learning with heterogeneous resources in the mobile edge. In: *ICC 2019–2019 IEEE International Conference on Communications (ICC)* (pp. 1–7). IEEE.
83. Chen, H., Shaocheng, H., Deyou, Z., Ming, X., Mikael, S., & Vincent, P. H., (2022). Federated learning over wireless IoT networks with optimized communication and resources. *IEEE Internet of Things Journal*.
84. Zhao, Z., Junjuan, X., Lisheng, F., Xianfu, L., George, K. K., & Arumugam, N., (2021). System optimization of federated learning networks with a constrained latency. *IEEE Transactions on Vehicular Technology*, 71(1), 1095–1100.
85. Zeng, Q., Yuqing, D., Kaibin, H., & Kin, K. L., (2020). Energy-efficient radio resource allocation for federated edge learning. In: *2020 IEEE International Conference on Communications Workshops (ICC Workshops)* (pp. 1–6). IEEE.
86. Yang, Z., Mingzhe, C., Walid, S., Choong, S. H., & Shikh-Bahaei, M., (2020). Energy efficient federated learning over wireless communication networks. *IEEE Transactions on Wireless Communications*, 20(3), 1935–1949.
87. Zaw, C. W., Shashi, R. P., Kitae, K., & Choong, S. H., (2021). Energy-aware resource management for federated learning in multi-access edge computing systems. *IEEE Access*, 9, 34938–34950.

## CHAPTER 8

---

# Federated Learning and Privacy, Challenges, Threat and Attack Models, and Analysis

SHEEMA MADHUSUDHANAN,<sup>1</sup> ARUN CYRIL JOSE,<sup>1</sup> and  
REZA MALEKIAN<sup>2</sup>

<sup>1</sup>*Department of Computer Science and Engineering, Indian Institute of Information Technology Kottayam (IIITK), Kerala, India*

<sup>2</sup>*Department of Computer Science and Media Technology, Internet of Things and People Research Center, Malmö University, Malmö, Sweden*

---

### ABSTRACT

The advent of intertwined technology, conjoined with powerful centralized machine algorithms, spawns the need for privacy. The efficiency and accuracy of any Machine Learning (ML) algorithm are proportional to the quantity and quality of data collected for training, which could often compromise the data subject's privacy. Federated Learning (FL) or collaborative learning is a branch of Artificial Intelligence (AI) that decentralizes ML algorithms across edge devices or local servers. This chapter discusses privacy threat models in ML and expounds on FL as a Privacy-preserving Machine Learning (PPML) system by distinguishing FL from other decentralized ML algorithms. We elucidate the comprehensive secure FL framework with Horizontal FL, Vertical FL, and Federated Transfer Learning that mitigates privacy issues. For privacy preservation, FL extends its capacity to incorporate Differential Privacy (DP) techniques to provide quantifiable measures on data anonymization. We have also discussed the concepts in FL that comprehend Local Differential Privacy (LDP) and Global Differential Privacy (GDP). The chapter concludes with

open research problems and challenges of FL as PPML with implications, limitations, and future scope.

## 8.1 INTRODUCTION

The advancements in algorithms and technology in this era of ubiquitous computing have posed challenges to user privacy and data confidentiality. Traditional Machine Learning (ML) methods involve training over a dataset on a machine, data center, or server [1]. Data integration faces significant resistance, making it challenging to effectively share data [2–4], resulting in data inconsistency and disparity. Consequently, consolidating this data in a data center to ensure privacy acts or rights can be costly. The General Data Protection Regulation (GDPR) [5, 6] and the California Consumer Privacy Act (CCPA) [7, 8] state that all personal sensitive information must be used or reproduced prudently. Often, users are unaware of the privacy policies or sharing rights of their data over electronic mediums. Many neglect reading the privacy rights or sharing policies, leading to unauthorized data sharing with websites or social media platforms [9]. The data shared over the Internet without proper authorization can later reach the Darknet [10]. Recently, attackers have posted patient medical records from various hospitals on the Darknet [11, 12]. People are unaware that the demand for sensitive personal information is very high, and it is being sold on the Darknet at a significant cost. We should remain vigilant, as someone is always watching over our data.

Personal sensitive information must require explicit authorization and the data owner's consent. Thus, all the data in the machine or the data center must ensure the protection of personal sensitive information that cannot be mapped or tracked by any unlawful authority [13]. It urges the need for a decentralized ML model. In the Federated Learning (FL) model by Google [14, 15], training data can be retained in the portable device by collaboratively sharing the learned predicted model. Privacy-preserved data collection methods assist LDP and GDP, which perturbs the user data before being collected, stored, or accessed [16–18].

Differential Privacy (DP) [19, 20] ensures that the DP function result is not sensitive to any distinct record in the dataset. The concept of DP involves adding noise to the data before it leaves the personal device, making it impossible for the data-receiving entity to determine an individual's data. In other words, an attacker cannot differentiate whether a user's data is present in the dataset or identify an individual's personal information. In 2018, Google introduced privacy preservation in Federated Learning (FL) through

the proposal of federated optimization algorithms: Federated Stochastic Gradient Descent (FedSGD) and Federated Averaging (FedAvg). These algorithms guarantee user-level DP [15].

### **8.1.1 DATA PRIVACY**

Privacy refers to any information about a specific individual or group that wishes to avoid indiscriminate exposure [21]. The main focus of privacy protection lies in the data held by the data owner or authority [22–25]. The data possessed by the data owner or authority is complete and unprocessed. Sensitive information, such as location-based data, facial images, and health information, can be utilized for targeted advertising and recommending actions that pose potential privacy threats. Various techniques, including generalization, suppression, and anonymization, have been developed to anonymize all sensitive personal information [26–29]. However, anonymization techniques are susceptible to different attacks [30–33], such as background knowledge attacks [34, 35] and linkage (membership inference) attacks [36, 37]. Over the past few decades, data privacy preservation models like k-anonymity [38],  $\ell$  diversity [39], t-closeness [40, 41], m-invariance [42], Randomization [43], and DP [19, 20] have successfully mitigated diverse privacy attacks. Nevertheless, these privacy preservation models are still vulnerable to various privacy attacks. Therefore, the pursuit of advancing privacy preservation models is an ongoing endeavor. In data privacy preservation, it is anticipated that attackers will possess more extensive background knowledge through data mining analysis technology, enabling them to launch multi-source data linkage attacks. Privacy-preserving mining techniques rely on data clustering, classification, and rule mining. However, most research focuses on specific data mining tasks, lacking the general applicability required for data independence and exposure.

### **8.1.2 IMPORTANCE OF PRIVACY**

Recent research shows that many industries and IT organizations globally have the third-highest per capita cost of data breaches [44–49]. The data breach rate is alarming, regardless of whether it is a large or small company. Recently, data breaches have been reported from various sectors dealing with sensitive data, including healthcare, information technology, social networking, telecommunications, finance, transportation, entertainment or

media, hospitality, retail, government, manufacturing, and energy [46–49]. Therefore, the threat landscape is well-versed. Protecting private or personal data from attackers is vital, as data breaches have occurred in companies such as Facebook, Amazon, Yahoo, VeriSign, Syniverse, MySpace, Uber, Marriott International, River City Media, Crypto.Com, Plex media server, and others [46–49]. We have listed some of the significant or recent data privacy attacks reported [46–49]. Yahoo suffered a data breach involving 3 billion customer information, while in 2021, the social media giant Facebook reported that 533 million users' data was posted in the hacking forum. An ex-Amazon employee was convicted for stealing the personal information of approximately 100 million customers. Verisign, the security certificate issuer and Internet infrastructure provider, was repeatedly hacked, resulting in undisclosed information theft. Syniverse, a telecommunication infrastructure provider, was targeted in 2021, and attackers gained access to 500 million customers' data. MySpace, a social networking service provider, reported unauthorized access to 400 million user records. Uber, an American mobility service provider, admitted that in 2016, they suffered a massive data breach, and the company paid US\$100,000 to hackers for deleting the stolen information. Cyber criminals hacked the reservation system of Marriott International hospitality group and stole 383 million users' records. River City Media, a company specializing in creative visual solutions, reported that a spam email operator accidentally exposed 1.37 billion records, becoming one of the worst data breach nightmares. In 2022, Crypto.Com experienced a compromise of crypto-based transactions and lost US\$18 million worth of bitcoin and US\$15 million worth of Ethereum and other cryptocurrencies. In the same year, the Plex media server app data breach compromised customers' encrypted data, including passwords, usernames, and emails. It is reported that the number of people using OSN in 2022 is 4.59 billion, which is more than half of the global population. From 2020 to 2021, the average per capita cost of a data breach increased by 10.3% [47]. Furthermore, a study reveals that 53% of online users are concerned about the probability of sensitive data being disclosed [47]. Therefore, it is essential to have robust privacy preservation methods to protect private, personal, or sensitive data circulated in various sectors as part of our socio-internet life.

Most organizations and industries rely on cryptographic techniques and network monitoring solutions to protect the data and work environment. However, many face the risk of data breaches due to the lack of appropriate protection mechanisms for sensitive data, which affects both production and development environments. With data in transit and the proliferation of such data into various systems, identifying and continuously tracking the location

of sensitive data is challenging, requiring attention to the possibility of data leakage.

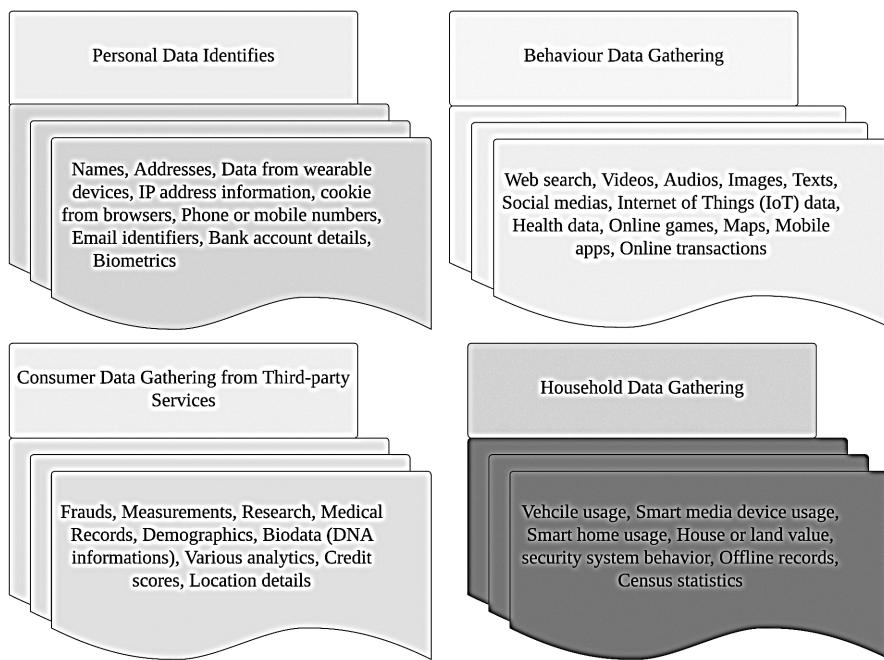
The rest of the chapter is organized as follows. Section 2 discusses the evolution of privacy and the envisioned privacy regulations laws. Section 3 covers concepts related to threat and adversarial models, Privacy-preserving Machine Learning (PPML), and Distributed Machine Learning (DML). The core of the chapter is presented in Section 4, which includes the secure framework, various privacy preservation techniques, and data utility optimizations of Federated Learning (FL). Section 5 explores the privacy challenges and attacks in FL, followed by a discussion on open challenges in FL. The chapter concludes in Section 6.

## **8.2 THE EVOLUTION OF PRIVACY**

The perspective on personal information has been changing since we became connected in the world of Internet-connected devices, such as smartphones, smart home devices, smart cars, smart buildings, and smart health devices. These devices collect various types of data at regular intervals and send it for analysis and processing [5, 6]. However, not all data can be considered personal or sensitive. The level of sensitivity depends on factors such as type, priority, viewpoint, purpose, belief, and opinion. The concept of sensitive data is illustrated in Figure 8.1 [50], which shows parameters for determining sensitiveness. We can classify data as public (low data sensitivity), internal-only (moderate data sensitivity), confidential, or restricted (high data sensitivity). Privacy rights, ethics, and the law have a complex history since concerns about electronic data safety were introduced. Various authorities and international organizations have established privacy principles to ensure the free flow of data [5, 6]. Despite the technical and regulatory efforts to safeguard personal information privacy, data breaches still occur. However, privacy plays a vital role in building trust and transparency in the digital era, as data flow between systems is inevitable.

### **8.2.1 THE TRANSITION IN PRIVACY REGULATION LAWS**

The GDPR came into existence in 2018 and defined personal information as any information relating to an individual, such as name, identification number, location data, or an online identifier [5, 6, 50]. In the GDPR, the authority has also defined a special category of data.



**FIGURE 8.1** Sensitive data parameters.

In the Data Protection Act (1988), it was named sensitive personal data [51]. The special category data has priority over standard personal data and includes biometric and genetic data, but excludes conviction data of criminals. In GDPR, personal data under the special category is related to racial or ethnic origin, gender, religious or philosophical beliefs, political or trade union information, political beliefs, physical or mental health, and genetic or biometric data.

The CCPA came into effect in 2020, introducing significant privacy laws for the world's largest markets, including Brazil and India [7, 8]. Traders will face restrictions on accessing the data they rely on for their digital marketing campaigns. Under the CCPA, personal information refers to data that can be directly or indirectly linked to a consumer or specific household. Unlike the GDPR, which focuses on individual data, the CCPA categorizes data as household data.

The process of zero-trust data sharing involves strict verification between participating entities in a transaction [52]. These entities must ensure that the data encryption logic is not shared with any individual or other entities. They should use data protection or random secrets for each participating

entity. Additionally, the entities should replace identifiers with appropriate pseudonyms and implement anonymization using a one-time ephemeral matching key. Furthermore, participating entities must ensure that no identity transfer occurs among them. There should be a commitment to not disclose, retain, or persistently store identifiers, making re-identification impossible.

### **8.3 MACHINE LEARNING AND PRIVACY**

ML automates the methods for analysis and processing of large datasets to perform application-related tasks based on what is learned by the machines. We may use distinct learning techniques, such as supervised and unsupervised, with classifications, regression, clustering, and rule-mining algorithms. While cryptography in ML provides confidentiality to the dataset regarding privacy preservation in the dataset, we should have some technologies that support ML to enhance data privacy. Recent works focus on leveraging ML executions in untrusted environments to Trusted Execution Environments (TEE). Protecting privacy frequently poses significant difficulties for the ML system design. Lack of privacy protection may even be detrimental [53]. Currently, there are two major approaches to attaining privacy in ML.

- Leveraging model parameters and dataset to be private; and
- Enhancing ML model to predict any data leakage.

It is also interesting to note that recent research works revolve around considering ML as an attack vector and thus improving the design and methodical practices on ML. To prevent ML systems from accessing private information from photos, Liu et al. suggested strategies for applying adversarial perturbations [54]. DML allows individual users/devices to transmit model parameters for model training rather than data samples. The participants in conventional DML algorithms send the locally computed parameters or gradients coordinator to support collaborative learning [55].

#### **8.3.1 PRIVACY THREAT MODELING**

Threat modeling must be performed in a product's development cycle to identify and address potential issues early on. In privacy threat modeling, we systematically analyze system features, such as faulty design or implementation flaws, architecture vulnerabilities, or loopholes in the data life cycle, to assess their impact on the privacy goals of our product or work.

The selection of a threat model generally depends on the type of application being developed and the level of privacy assurance required for end-users.

We have various threat modeling methods based on the targets and their processes. STRIDE from Microsoft [56] is one of the most mature and adaptive threat models for evaluating the system's detailed design. STRIDE helps us identify product boundaries, system entities, and events. It looks for spoofing, tampering, and repudiation of data, as well as checks for information disclosure, Denial of Service (DoS), and elevation of privilege. All the violation checks are associated with authentication, integrity, non-repudiation, confidentiality, availability, and authorization. However, we do not have anything specific that relates to threat modeling for privacy. Microsoft has also developed DREAD [57], which is associated with potential damage, reproducibility, users affected, exploitability, and discoverability. However, there is nothing specific that we can relate to privacy. Apart from STRIDE and DREAD, we have various other famous threat models such as PASTA [58], LINDDUN [59], CVSS from NIST [60], Attack Trees [61], Persona Non-Grata (PnG) [62], Security Cards [63], htMM from SEI [64], Trike [65], and VAST [66]. Out of these, for privacy, we go for LINDDUN. LINDDUN stands for linkability, identifiability, non-repudiation, detectability, information disclosure, unawareness, and non-compliance.

LINDDUN focuses on privacy concerns that are systematically addressed, creating a map from the problem space to the solution space. The approach to privacy assessment in LINDDUN begins with the creation of a Data Flow Diagram (DFD), which maps privacy threats to the DFD elements and identifies threat scenarios in the problem space. After this session, in the solution space, LINDDUN prioritizes threats, proposes mitigation strategies, and determines Privacy Enhancing Technologies (PETs). Therefore, LINDDUN emphasizes data flows and the anonymity of data flows, which is beneficial when aiming for total anonymity minimization. However, LINDDUN does not provide a concept of the data life cycle; instead, it offers a snapshot of the system.

Identifying or discovering the privacy threat from the data life cycle phase is necessary, which consists of five phases. The phase starts with data collection, data storage, data handling, data anonymization or pseudonymization, and data removal.

- **Data Collection:** In this phase, we gather information about a specific entity or subject through various sources. For example, in the data collection phase, filling out an online registration form will collect the name, social security number, address, phone number, birth date,

etc. All this information is visible to us, but in the application's background, it might also collect metadata such as the system's IP address, operating system name, version, browser settings, browser language, etc. Therefore, we can state that there is an apparent privacy violation in the application's background by collecting numerous details without the knowledge of the end users. We can summarize various threats during the data collection phase as follows:

- o Are the data incorrect?
- o What is the purpose of data collection?
- o Is the metadata getting stored?
- o Whether the data received from the primary source or a secondary source?
- o Is the data or the context of data carry sensitiveness?
- o Is it unsafe for transmission?
- **Data Storage:** In this phase, we have the database components, logs, caches, history, download information, etc. Here, we need to be concerned about storage that may have various stakeholders. For instance, consumers may want to utilize our data for testing, generating future changes, predictions, and suggestions, logging a complete HTTP request for debugging purposes, and downloading customer data to a personal device. Therefore, we can clearly state that there is a visible threat in this phase as our data is being shared among many. We can summarize the various threats to be addressed during the data storage phase as follows:
  - o Is the data confidential?
  - o Where do we store the data?
  - o Is there a cache or temporary storage?
  - o Who can access the data?
- **Data Handling:** In this phase, we need to address the situations where data leaks can occur due to mishandling of data or personal identification information. Therefore, the threats associated with data handling must address the following questions.
  - o Is the data sensitive?
  - o Where do we transfer the data?
  - o How do other systems use this data?
  - o Can we track who has used or seen this data?
  - o Can a user change their data?
  - o Are there any new use cases?
- **Data Anonymization or Pseudonymization:** In this phase, we must address data usage without revealing the data's secrecy, according to GDPR. In anonymization, we remove personal or sensitive data,

which is a non-reversible process. In pseudonymization, we can make the data unintelligible, but it is always a reversible process. The threats we have to address in this phase are as follows.

- o Can somebody reverse the algorithm?
- o Whether the key (pseudonym) exposed?
- o Can anyone access multiple data sources and relate the data anonymized/pseudonymized?
- o Is the dataset too small?
- o Whether only direct identifiers removed from the dataset?
- o Are there a large number of attributes to handle?
- **Data Removal:** In this phase, we need to address when to remove the data or certain identifiable parts of the data. For instance, when we uninstall an app from a mobile phone, we should clarify whether it only removes the app itself or also deletes all the data associated with the app or the app's data usage. Therefore, we must address the following threats.
  - o Is the data removal automatic or manual?
  - o Why has it been removed?
  - o What happens if we restore the backups?
  - o Is it there in the secondary storage after the removal?
  - o Is all the data removed, or is some data kept aside for later purposes?
  - o How do we get the removed data?

Considering all the above-mentioned points, we need to evaluate the privacy threats in the appropriate cluster based on the types of threats being addressed, the existing system vulnerabilities, the impact on individuals, the context or nature of the data, and the types of users accessing the data.

### **8.3.2 ADVERSARIAL AND ML SECURITY MODELS**

Adversarial data is the input to ML models that attackers intentionally create to manipulate the ML model into producing incorrect outputs [67]. For instance, valid input data can be corrupted by introducing minor perturbations, which can have a significant impact on the ML model, leading to the generation of incorrect outputs. Even a small perturbation is sufficient to cause misclassification in the target for an ML model [67].

The Limited-memory Broyden-Fletcher-Goldfarb-Shanno (LBFGS) [68] is a non-linear gradient-based numerical optimization algorithm that minimizes the perturbation of the input data. This algorithm can be used to

generate various adversarial attack test cases for testing machine learning models. However, it can be computationally expensive as it relies on the box constraint optimization method. Another adversarial model is the Fast Gradient Sign Method (FGSM) [69], which generates adversarial input test cases by minimizing the number of perturbations needed to produce incorrect classifications. This method is computationally less expensive, but it may require adding perturbations to every feature, increasing the pre-processing complexity. The Jacobian-based Saliency Map Attack (JSMA) [70] uses feature selection to minimize the number of modified features that cause incorrect classifications. It iteratively adds perturbations based on the saliency value, which increases computational complexity. DeepFool attack [71] is an input generation technique that minimizes the Euclidean distance between the perturbed and the actual inputs. Perturbations are added iteratively based on the decision boundaries estimated among the classes. DeepFool attacks can generate a variety of adversarial inputs. However, it lacks control of incorrect classifications since there are fewer perturbations due to the decision boundary estimation. It could be computationally intensive since so many iterations can occur for the perturbation completions. Carlini & Wagner Attack (C&W) [72] is an adversarial input generation model that does not use optimized box constraints with distinct objective functions. This method has proven the potential for the effective generation of adversarial input test samples and has better adversarial defenses. However, this could be computationally expensive since box constraints are not supported. Generative Adversarial Networks (GAN) [73] is an adversarial input generation model that has proven that it can induce various adversarial attacks where two Neural Networks (NN) compete. One neural network (NN) will act as a generator, while the other will act as a discriminator. The generator NN attempts to produce input test cases, while the discriminator NN continuously misclassifies them and distinguishes between the actual input test cases and those created by the generator. The generator NN always generates different input test cases compared to those used in the training process. GAN can be computationally expensive due to the significant synchronization required between the generator NN and discriminator NN.

### **8.3.3 PRIVACY-PRESERVING MACHINE LEARNING AND SECURE MACHINE LEARNING**

ML models can be applied in a wide range of applications, such as text prediction, suggestion generation, image prediction, video frame prediction,

and more. However, the issue of privacy arises when dealing with sensitive data in the dataset. It is crucial to protect individuals' privacy and earn their trust by ensuring that their sensitive personal information is never leaked.

Secure ML models are designed with the anticipation that an attacker may compromise the availability and integrity of a data-analytic system. Defensive distillation, adversarial training, and regularization are some of the main defensive measures used to build secure ML models. On the other hand, PPML aims to safeguard the sensitive information of individuals or entities, including industries or enterprises. Collaborative learning enables the sharing of data from various sources without data leakage. This can be achieved through cryptographic techniques and differential privacy (DP). Homomorphic Encryption (HE), garbled circuits, secret sharing, and secure processors are among the most popular cryptographic methods used to create PPML. LDP can be utilized in federated learning (FL) to enable model training on distributed datasets maintained by multiple parties. The perturbed response is the primary concept of LDP.

### **8.3.4 DIFFERENTIAL PRIVACY**

Here, we delve deep into methods such as DP to address data preservation in datasets. DP improves learning by providing stability, better ML models, and broad applicability in diverse areas of Artificial Intelligence (AI) with a secure background [74]. Let's consider a situation where we have a Query (Q) on a Dataset (D). The data sensitivity in D is determined based on the parameters defined in Section 1, and a reasonable amount of noise is added to adjust the sensitivity. Therefore, we can map the sensitivity as shown in Eqn.(1).

$$\text{Sensitivity}_i(F) = \max_{D, D'} |Q(D) - Q(D')|_i \quad (1)$$

where;  $i \in \{1,2\}$  and  $D'$  is considered as the neighboring dataset.

Suppose we have dataset D and perform the Query operation Q on it. We have knowledge of the location of X individuals who reside in Los Angeles. Now, let's say we add a new record to create a new dataset  $D'$ . As a result of the query, it is revealed that there are  $X+1$  individuals. If the location is in Los Angeles, we can infer that the newly added person's location is also in Los Angeles. This poses a risk as the attacker can deduce sensitive information based on background knowledge. To mitigate such a background knowledge attack, we can use Differential Privacy (DP) by introducing random noise

$R$  to the query results  $\{Q(D), Q(D')\}$  as shown in Eqns. (2) and (3). This ensures that the query results for  $D$  and  $D'$  are not specific to the attacker. Additionally, the results fall within a certain probability distribution range. Consequently, the attacker cannot determine which dataset the query result originates from, and the location of the person becomes ambiguous to the attacker. This ensures the privacy of individual data within the dataset.

$$S(D) = Q(D) + \text{Random\_Value} \quad (2)$$

$$S(D') = Q(D') + \text{Random\_Value} \quad (3)$$

Let us suppose that the dataset and the neighboring dataset  $D$  and  $D'$  with only one record are different. Consider the randomization algorithm  $S$  that changes the value by distorting the probability distribution. We also have to calculate a privacy budget ( $\epsilon$ ) which controls the degree of privacy protection. The smaller the  $\epsilon$ , the better data protection it provides. The Privacy Loss (PL) represented in 4 will be in absolute value since a greater positive value or a greater negative value will lead to a greater PL.

$$PL = \left| \ln\left(\frac{\text{Probability}[S(D) \in R]}{\text{Probability}[S(D') \in R]}\right) \right| \quad (4)$$

DP for the neighboring datasets  $\{D, D'\}$  with only one record different added with random noise to these two datasets  $S$  for all ( $R \subseteq \text{Range}(S)$ ), if as shown in Eqn. (5).

$$\begin{aligned} & \text{Probability}[S(D) \in R].\text{Probability}[S(D') \in R] \times e^{\epsilon} \\ &= \max_s [\ln\left(\frac{\text{Probability}[S(D) \in R]}{\text{Probability}[S(D') \in R]}\right)]\epsilon \end{aligned} \quad (5)$$

If it holds well, then  $S$  satisfies  $\epsilon$ -DP, where  $\text{Range}(S)$  is the value space of the random variable of the random algorithm  $S$  with the mapping result,  $R$ , as its subset. Therefore, for all  $S \subseteq \text{Range}(S)$ , it holds for all subsets of  $\text{Range}(S)$ . We may also represent it as in 6.

$$\begin{aligned} & \text{Probability}[S(D) = x].\text{Probability}[S(D') = x] \times e^{\epsilon} \\ &= \max_{x \in s} [\log\left(\frac{\text{Probability}[S(D) = x]}{\text{Probability}[S(D') = x]}\right)]\epsilon \end{aligned} \quad (6)$$

Hence;  $\{\epsilon, \sigma\}$ -DP introduced by Dwork Cynthia [19, 20] eased the DP algorithm by adding a small constant  $\delta$  and can be represented as in Eqn.(7).

$$\text{Probability}[S(D) \in R].\text{Probability}[S(D') \in R] \times e^{\epsilon} + \delta \quad (7)$$

### **8.3.5 PRIVACY-PRESERVING DISTRIBUTED MACHINE LEARNING AND ITS SIGNIFICANCE**

In the conventional centralized ML approach, the privacy of individual users' data or entities can be hindered since the entire data rests in the storage of a centralized server. DML preserves privacy by conducting the learning of an ML model from data owners' portable or wired devices themselves or in a coordinating server [75–77]. That is, the Privacy-preserving DML model will function in a distributed environment by preserving governance and regulatory aspects. Therefore, except for an encrypted data version, the original data never goes out of our device; hence, DML preserves the legal and ethical aspects of data privacy [78–81]. Ensemble, Split, and FL are the major DML techniques. This chapter focuses on privacy preservation via FL. Various attack methods, such as membership inference or linkage attacks, can exploit the vulnerabilities of ML models and the coordinating servers [82, 83]. Hence, the adversaries will be able to retrieve sensitive personal identification data. Hence, DML still needs additional steps to guarantee the utmost data privacy.

The following provides insight into various cryptographic methods involved in DML to attain privacy [84, 85].

- **Secure Multiparty Computation (SMC):** Initially proposed by Yao et al. [86], in which  $n$  participants collaboratively compute a secret ( $S$ ), where  $S = f(x_1, x_2, \dots, x_n)$ . Any participant will only know  $S$  and not any other information on private data  $x_i$ .
- **Homomorphic Encryption (HE):** Nowadays, Data Masking Language (DML) is advancing with Homomorphic Encryption (HE) strategies to provide data protection. In HE, both the collected data and the data used for training are encrypted, and only the data owner can decrypt them. HE is categorized into three types: (i) Partial Homomorphic Encryption, (ii) Somewhat Homomorphic Encryption (SWHE), and (iii) Fully Homomorphic Encryption (FHE). However, employing HE for large datasets is computationally intensive and practically challenging.

Despite these advancements, there are still security issues that need to be addressed in DML. For instance, a distributed privacy-preserving algorithm utilizes a data perturbation mechanism, which can be computationally expensive in terms of the time required to generate results with reasonable utility and privacy. In terms of efficiency, perturbed data can lead to better performance but may compromise privacy to some extent [87].

## 8.4 FEDERATED LEARNING AND PRIVACY

FL is a type of DML that is designed to enhance privacy. In DML, trained data or models can be shared among participants, whereas FL prohibits this by utilizing a central server coordinator for transfers. FL can be implemented across regions or organizations, while DML is limited to a specific region or single organization. FL processes large amounts of data while preserving privacy. FL is a technique that involves multiple devices collaborating to solve an ML problem, with the potential monitoring of a centralized server or provider. The raw data remains on the devices and is not transferred to other devices or the centralized server. Instead, the devices perform the learning process on the data and send scheduled updates for aggregation to another device, ultimately achieving the learning objective. The centralized server strictly monitors these update indentations and aggregations, and all updates are encrypted for security purposes. Since all the data remains solely on the devices, the encrypted update indentations ensure the privacy and security of the data used for learning. We aim to achieve transparency and trust in data privacy. As mentioned in Section 1.2, when handling sensitive data such as personally identifiable information, it is essential to adhere to the privacy policies set by the data ownership authority. However, there are instances where data privacy fails, leading to data leakage. The fundamental concept of Federated Learning (FL) revolves around ensuring the privacy of sensitive data by eliminating the need for raw data sharing. Cross-silo FL enables the distribution of data across pre-established silos, such as organizational or geographic boundaries. The typical number of participating clients in this setup ranges from 2 to 100. On the other hand, Cross-device FL involves a large number of clients (up to 1010 devices), but only a small percentage of those clients participate in each training round. In recent years, industries such as medicine and healthcare, finance, and manufacturing have greatly benefited from cross-silo FL.

### 8.4.1 SECURE FRAMEWORK OF FEDERATED LEARNING

- **Centralized Federated Learning Framework with Co-Ordinating Server:** FL has a framework consisting of a coordinating server (Figure 8.2) that includes a Secure Aggregation (SA) protocol for encrypting update indentation. FL utilizes the concept of FedAvg, where the centralized server calculates the average update from all user devices participating in FL. Subsequently, the SA procedure is

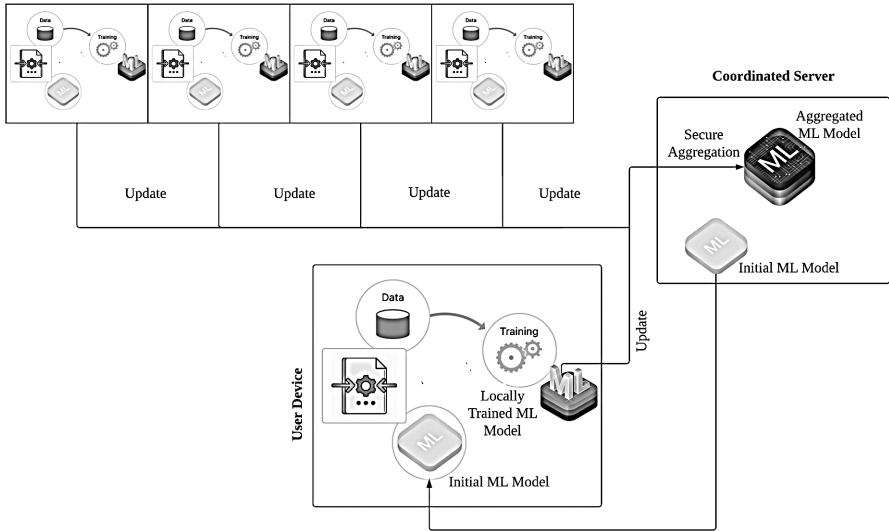
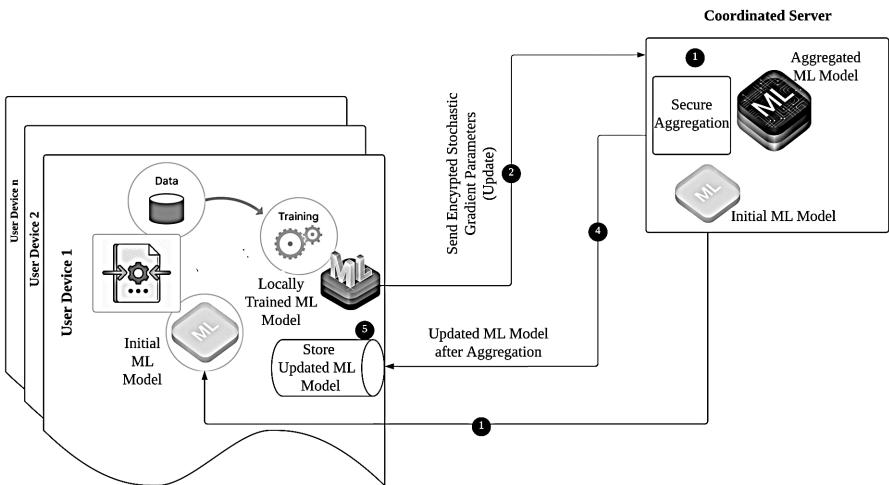
performed on the updates from user devices. The federated optimization algorithm, FedSGD, conducts a batch gradient calculation on randomly selected user devices in each round of communication. FedSGD is computationally efficient but requires multiple training rounds to generate reliable ML models. Essentially, in the FL framework, the process unfolds as follows.

- o All user devices participating in FL will receive the trained ML models.
- o All user devices will perform the training process using the data stored locally on each device.
- o All user devices will send the encrypted update information to the coordinating server.
- o All user devices are clustered and allocated to a group by the coordinating server. The server will then conduct a self-assessment of the update indentation from the group of devices in order to perform an update to the ML model.
- o Later, the coordinating server will handle the extensive and numerous training rounds and generate an updated ML model based on the results of the training process. This updated ML model will be sent to the devices in the cluster for on-device testing.

Privacy preservation techniques used on the server side include SMC, SA, and HE. The same applies to clients, along with batch-level DP and user-level DP.

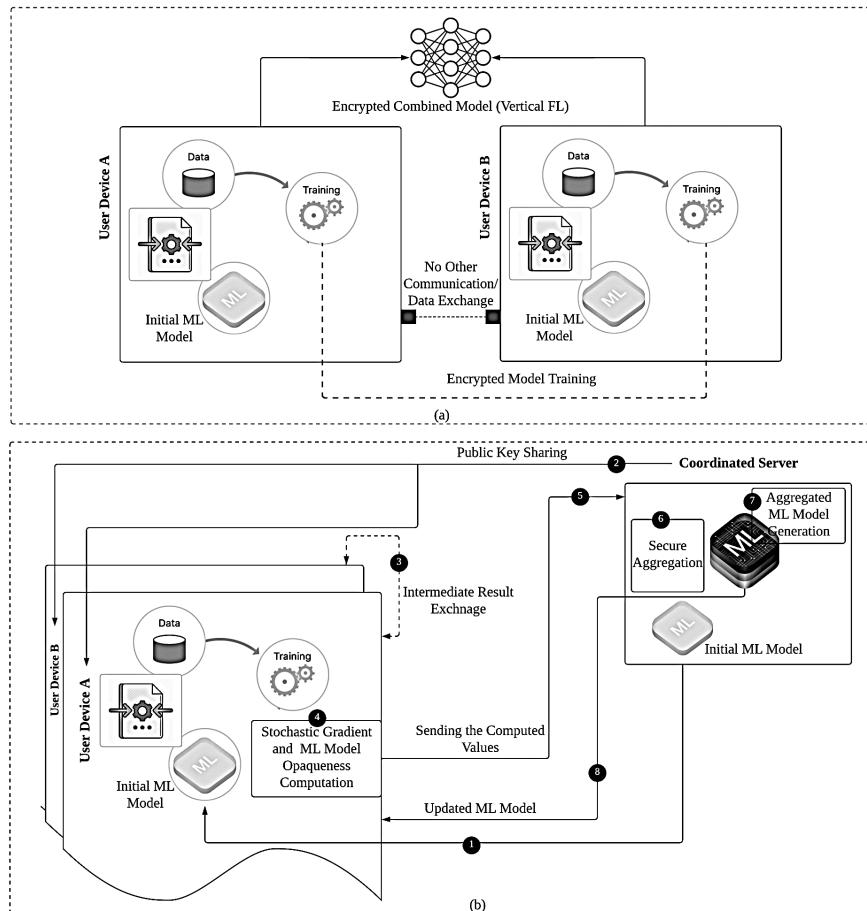
- **De-Centralized Federated Learning Framework:** During the absence of an aggregator and if the topology is highly scalable, centralized federated learning is replaced with decentralized federated learning. Each neighboring participant who communicates with another takes the role of coordinator via the polling method [89].

FL can be categorized into three groups based on distinct data distributions across numerous participants – Horizontal FL, Vertical FL, and Federated Transfer Learning. In Horizontal FL, datasets share the same data features but differ in user set. Here, only the local gradients parameter is shared, ensuring privacy protection. The framework of secure Horizontal FL is shown in Figure 8.3. Therefore, the user's data privacy is protected at the cost of some opaqueness in the ML model. Google has already presented a Horizontal FL method for Android phones. In this FL method, the user device updates the ML model parameters and uploads the updates to the Android cloud, training the combined ML model with other user devices. The coordinating server uses the SA procedure to protect the privacy of aggregated user updates [88].

**FIGURE 8.2** Secure framework of FL.**FIGURE 8.3** The secure framework of horizontal FL.

In Vertical FL, user sets remain the same but are applied to different datasets. For instance, airlines and hotels may share the same user dataset, but their datasets are distinct. In vertical FL, users cannot access the data or characteristics of other users' devices. The secure Vertical FL architecture is depicted in Figure 8.4, with Figure 8.4a illustrating the communication

between user devices and Figure 8.4b showing the communication between participating user devices and the coordinated server. In the case of Federated Transfer Learning, we work with a small set of datasets that have a limited number of identical users. The architecture of secure Federated Transfer Learning is shown in Figure 8.5.

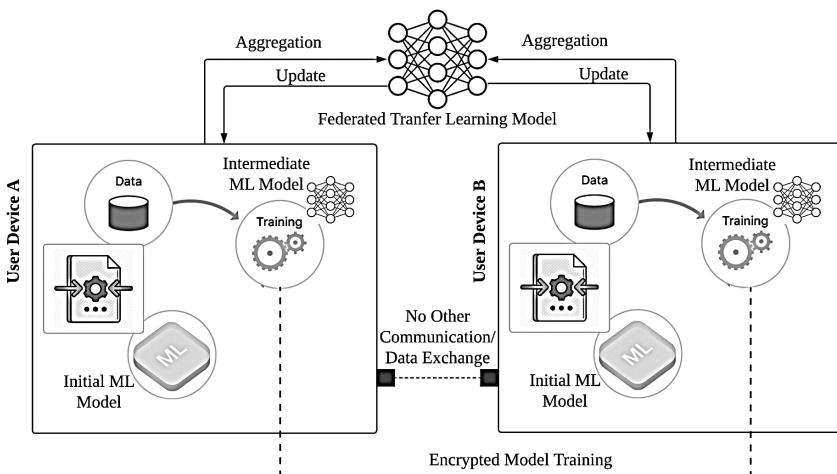


**FIGURE 8.4** The secure framework of vertical FL.

#### 8.4.2 SECURE MULTIPARTY COMPUTATION IN FEDERATED LEARNING

During the process of aggregation, gradients in FedSGD and/or local model weights in FedAvg are protected with the Secure Aggregation (SA) protocol. If a malicious server receives an update, it may attempt to construct private

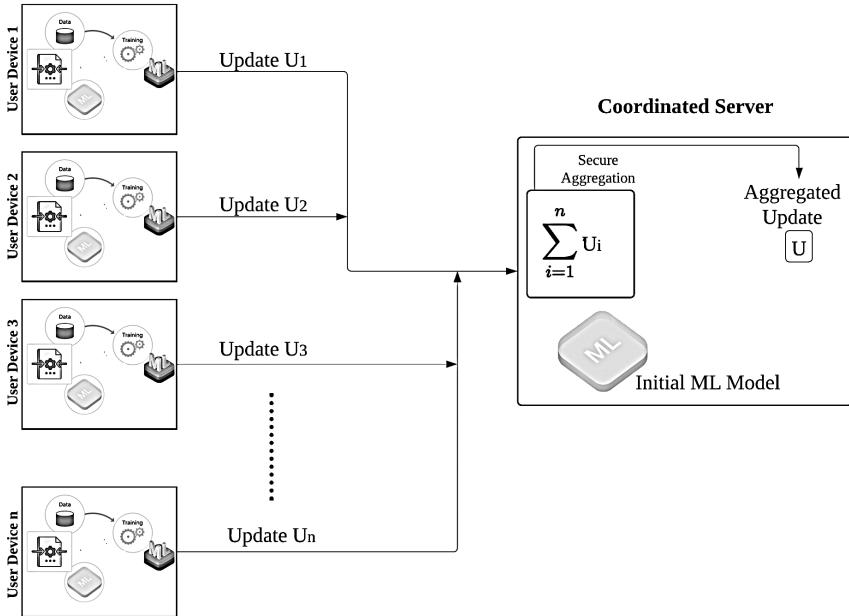
training data that can generate a similar update indentation. However, this approach will not be effective if the update indentation is averaged from many devices, as the nature of data from various devices is heterogeneous and sparse. In the SA protocol, the update vector is masked with a random vector of the same size. User devices participating in Federated Learning (FL) cooperate to generate random mask vectors and later distribute the mask vector within the cluster. Once the update with the mask vector is added, it is sent to the coordinating server. Similar to the concept of secret sharing, user devices use Diffie-Hellman (DH) key exchange for mutual agreement on the mask vector. Once the coordinating server receives the update indentation from each user device, it averages the updates together and removes the masked vectors. Figure 8.6 illustrates the process of the SA protocol, where Secure Multi-Party Computation (SMC) and SA combined provide privacy protection against reconstruction, membership inference, and model poisoning attacks.



**FIGURE 8.5** Secure framework of federated transfer learning.

#### 8.4.3 DIFFERENTIAL PRIVACY IN FEDERATED LEARNING

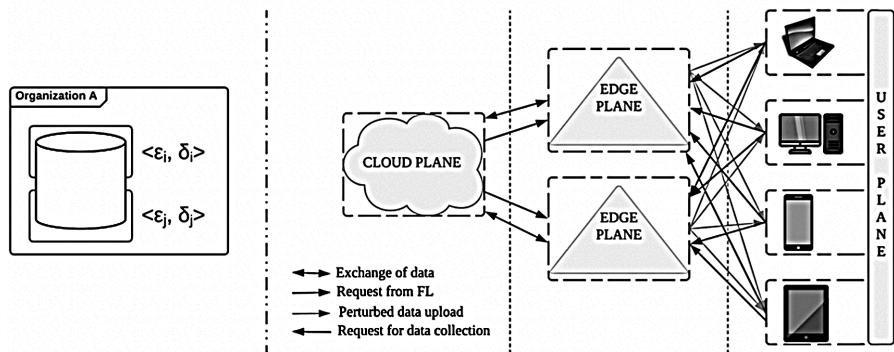
FL permits the ML model to work on the data locally on the device, and the concept of DP plays a significant role in data anonymization. In FL, the notion of DP addresses the issues related to data privacy concerns since FL does not provide an opportunity for the ML model to memorize anything about sensitive user data [73, 90, 91]. GDP and LDP are employed in FL to achieve privacy.



**FIGURE 8.6** Secure aggregation procedure.

- **GDP:** In GDP, let's consider that we have 'k' users with databases  $\{D_1, D_2, \dots, D_k\}$ . After training on their respective local devices, users upload their respective parameters to the cloud server. The server then calculates the average of the weights obtained from the end users and adds Gaussian noise with a mean of 0 and variance of  $\sigma^2$ . In the subsequent training phase, each user receives the averaged parameters from the server.
- **LDP:** It guarantees a higher level of privacy as the data owner perturbs data in their edge devices [16–18]. The mathematical computation of LDP is the same as that of DP given in Equations (4) and (5).
- **DP in Cross-Silo FL:** There are different methods followed for imbibing privacy in Cross-Silo FL.
- **Silo-Specific Item-Level Privacy [92]:** DP is distributed across mutually exclusive datasets in each silo  $k$  with a privacy budget of  $\{\epsilon_k, \delta_k\}$ . For example, if an organization A has two disjoint datasets  $i$  and  $j$ , then item level protection with DP is provided as shown in Figure 8.7(a).
- **Cross-Silo FL with LDP[93]:** For LDP, as shown in Figure 8.7(b), Cross-Silo FL is considered to have three planes: cloud, edge, and user planes. The LDP perturbation mechanism is applied in the user plane

to protect user data collected by the client. After converting user input into bit strings, the LDP perturbation algorithm generates random answers for each bit in each string. The edge plane computes original data using the reconstruction algorithm based on Bayes' theorem. Once the edge servers have recovered the data, FL operates between the cloud and edge servers. The edge servers receive the shared ML model from the cloud server and then execute ML algorithms (SGD) to compute model parameter updates locally.



**FIGURE 8.7** Differential privacy techniques in cross-silo federated learning include (a) silo-specific differential privacy; and (b) local differential privacy (LDP) on cross-silo [93].

#### 8.4.4 DATA UTILITY OPTIMIZATION IN FEDERATED LEARNING

Data utility is crucial for any privacy preservation mechanism and has an impact on end users who collaborate with data [90, 91]. The convergence performance of Federated Learning (FL) is affected by the number of users and iterations, as the accumulated noise level in such cases is high. To enhance data utility in Global Differential Privacy (GDP), dynamic adjustment of noise scale can be applied, where the collaborator verifies the global model and adjusts the noise accordingly. If the increase in verification accuracy is lower than the threshold, the noise scale is reduced by  $k$ . If ' $a$ ' represents accuracy, the noise scale can be computed as shown in Equation (8) [94]:

$$\sigma_t = \begin{cases} k\sigma_t; & a_t - a_{t-1} \leq \alpha \\ \sigma_t; & \text{otherwise} \end{cases} \quad (8)$$

For enhancing utility optimization in LDP, user devices will compute minimal random noise to perturb data before the SA protocol is executed.

The coordinating server checks for the noise added by the user devices and provides a better DP guarantee based on the total noise identified on the update indentation from all the user devices.

## 8.5 PRIVACY CHALLENGES AND THREATS IN FEDERATED LEARNING

Typically in FL, the hostile agent makes use of flaws to take control of the global model. Attackers target training data at rest and updated weights that are in transit. A semi-honest adversary model passively infers sensitive data from aggregators on training data or gradients, whereas malicious adversary models directly infer sensitive parameters from the global model. Malicious aggregators masquerade as benign servers, but they attempt to reveal record-level training samples of participants. The original training data can be recovered by an adversary aggregator without any prior information. The following are the major privacy challenges in FL.

1. **Poisoning Attack:** The probability of poisoning attacks is high in Florida as the model updates are gathered from a sizable clientele [95].
  - i. **Data Poisoning:** It is a type of poisoning attack in which attackers generate contaminated samples for training, thereby producing fabricated and falsified model parameters. When a rogue client injects malicious data into the client's local model, this is referred to as data injection, which is a subset of data poisoning.
  - ii. **Model Poisoning:** A malevolent party can alter the updated model before sending it for aggregation, resulting in a poisoned global model.
  - iii. **Data Modification:** These attacks tamper with data by merging two or more classes in the dataset. Random swapping of labels also results in data modification. The global model tends to misclassify targets in such cases.
2. **Byzantine Attack:** The malicious Byzantine individuals design their outputs to have a similar distribution on a global model, which makes it difficult to identify the attack [96].
3. **Membership Inference Attack:** In this attack, the information on the training dataset can be deduced through brute force or guesswork. Neural network training can be exploited to predict the original training data. Inference attacks are possible with a mixture

generative adversarial network (mGAN)-AI when directed towards the central server of federated learning.

4. **Watermark Attack:** Extract text records of training data that fall under Natural Language Processing (NLP) with FL. If a character sequence is given as input, FL combines the private records of the individual in the model to predict the next word.
5. **Property Inference Attack (PIA):** The goal of PIA is to extract valuable attributes that other participants may not be willing to disclose, and that may even be unrelated to the main task. For instance, if a person is trained to analyze gender, their accessories or wearables can be inferred from the images [97].

### **8.5.1 OPEN CHALLENGES IN FEDERATED LEARNING**

Challenges in federated learning (FL) include the trade-off between efficiency and privacy. Secure multi-party computations (SMCs) are computationally expensive and introduce additional run-time overhead. DP techniques also hinder training performance in FL. Although Verify Net is an FL framework for privacy preservation, the involvement of a central server for verification with each client is a time-consuming approach. GAN-based attacks are becoming predominant in FL, requiring a promising solution for inference-based attacks. There are methods to develop a robust FL model through ensemble adversarial training techniques, but they come at the cost of computation power and training time for adversarial samples [95]. FL is susceptible to poisoning and backdoor attacks when dealing with high-dimensional parameter vectors. The future of FL lies in proposing practically feasible solutions for collaboratively training models with heterogeneous architectures while adapting to privacy-preserving requirements. Building a decentralized framework for FL without the need for any server and randomly selecting each edge device as a coordinator is currently under research. Efficiency in terms of privacy and security needs to be evaluated in such a scenario.

## **8.6 CONCLUSION**

This chapter differentiates between PPML and Secure ML concepts. It discusses the importance of privacy and its implications in society, with a brief explanation of prominent privacy regulation laws. Later, the chapter sheds light on privacy issues posed by traditional ML algorithms. Various

threat modeling methods to evaluate data protection during each phase of ML are also elaborated in detail. Furthermore, the chapter discusses DML and its significance in data protection. Privacy preservation techniques in FL are explained, including cryptographic and DP techniques. DP in FL is conceptualized here with LDP and its difference with GDP. This chapter illustrates secure frameworks of FL differentiated by the type of user set and dataset. FL needs improvement with respect to data utility optimization. Additionally, we have elaborated on major threat models that can result in sensitive data spills. The chapter concludes with a few open challenges and research directions for privacy preservation using FL.

## KEYWORDS

- **data modification**
- **data poisoning**
- **data utility optimization**
- **federated learning**
- **privacy preservation**
- **property inference attack**
- **watermark attack**

## REFERENCES

1. Roh, Y., Heo, G., & Whang, S. E., (2021). A survey on data collection for machine learning: A big data–ai integration perspective. *IEEE Transactions on Knowledge and Data Engineering*, 33(4), 1328–1347.
2. Escalante, H. J., Yao, Q., Tu, W. W., Pillay, N., Qu, R., Yu, Y., & Houlsby, N., (2021). Guest editorial: Automated machine learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(09), pp. 2887–2890.
3. Whang, S. E., & Lee, J. G., (2020). Data collection and quality challenges for deep learning. *Proceedings of VLDB Endowment, ACM*, 13(12), 3429–3432.
4. Munappy, A. R., Bosch, J., Olsson, H. H., Arpteg, A., & Brinne, B., (2022). Data management for production quality deep learning models: Challenges and solutions. *Journal of Systems and Software, Science Direct*, 191, 111359.
5. Bentotahewa, V., Hewage, C., & Williams, J., (2022). The normative power of the GDPR: A case study of data protection laws of South Asian countries. *SN Computer Science*, 3(183), 1–18. Springer.

6. Truong, N. B., Sun, K., Lee, G. M., & Guo, Y., (2020). GDPR-compliant personal data management: A blockchain-based solution. *IEEE Transactions on Information Forensics and Security*, 15, 1746–1761.
7. State of California Department of Justice Office of the Attorney General; California Consumer Privacy Act. [Online]. Available: <https://oag.ca.gov/privacy/ccpa> (accessed on 25 January 2024).
8. Chauhan, P. S., & Kshetri, N., (2021). State of the practice in data privacy and security. *Computer*, 54(8), 125–132. IEEE.
9. Jain, A. K., Sahoo, S. R., & Kaubiyal, J., (2021). Online social networks security and privacy: Comprehensive review and analysis. *Complex & Intelligent Systems*, 7, 2157–2177. Springer.
10. Mirea, M., Wang, V., & Jung, J., (2019). The not so dark side of the darknet: A qualitative study. *Security Journal* (Vol. 32, pp. 102–118). Palgrave Macmillan and Springer.
11. Jercich, K., (2021). *Tens of Thousands of Patient Records Posted to Dark Web*. [Online]. Available: <https://www.healthcareitnews.com/news/tens-thousands-patient-records-posted-dark-web> (accessed on 25 January 2024).
12. Collier, K., (2021). *Hackers Post Detailed Patient Medical Records from Two Hospitals to the Dark Web*. [Online]. Available: <https://www.nbcnews.com/tech/security/hackers-post-detailed-patient-medical-records-two-hospitals-dark-web-n1256887> (accessed on 25 January 2024).
13. Tene, O., Evans, K., Gencarelli, B., Maloff, G., & Zanfir-Fortuna, G., (2019). GDPR at year one: Enter the designers and engineers. *IEEE Security & Privacy*, 17(6), 7–9.
14. Tan, A. Z., Yu, H., Cui, L., & Yang, Q., (2022). Towards personalized federated learning. *IEEE Transactions on Neural Networks and Learning Systems, Early Access*, 1–17.
15. McMahan, B., & Ramage, D., (2017). *Federated Learning: Collaborative Machine Learning Without Centralized Training Data*.<https://ai.googleblog.com/2017/04/federated-learning-collaborative.html> (accessed on 25 January 2024).
16. Xue, Q., Ye, Q., Hu, H., Zhu, Y., & Wang, J., (2022). DDRM: A continual frequency estimation mechanism with local differential privacy. *IEEE Transactions on Knowledge and Data Engineering, Early Access*, 1–14.
17. Nair, A. K., Jayakrushna, S., & Ebin, D. R., (2023). Privacy-preserving federated learning framework for IoMT-based big data analysis using edge computing. *Computer Standards & Interfaces*, 103720.
18. Wei, K., Li, J., Ding, M., Ma, C., Su, H., Zhang, B., & Poor, H. V., (2022). User-level privacy-preserving federated learning: Analysis and performance optimization. *IEEE Transactions on Mobile Computing*, 21(9), 3388–3401.
19. Dwork, C., (2006). Differential privacy. In: Bugliesi, M., Preneel, B., Sassone, V., & Wegener, I., (eds.), *Proceedings of Automata, Languages and Programming* (pp. 1–12). Springer Berlin Heidelberg.
20. Dwork, C., (2008). Differential privacy: A survey of results. In: Agrawal, M., Du, D., Duan, Z., & Li, A., (eds.), *Proceedings of Theory and Applications of Models of Computation* (pp. 1–19). Springer Berlin Heidelberg.
21. Domingo-Ferrer, J., Muralidhar, K., & Bras-Amorós, M., (2021). General confidentiality and utility metrics for privacy-preserving data publishing based on the permutation model. *IEEE Transactions on Dependable and Secure Computing*, 18(5), 2506–2517.
22. Ray, S., Palanivel, T., Herman, N., & Li, Y., (2021). Dynamics in data privacy and sharing economics. *IEEE Transactions on Technology and Society*, 2(3), 114–115.

23. Tsao, M., Yang, K., Zoepf, S., & Pavone, M., (2022). Trust but verify: Cryptographic data privacy for mobility management. *IEEE Transactions on Control of Network Systems*, 9(1), 50–61.
24. Ding, J., & Ding, B., (2022). Interval privacy: A framework for privacy-preserving data collection. *IEEE Transactions on Signal Processing*, 70, 2443–2459.
25. Domingo-Ferrer, J., & Blanco-Justicia, A., (2020). Privacy-preserving technologies. In: *Proceedings of The Ethics of Cyber Security*, Springer International Publishing, 279–297.
26. Yao, L., Chen, Z., Wang, X., Liu, D., & Wu, G., (2021). Sensitive label privacy preservation with anatomization for data publishing. *IEEE Transactions on Dependable and Secure Computing*, 18(2), 904–917.
27. Girka, A., Terziyan, V., Gavriushenko, M., & Gontarenko, A., (2021). Anonymization as homeomorphic data space transformation for privacy-preserving deep learning. *Procedia Computer Science, Science Direct*, 180, 867–876.
28. Rao, P. R. M., Krishna, S. M., & Kumar, A. P. S., (2018). Privacy preservation techniques in big data analytics: A survey. *Journal of Big Data, Springer*, 5(33), 1–12.
29. Jin, F., Hua, W., Francia, M., Chao, P., Orlowska, M., & Zhou, X., (2022). A survey and experimental study on privacy-preserving trajectory data publishing. *IEEE Transactions on Knowledge and Data Engineering, Early Access*, 1–19.
30. Wong, R. C. W., Fu, A. W. C., Wang, K., & Pei, J., (2009). Anonymization-based attacks in privacy-preserving data publishing. *ACM Transactions on Database Systems*, 34(8), 1–46.
31. Narayanan, A., & Shmatikov, V., (2009). De-anonymizing social networks. In: *Proceedings of 30<sup>th</sup> IEEE Symposium on Security and Privacy (S&P)*(pp. 173–187). IEEE
32. Eshun, S. N., & Palmieri, P., (2022). Two de-anonymization attacks on real-world location data based on a hidden Markov model. In: *Proceedings of IEEE European Symposium on Security and Privacy Workshops (EuroSPW)*(pp. 1–9). IEEE.
33. Wang, H., Li, Y., Gao, C., Wang, G., Tao, X., & Jin, D., (2021). Anonymization and de-anonymization of mobility trajectories: Dissecting the gaps between theory and practice. *IEEE Transactions on Mobile Computing*, 20(3), 796–815.
34. Gambs, S., Killijian, M., & Miguel, N. D. P. C., (2014). De-anonymization attack on geolocated data. *Journal of Computer and System Sciences, Science Direct*, 80(8), 1597–1614.
35. Olatunji, I. E., Rauch, J., Katzensteiner, M., & Khosla, M., (2021). *A Review of Anonymization for Healthcare Data*. CoRR, abs/2104.06523.
36. Merener, M. M., (2012). Theoretical results on de-anonymization via linkage attacks. *ACM Transactions Data Privacy*, 5(2), 377–402.
37. Truex, S., Liu, L., Gursoy, M. E., Yu, L., & Wei, W., (2021). Demystifying membership inference attacks in machine learning as a service. *IEEE Transactions on Services Computing*, 14(6), 2073–2089.
38. Sweeney, L., (2002). K-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(5), 557–570.
39. Machanavajjhala, A., Kifer, D., Gehrke, J., & Venkitasubramaniam, M., (2007). L-diversity: Privacy beyond k-anonymity. *ACM Transactions on Knowledge Discovery from Data*, 1(1), 3–es.
40. Li, N., Li, T., & Venkatasubramanian, S., (2007). t-closeness: Privacy beyond k-anonymity and l-diversity. In: *Proceedings of IEEE 23<sup>rd</sup> International Conference on Data Engineering* (pp. 106–115). IEEE

41. Soria-Comas, J., & Domingo-Ferrer, J., (2013). Differential privacy via t-closeness in data publishing. In: *Proceedings of 11<sup>th</sup> Annual Conference on Privacy, Security and Trust* (pp. 27–35). IEEE.
42. Xiao, X., & Tao, Y., (2007). M-invariance: Towards privacy preserving re-publication of dynamic datasets. In: *Proceedings of ACM SIGMOD International Conference on Management of Data*. (pp. 689–700).
43. Machanavajjhala, A., & Gehrke, J., (2009). Randomization methods to ensure data privacy. In: *Proceedings of Encyclopedia of Database Systems* (pp. 2319–2324). Springer, Boston, MA.
44. Statista, *Compromised Data Records in Selected Data Breaches as of January 2021*. [Online]. Available: <https://www.statista.com/statistics/290525/cyber-crime-biggest-online-data-breaches-worldwide/> (accessed on 25 January 2024).
45. IBM. *Cost of a Data Breach 2022 A Million-Dollar Race to Detect and Respond*. [Online]. <https://www.ibm.com/reports/data-breach> (accessed on 25 January 2024).
46. Verizon; *2022 Data Breach Investigations Report*. [Online]. <https://www.verizon.com/business/resources/reports/dbir/> (accessed on 25 January 2024).
47. Varonis, (2022). *89 Must-Know Data Breach Statistics*. [Online]. Available: [Https://www.varonis.com/blog/data-breach-statistics](https://www.varonis.com/blog/data-breach-statistics) (accessed on 25 January 2024).
48. US Data Breach Notification Law Interactive Map | BakerHostetler (bakerlaw.com).
49. Termly; *75 Biggest Data Breaches, Hacks, and Exposures* [2022 Update]. [Online]. <https://termly.io/resources/templates/privacy-policy-template/> (accessed on 25 January 2024).
50. Alizadeh, F., Jakobi, T., Boden, A., Stevens, G., & Boldt, J., (2020). GDPR reality check—claiming and investigating personally identifiable data from companies. In: *Proceedings of IEEE European Symposium on Security and Privacy Workshops (EuroSPW)*. IEEE (pp. 120–129).
51. Legislation.gov.uk; *Data Protection Act 1998*. [Online]. Available: <https://www.legislation.gov.uk/ukpga/1998/29/contents> (accessed on 25 January 2024).
52. Microsoft; *Secure Data With Zero Trust*. [Online]. Available: <https://docs.microsoft.com/en-us/security/zero-trust/deploy/data> (accessed on 25 January 2024).
53. Liu, B., Ding, M., Shaham, S., Rahayu, W., Farokhi, F., & Lin, Z., (2021). When machine learning meets privacy: A survey and outlook. *ACM Computing Surveys*, 54(2), 1–36.
54. Liu, B., Ding, M., Zhu, T., Xiang, Y., & Zhou, W., (2018). Adversaries or allies? Privacy and deep learning in the big data era. *Concurrency and Computation: Practice and Experience*, Vol. 31, p. e5102). Wiley.
55. Gursoy, M. E., Inan, A., Nergiz, M. E., & Saygin, Y., (2017). Privacy-preserving learning analytics: Challenges and techniques. *IEEE Transactions on Learning Technologies*, 10(1), 68–81.
56. Geib, J., Santos, B., Berry, D., Baldwin, M., & Kess, B., (2022). Microsoft threat modeling tool threats. <https://learn.microsoft.com/en-us/azure/security/develop/threat-modeling-tool-threats> (accessed on 25 January 2024).
57. Marshall, D., & Viviano, A., (2022). *Threat Modeling for Drivers*. <https://learn.microsoft.com/en-us/windows-hardware/drivers/driversecurity/threat-modeling-for-drivers> (accessed on 25 January 2024).
58. UcedaVelez, T., (2021). *Real-World Threat Modeling Using the Pasta Methodology*. <https://versprite.com/blog/what-is-pasta-threat-modeling/> (accessed on 25 January 2024).
59. LinddunPrivacy Engineering, (2021). <https://www.nist.gov/privacy-framework/linddun-privacy-threat-modeling-framework> (accessed on 25 January 2024).

60. *Common Vulnerability Scoring System (CVSS)*, (2021). <https://nvd.nist.gov/vuln-metrics/cvss> (accessed on 25 January 2024).
61. Schneier, B., (2021). *Attack Trees: Modeling Security Threats*. [https://www.schneier.com/academic/archives/1999/12/attack\\_trees.html](https://www.schneier.com/academic/archives/1999/12/attack_trees.html) (accessed on 25 January 2024).
62. Cleland-Huang, J., (2014). How well do you know your personae non gratae? *IEEE Software*, 31(4), 28–31.
63. SEI, Carnegie Mellon University, (2018). *Threat Modeling: 12 Available Methods*. <https://insights.sei.cmu.edu/blog/threat-modeling-12-available-methods/> (accessed on 25 January 2024).
64. Mead, N., & Shull, F., (2018). *The Hybrid Threat Modeling Method*. <https://insights.sei.cmu.edu/blog/the-hybrid-threat-modeling-method/> (accessed on 25 January 2024).
65. *Trike: Open-Source Security Threat Modeling*, (2012). <http://www.octotrike.org/> (accessed on 25 January 2024).
66. Fruhlinger, J., (2020). *Threat Modeling Explained: A Process for Anticipating Cyber Attacks*. <https://www.cscoonline.com/article/3537370/threat-modeling-explained-a-process-for-anticipating-cyber-attacks.html> (accessed on 25 January 2024).
67. Rey, R. W., Anqi, X., Ousmane, D., & Archy De, B., (2019). *Adversarial Examples in Modern Machine Learning: A Review*. CoRR, abs/1911.05268, 1–97.
68. Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., & Fergus, R., (2013). *Intriguing Properties of Neural Networks*. CoRR, abs/1312.6199, 1–10.
69. Geirhos, R., Temme, C. R. M., Rauber, J., Schütt, H. H., Bethge, M., & Wichmann, F. A., (2018). Generalization in humans and deep neural networks. In: *Proceedings of 32<sup>nd</sup> International Conference on Neural Information Processing Systems (NIPS)*(pp. 7549–7561). Curran Associates Inc.
70. Papernot, N., McDaniel, P. D., Jha, S., Fredrikson, M., Celik, Z. B., & Swami, A., (2015). *The Limitations of Deep Learning in Adversarial Settings*. CoRR, abs/1511.07528, 1–16.
71. Moosavi-Dezfooli, S. M., Fawzi, A., & Frossard, P., (2016). Generalization in humans and deep neural networks. In: *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*(pp. 2574–2582). IEEE.
72. Carlini, N., & Wagner, D. A., (2016). *Towards Evaluating the Robustness of Neural Networks*. CoRR, abs/1608.04644, 1–19.
73. Xiao, C., Li, B., Zhu, J., He, W., Liu, M., & Song, D., (2018). *Generating Adversarial Examples with Adversarial Networks*. CoRR, abs/1801.02610, 1–8.
74. Zhu, T., Ye, D., Wang, W., Zhou, W., & Yu, P. S., (2022). More than privacy: Applying differential privacy in key areas of artificial intelligence. *IEEE Transactions on Knowledge and Data Engineering*, 34(6), 2824–2843.
75. Peteiro-Barral, D., & Guijarro-Berdiñas, B., (2013). A survey of methods for distributed machine learning. *Progress in Artificial Intelligence*, 2, 1–11. Springer.
76. Verbraeken, J., Wolting, M., Katzy, J., Kloppenburg, J., Verbelen, T., & Rellermeyer, J. S., (2020). A survey on distributed machine learning. *ACM Computing Surveys*, 53(2), 30:1–30:33.
77. Bonawitz, K., Kairouz, P., McMahan, B., & Ramage, D., (2021). Federated learning and privacy: Building privacy-preserving systems for machine learning and data science on decentralized data. *Queue, ACM*, 19(5), 87–114.
78. Gai, K., Qiu, M., & Zhao, H., (2021). Privacy-preserving data encryption strategy for big data in mobile cloud computing. *IEEE Transactions on Big Data*, 7(4), 678–688.

79. Lu, Y., & Zhu, M., (2018). Privacy-preserving distributed optimization using Homomorphic encryption. *Automatica*, 96, 314–325. Science Direct.
80. Yang, Y., Xiao, X., Cai, X., & Zhang, W., (2020). A secure and privacy-preserving technique based on contrast-enhancement reversible data hiding and plaintext encryption for medical images. *IEEE Signal Processing Letters*, 27, 256–260.
81. Jiang, Y., Zhang, K., Qian, Y., & Zhou, L., (2022). Anonymous and efficient authentication scheme for privacy-preserving distributed learning. *IEEE Transactions on Information Forensics and Security*, 17, 2227–2240.
82. Shokri, R., Stronati, M., Song, C., & Shmatikov, V., (2017). Membership inference attacks against machine learning models. In: *Proceedings of IEEE Symposium on Security and Privacy (S&P)*(pp. 3–18). IEEE.
83. Hu, H., Salcic, Z., Dobbie, G., & Zhang, X., (2022). Membership inference attacks on machine learning: A survey. *ACM Computing Surveys*, 54(235), 1–37.
84. Truong, N., Sun, K., Wang, S., Guittot, F., & Guo, Y., (2021). Privacy preservation in federated learning: An insightful survey from the GDPR perspective. *Computers Security*, 110, 102402.
85. Zhang, L., Xu, J., Vijayakumar, P., Sharma, P. K., & Ghosh, U., (2022). Homomorphic encryption-based privacy-preserving federated learning in IoT-enabled healthcare system. In: *IEEE Transactions on Network Science and Engineering, Early Access*, 1–17.
86. Koutsos, V., Papadopoulos, D., Chatzopoulos, D., Tarkoma, S., & Hui, P., (2021). Agora: A privacy-aware data marketplace. In: *IEEE Transactions on Dependable and Secure Computing, Early Access* (pp. 1–14).
87. Okkalioglu, B. D., Okkalioglu, M., Koc, M., & Polat, H., (2015). A survey: Deriving private information from perturbed data. *Artificial Intelligence Review* (Vol. 44, No. 4, pp. 547–569). Kluwer Academic Publishers and Springer.
88. So, J., Guler, B., & Avestimehr, A. S., (2021). Byzantine-resilient secure federated learning. *IEEE Journal on Selected Areas in Communications*, 39, 2168–2181.
89. Lyu, L., Yu, J., Nandakumar, K., Li, Y., Ma, X., Jin, J., Yu, H., & Ng, K. S., (2021). Towards fair and privacy-preserving federated deep models. *IEEE Transactions on Parallel and Distributed Systems*, 31(11), 2524–2541.
90. Wei, K., et al., (2020). Federated learning with differential privacy: Algorithms and performance analysis. *IEEE Transactions on Information Forensics and Security*, 15, 3454–3469.
91. McMahan, B., & Thakurta, A., (2022). *Federated Learning with Formal Differential Privacy Guarantees*. <https://ai.googleblog.com/2022/02/federated-learning-with-formal.html> (accessed on 25 January 2024).
92. Kanani, P., Marathe, V. J., Peterson, D., Harpaz, R., & Bright, S., (2021). Private cross-silo federated learning for extracting vaccine adverse event mentions. In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases* (pp. 490–505). Cham: Springer International Publishing (p. 23).
93. Wang, C., Wu, X., Liu, G., Deng, T., Peng, K., & Wan, S., (2022). Safeguarding cross-silo federated learning with local differential privacy. *Digital Communications and Networks*, (Vol. 8, No. 4, pp. 446–454). Science Direct.
94. Zhu, H., Zhang, H., & Jin, Y., (2021). From federated learning to federated neural architecture search: A survey. *Complex Intelligent Systems* (Vol. 7, pp. 639–657). Springer.

95. Nair, A. K., Ebin, D. R., & Jayakrushna, S., (2023). A robust analysis of adversarial attacks on federated learning environments. *Computer Standards & Interfaces*, 103723.
96. Li, Q., et al., (2021). A survey on federated learning systems: Vision, hype, and reality for data privacy and protection. *IEEE Transactions on Knowledge and Data Engineering, Early Access*, 1–20.
97. Liu, P., Xu, X., & Wang, W., (2022). Threats, attacks, and defenses to federated learning: Issues, taxonomy, and perspectives. *Cybersecurity*, 5(4), 1–19. Springer.

## CHAPTER 9

---

# Analyzing Federated Learning From a Security Perspective

AKARSH K. NAIR, JAYAKRUSHNA SAHOO, and EBIN DENI RAJ

*Department of Computer Science and Engineering, Indian Institute of Information Technology, Kottayam, Kerala, India*

---

### ABSTRACT

In the current technological scenario, privacy and security-related issues have been a significant concern for most companies working with vast volumes of data with high monetary value. User expectations associated with privacy have also increased drastically, forcing companies and service providers to shift to methodologies that ensure enhanced user and data privacy. Based on these issues, traditional artificial intelligence (AI) has also been transitioning from centralized to decentralized approaches as part of measures to preserve privacy. One such approach is federated learning, a type of distributed learning paradigm that aims to improve privacy by localizing data storage and conducting on-site training, thereby minimizing data transfer and reducing the risk of privacy breaches. However, even federated learning is not foolproof and is susceptible to security breaches. Adversarial attacks, such as poisoning attacks and leakage attacks, can compromise the performance of the model. Moreover, advancements in AI-based technologies have shown that data leakages in such systems can lead to the reconstruction of private data and models. This article provides a comprehensive analysis of various privacy issues associated with federated learning and presents proven methodologies that serve as preventive measures in adversarial scenarios. We discuss the effectiveness of attacks such as poisoning attacks, reconstruction attacks, backdoor attacks, and GAN-based attacks in federated learning systems.

Additionally, we explore future research directions to further enhance the security of this technology.

## 9.1 INTRODUCTION

In the current scenario, artificial intelligence (AI)-based applications are being overtaken by data-driven machine learning approaches. Such extensive reliance on data has raised concerns associated with these approaches. These challenges can be categorized into three types. The primary challenge is related to data privacy, as AI techniques deal with various data types, including highly personalized and sensitive data in healthcare and finance applications. This necessitates the implementation of exclusive privacy-preserving mechanisms. With the increasing awareness of privacy breaches, regulatory measures like the General Data Protection Regulation (GDPR) have been introduced to ensure consistent privacy across all applications. The second challenge is ensuring data availability, which is a significant research problem in AI due to storage constraints, communication issues, and processing bottlenecks. This has led to discussions about transitioning from a centralized framework to a distributed one, particularly for learning applications involving lightweight devices like IoT and edge computing systems. The third challenge is to develop efficient methodologies for data processing and optimal data utilization without compromising security and privacy. In certain instances, centralized systems just don't provide enough resources or ample working environment for certain learning algorithms. However, the sensitive nature of the data also prevents the user from transferring the data to a third-party system. Thus, certain security measures and regulations need to be embedded into the distributed systems to tackle such instances and deal with scenarios where additional privacy requirements need to be satisfied. With distributed data storage, issues relating to non-uniformity in data distribution arise, for which solutions should be devised.

Federated Learning is an emerging distributed learning methodology proposed as a prospective solution to all three challenges stated above when working on centralized learning systems. Such systems are built on a federated approach, giving prime concentration to privacy preservation alongside tackling issues associated with conventional distributed learning [1]. The final model generated in such a system is through a collaborative training procedure where training data is always kept in-house, and model parameters are shared for optimizations. Such an aspect of data locality considerably reduces concerns related to unauthorized access to private data.

Traditional machine learning approaches are susceptible to adversarial interference from various sources, including adversarial attacks, malicious users, intentional and unintentional data leaks, and more. These factors primarily aim to compromise the security and privacy of the system, thereby disrupting its normal functioning. Since federated learning (FL) systems are also based on such learning approaches, they encounter similar challenges. Although most adversaries share similar motives, their methods of attack vary significantly. Some attacks primarily focus on data manipulation, which can only be achieved in FL through data leaks and the presence of adversarial clients. Such attacks typically undermine the integrity of the system and pose significant threats. Additionally, over time, several other vulnerabilities have been identified in FL systems, and various defensive measures have been proposed. The extensive body of literature on this subject has motivated us to undertake a study where we aim to present various privacy and security concerns in FL, along with a comprehensive discussion of adversarial scenarios and the proposed defensive measures.

## 9.2 INTRODUCTION TO FEDERATED LEARNING

In this section, we will discuss some of the basic terminologies associated with federated learning. We will compare and contrast FL with traditional learning methodologies. We will delve into major terminologies related to federated learning, including basic architecture, network topology, model aggregation principles, and more.

### 9.2.1 FROM TRADITIONAL TO FEDERATED APPROACH

The classical machine learning approach works based on a centralized mechanism, meaning that the system comprises a single model generated from a single source of data, usually stored in a central system or server. Although models generated through this approach can function effectively, the framework as a whole has several shortcomings. Dealing with such huge volumes of data requires highly complex systems with sophisticated communication channels to ensure effective model generation. Additionally, storing such large amounts of data in a single location also raises security concerns, as a compromised server can result in significant economic losses. To overcome these issues associated with traditional learning approaches, a distributed learning paradigm was initially proposed. The federated learning

approach can be considered a successor to the distributed approach with numerous updates. The initial proposal of federated learning was made by McMahan et al. [2] in 2016. In contrast to distributed learning, federated learning mainly focuses on a collaborative approach to model generation. Initially, the training data is distributed across multiple devices, or existing data on various devices is utilized for training. The central server broadcasts the model to individual client devices, which perform local training on-site. After training, the model updates are transferred back to the central server for aggregation using procedures like federated averaging. Training data remains stored on local devices throughout their lifetime, ensuring privacy and eliminating security concerns. Additionally, the reduced data transfer in each iteration results in lower communication costs. These additional benefits make federated learning an ideal choice for applications that require enhanced security, such as healthcare and finance-based applications.

### **9.2.2 USE CASES OF FEDERATED LEARNING**

One of the initial large-scale applications of federated learning was based on the predictive keyboard for Android smartphones released by Google. The application utilized the computational power of mobile devices, using on-device data for AI training. The gradients obtained from locally generated models were periodically transferred to a central server, where an updated version of the existing model was generated and then distributed to individual users. The main objective of the application was to optimize word prediction accuracy on devices through this process. The overall accuracy of the application could be significantly improved due to the large number of users utilizing the service, which increased the cumulative volume of data, even though the data on each device was relatively low. If the same application were to be run in a centralized manner, every individual's typing data would need to be transferred to the central server for training, which raises concerns about data privacy [3]. Additionally, such applications also inherit the demerits and limitations of traditional ML methodologies. In contrast to federated learning, traditional ML can never generate a model with the same performance efficiency under identical parameters. It is never easy to fit a fully-fledged DL or machine learning model into a small-scale device as done in FL. Similarly, traditional approaches require a higher volume of data which also makes it the least preferred choice for lightweight devices.

Healthcare is another domain where Federated Learning (FL) has found strong applicability in recent times. As data locality is a prime feature of FL,

security concerns associated with medical data can be eradicated. During the COVID-19 pandemic, the healthcare industry witnessed a huge shift from traditional to FL-based applications as patients started using more wearable devices and home-based smart healthcare systems, which by default run in distributed frameworks.

### **9.2.3 SYSTEM ARCHITECTURE**

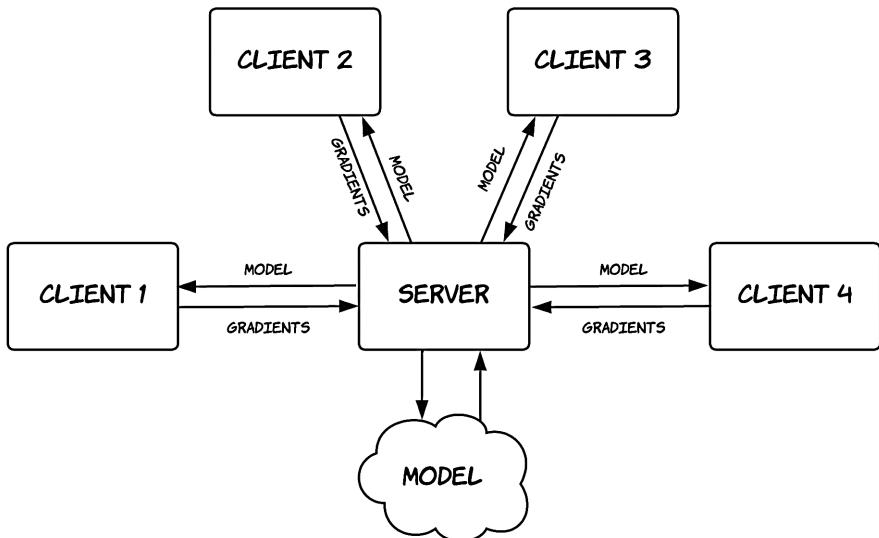
Even though the generalized architecture of federated learning is distributed, the system architecture is further divided into two types depending on the role of the server. The two types of FL classifications are centralized FL and decentralized FL. Both methodologies will be discussed in detail in the upcoming section.

#### *9.2.3.1 CENTRALIZED FL*

The centralized FL framework was the initially proposed framework for FL systems. As the name suggests, the system architecture is composed of a central device (server) and several other devices (clients). The communication between the client devices is channeled through the server exclusively. The central model is initially sent to the client devices by the server, and the model aggregation also takes place at the server itself. The server acts as a coordinator of the training procedure in such an architecture. Figure 9.1 illustrates the centralized FL system architecture. The model generation in such systems can be divided into three phases:

- **Global Model Broadcasting:** In the centralized federated learning methodology, the server initially broadcasts the global model to the clients to initiate local model training. The model is then further optimized in each round through model aggregation.
- **Local Model Generation:** Before the training begins, a set of client devices is initialized. Once the total number is initialized, the global model is broadcasted to every single device participating in the training. They train the received model using the local data to fine-tune the parameters. The obtained gradients are then transferred to the server device for aggregation.
- **Model Aggregation:** Once the updated gradients from the local devices are received at the central server, aggregation is performed where all the similarly received gradients are subjected to certain

procedures. After these procedures, an optimized global model is generated. There are several procedures currently being employed for aggregation, such as federated averaging. The optimized global model is then sent back to the clients, and the previous steps are iteratively performed until the model converges.



**FIGURE 9.1** Centralized federated learning architecture.

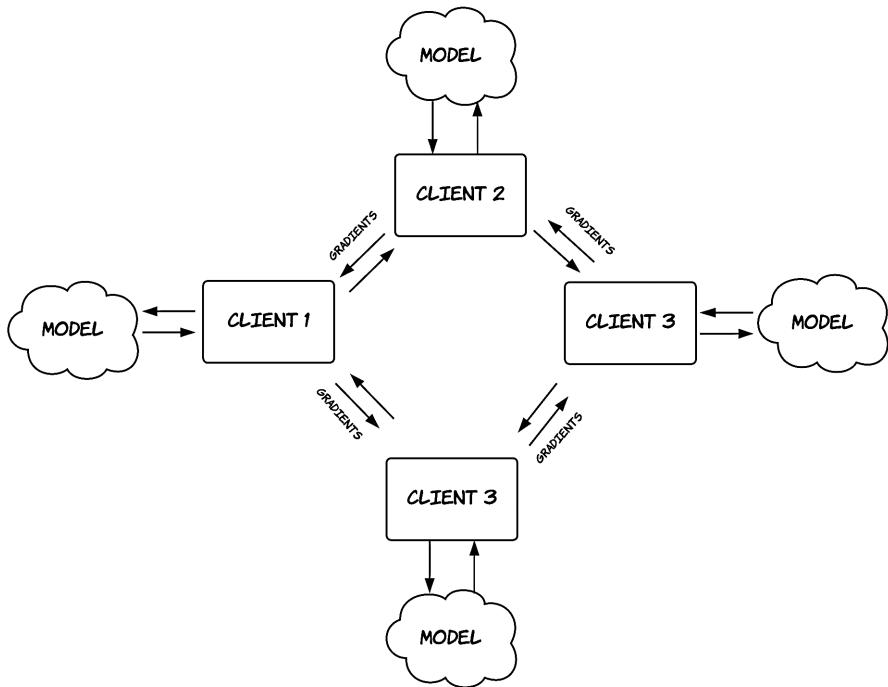
### 9.2.3.2 DECENTRALIZED FL

As the name suggests, decentralized FL employs a fully decentralized approach, meaning that every device in the system has equal status and there is no server-client separation. The network operates on a peer-to-peer strategy, and the aggregation is collaboratively performed by the clients themselves. Since a centralized coordinator is absent in such networks, communication between individual devices is key to ensuring the performance of the system [4]. Figure 9.2 illustrates the decentralized FL system architecture.

### 9.2.4 FL AGGREGATION METHODOLOGIES

Model aggregation is performed as a step to optimize the global model compared to the previous iteration. Most of the aggregation techniques

mainly employ averaging-based methodologies for combining the gradients. In this section, we will present some of the benchmark aggregation approaches in FL.



**FIGURE 9.2** Decentralized federated learning architecture.

- The Federated averaging (FedAvg) algorithm was initially proposed as an aggregation algorithm for FL systems. It is considered one of the default approaches in centralized FL. Most of the existing literature revolves around this algorithm, including the vanilla FL. In this approach, the received gradients are simply averaged to generate a set of weights that represent the newly generated model. Therefore, this approach is also referred to as gradient descent on the FL server [5]. The widespread popularity of this model has led to several extensions and updates of the federated averaging algorithm being proposed. Additionally, several optimization algorithms have been integrated into the classical FedAvg as part of optimization attempts.
- A generalized version of FedAvg was later proposed under the name of FedProx [6]. The FedProx approach takes into consideration the

individual work done by each client in model generation. It is ideally suited for heterogeneous environments where other factors such as data distribution, device energy status, availability of computational resources, communication bandwidth, and so on are also considered during aggregation.

- Another popular FL aggregation approach is the federated matched averaging approach (FedMa). The algorithm employs a wide aggregation approach where it attempts to match inner components at the neuron level. The proposed approach is highly specialized and only applicable to neural networks, thus making it useless for federated machine learning-based scenarios [7]. Even in the case of neural networks, its application is limited to basic network architectures such as conventional neural networks and recurrent neural network (RNN) architectures such as LSTM. This approach is also ideally suited for heterogeneous environments and guarantees better results than its predecessors for neural network-based models.

### **9.3 SECURITY ISSUES ASSOCIATED WITH FEDERATED LEARNING**

In any established federated learning (FL) system, the number of client devices participating in the training will be very high. This always opens up the scope for security breaches, as clients with adversarial intent may join such systems. This makes the system prone to various sorts of adversarial attacks on clients, servers, or during communication. Thus, when developing frameworks associated with FL, certain security measures and architectural principles should be followed to ensure system security, along with confidentiality and integrity. This article will attempt to present some of the major vulnerabilities in FL, followed by the major security threats as well. We will also attempt to identify some of the existing defensive measures for individual attack scenarios.

#### **9.3.1 VULNERABILITY IN FEDERATED ENVIRONMENT**

Vulnerability usually refers to the state of existence when a system is exposed to the possibility of being attacked with malicious or curious intent. Identifying every source of vulnerability and isolating them by improving defensive measures is mandatory for every system, irrespective of its architecture, and FL is no exception. We attempt to discuss some of the literature available on

vulnerability analysis on FL. Such vulnerability can be classified depending on the reasons for its occurrence. A major reason can be compromised communication channels. Any sort of communication channel with compromised security makes the system prone to high levels of vulnerability against both data leakages and adversarial attacks. Another reason associated with vulnerability in FL is a huge client size. As the number of clients increases, the possibility of adversarial nodes entering the system is also high. Thus, it becomes highly difficult to isolate byzantine models during model aggregation, which may lead to poisoning attacks such as model and data poisoning. In centralized FL scenarios, a major reason for system vulnerability may be due to single-node failure instances. As the central server acts as the heart and soul of the system, the failure of such a node can result in the entire network crashing. Although decentralized FL systems are robust to such issues, in reality, the majority of FL applications run on centralized architecture alone, thus making it a genuine concern in FL contexts.

### **9.3.2 ADVERSARIAL ATTACKS**

In any given centralized learning system, the possibility of adversarial attacks is very high. Even though the federated framework ensures a certain degree of robustness against classical attack strategies, adversaries devise newer methodologies to compromise the system. Some of the common security breaching attacks in FL include poisoning attacks and allied attacks following successful poisoning. Even with poisoning attacks, depending on the mode of poisoning, attacks can be divided into data and model poisoning approaches. We will be discussing various types of adversarial attacks in detail in the upcoming section.

## **9.4 PRIVACY ISSUES ASSOCIATED WITH FEDERATED LEARNING**

One of the primary motivations for shifting from centralized learning paradigms to federated ones was the privacy concerns associated with the former. In any organizational context, companies focus on securing and protecting their intellectual property from intruders. However, with conventional learning approaches, most of the privacy requirements cannot be maintained. As centralized machine learning relies on a data pool for model training, the necessity of transferring training data to that pool is mandatory regardless of the privacy level of the data. The data may become susceptible to third-party

interference during data transfer or at the storage device. The extent of such threats can be reduced by employing highly secure communication channels and other security measures at the point of storage, but threats can only be reduced up to a certain limit. As an efficient alternative to these approaches, the federated learning paradigm was introduced.

Even though federated learning works on the assumption that training data is safe and private due to the system architecture, that is not always the case. While each client only shares model updates instead of the full dataset, recent research has shown that adversarial attacks are possible in federated learning. Adversaries have discovered new attack methods where even a small amount of data leakage can be exploited to launch efficient attacks on these systems. Inference attacks and data leakage attacks can be performed based on data leakage, and more powerful attacks like reconstruction attacks can also be successfully executed. In the following section, we will present some well-known adversarial attacks that aim to breach privacy in the federated learning system.

## **9.5 ADVERSARIAL ATTACKS IN FEDERATED LEARNING SYSTEMS**

As discussed earlier, even federated learning is not immune to adversarial attacks and privacy breaches. It can be considered as an alternative to traditional learning approaches for enhanced privacy. FL systems are susceptible to a wide range of attacks at various levels. These attacks include security breaches like poisoning and reconstruction attacks, as well as privacy breaches like data leakage and inferencing attacks, among others. In this section, we will present detailed information about some of the major attack types and their methodologies.

### **9.5.1 POISONING ATTACKS**

Poisoning attacks are one of the most potent attacks compromising system security in FL systems. The general methodology of the attacks is that the adversary injects poisoned data into the system, gradually poisoning the system partially or completely to make it perform tasks according to adversarial intent [8]. Even though the mode of delivery of poisoning may vary, the basic principle remains the same for every case. Such poisoning attacks can be delivered in two different modes, which are discussed in the following subsections.

### 9.5.1.1 DATA POISONING ATTACK

The primary objective of such attacks is to inject malicious data into the system in order to introduce bias in the global model after aggregation. To achieve high levels of secrecy and undetectability, the adversary does not make radical changes to the model parameters. Instead, they employ a gradual approach by making slight modifications to individual parameters over multiple iterations [9]. This approach makes it extremely challenging to detect the poisoning of local model parameters. Once the poisoned parameter is aggregated into the global model, the process is repeated, causing the global model to become biased over several iterations. Such poisoning either degrades the system's performance or causes it to operate with adversarial intent.

### 9.5.1.2 MODEL POISONING ATTACKS

Such attacks employ a more direct methodology for poisoning the system. Rather than poisoning the data sets and eventually attempting to poison the model, such methodologies directly aim to poison the global model [10]. Successful execution of such attacks requires adversarial access at the server level.

## 9.5.2 INFERENCE ATTACKS

In certain instances in Florida, models tend to leak more information than is required. Such information can be related to intended or unintended features about training data or other client details. From an adversarial point of view, such leaked data presents possibilities to launch certain attacks through which various inferences regarding several critical factors can be derived. As FL gradients are generated by individual devices and are dependent on their training data, such gradients will have traces of their data [11]. Thus, effectively leveraging leaked data can easily help deliver powerful inference attacks. Even though several types of inference attacks exist, we will only be presenting two of the most popular modes of inference attacks in this section, namely membership inference attacks and attribute inference attacks.

### 9.5.2.1 MEMBERSHIP INFERENCE ATTACKS

In membership inference attacks, the adversary usually tries to infer the membership of a particular client to determine the presence of the device's

data in model generation. Such a task can be delivered in two modes: active mode and passive mode. During the active mode of attacks, the adversary directly participates in model generation, which gives better access for the adversary to infer information from the model [12]. The prime motto of such attacks is to keep track of the evolution and behavioral patterns of the global model. Such types of attacks can easily exploit classical algorithms such as stochastic gradient descent for extracting training dataset information. In contrast to active attacks, the adversary only attempts to observe the updates made for the global model, taking up a passive role.

#### **9.5.2.2 *ATTRIBUTE INFERENCE ATTACKS***

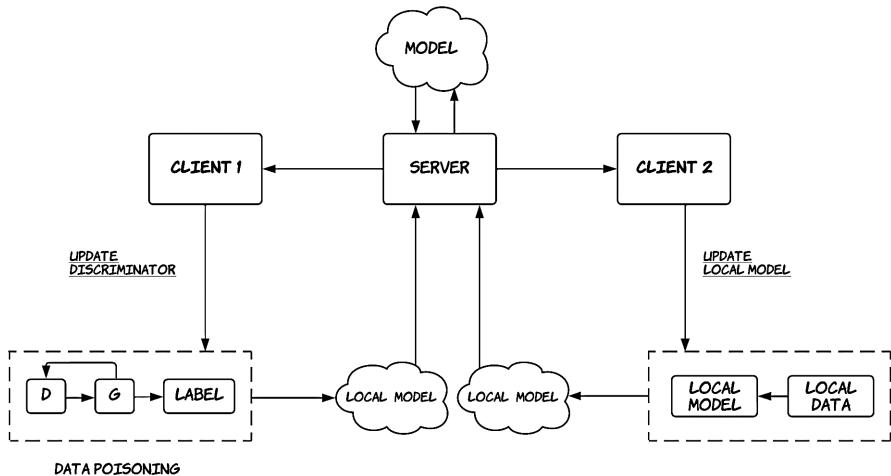
The attribute inference attacks are initiated with the assumption that the system has training data that are correctly labeled under their respective properties. In terms of property inference attacks (PIAs), the mode of delivery can be both active and passive. For passive adversaries, the maximum extent is to observe the model updates, while for active adversaries, techniques such as multi-task learning can be used to deceive the system into learning from another fragment of data that is independent of the attributes, thus extracting information about the attributes from the system. Active adversaries can even go to the extent of leveraging the power to identify instances when a certain attribute appears and disappears during system training [13].

#### **9.5.3 *BACKDOOR ATTACKS***

Bagdasaryan et al. [14] initially proposed the applicability of backdoor attacks in federated learning environments. The authors refer to the approach as a “semantic backdoor” and real-life instances acted as triggers for activating the attack. The goal of such attacks is not to compromise system performance, but to introduce biases that favor certain labels. Article [15] showcases the basic working of backdoor attacks, where the model is biased in an image processing-based task to assign a specific label to images based on the adversary’s intent. The peculiarity of these backdoor attacks is that with as little as 10% of compromised clients, adversaries can successfully launch an attack on the system [16]. Traditional preventive measures, such as label overriding by benign clients or the use of byzantine resilient aggregation approaches, are ineffective against backdoor attacks.

### 9.5.4 GENERATOR ADVERSARIAL NETWORK (GAN)-BASED ATTACKS

Over time, basic adversarial attack methodologies have evolved to surpass existing countermeasures and deliver potent attacks. One such attack methodology is delivered with the aid of GANs. An instance of a GAN-based poisoning attack is presented in the article [15], where the methodology is referred to as Poison GAN. The proposed methodology employs a GAN for generating synthetic yet realistic datasets that can be controlled according to adversarial intent. The GANs in such models are optimized relying on the gradients generated at the client end, thereby attempting to alter the global model. The graphical explanations of such attacks are shown in Figure 9.3.



**FIGURE 9.3** GAN-based poisoning in federated learning.

GAN-based attacks are not limited to just poisoning attacks, but also have applicability in inference-based attacks. These attacks can be delivered in both passive and active manners [17]. Passive inference attacks based on GANs mainly analyze inputs at the server end, while active attacks involve sending global models directly to individual clients. Typically, GAN-based attacks are used for launching reconstruction attacks, where data leakages are utilized to generate data sets or models using GANs. An article [18] presents a benchmark work on data reconstruction from gradient leakage. Several articles also confirm the possibility of efficient data reconstruction from small quantities of leaked gradients.

If properly implemented, GAN-based attacks are extremely difficult to detect and isolate. Successful implementation of these attacks can sabotage the system in terms of model accuracy and performance efficiency. However, GAN-based attacks also have certain disadvantages. These attacks are only possible in federated systems, as the attacker relies on data leakages and does not require access to the entire dataset. In a centralized architecture, where data leakages are comparatively low, the chances of such attacks occurring are reduced.

#### **9.5.5 MODEL RECONSTRUCTION ATTACKS**

Data leakage and attacks following such leakage are major issues that compromise privacy in Federated Learning (FL). Recent research has shown that even with zero knowledge of the training data, but with access to model gradients, it is completely possible to infer the dataset being used to generate those gradients [19]. Certain adversaries, driven by curiosity, always have a tendency to intercept the communication channels of FL clients and servers in an attempt to interpret the data, thus maintaining the possibility of data reconstruction. Unlike other attack methodologies in reconstruction attacks, the likelihood of adversaries directly interacting with the system is very low. They mainly rely on leaked data to perform reconstruction. Although data leakage is a common issue in FL, it is not always easy to efficiently execute such attacks. While they may yield good results on models trained on small datasets with a limited amount of diversity, they struggle to perform well for use cases involving large and non-IID data distribution.

#### **9.5.6 SERVER WITH ADVERSARIAL INTENT**

Even though we estimate that the server in a centralized FL architecture is an honest but curious entity, the possibility of the server turning into an adversary always exists. If the server starts performing with an adversarial intent, it can easily compromise the integrity of the whole system. The server having malicious intent can lead to all sorts of privacy and security breaches, as individual client models can be easily interpreted by the server, through which training data reconstruction is also possible. Even though such an issue can be terminated by employing a fully decentralized FL framework, the majority of the current literature is based on centralized FL architecture, thus adding validity to the concerns.

## 9.6 DEFENSIVE STRATEGIES

As with any other system, maintaining good network and information security by employing traditional methodologies such as encryption is a necessary step in FL as well. As no systems are entirely foolproof, the only way is to improve the protection of the system against such adversarial scenarios so that the extent of impact can be reduced. Generally, defensive strategies can be divided into two types: proactive methods and reactive methods. The strategies employed by the system for detecting and isolating prospective attacks even before the attack is triggered are referred to as proactive methodologies. When it comes to reactive methodologies, the attack will have already started affecting the system, and the techniques employed attempt to repair the system by eradicating the effects of the attack. When it comes to the federated scenario, the attack strategies are different, and so are the protection measures. We will be presenting some of the common defensive strategies employed in FL in the following session.

### 9.6.1 DEFENSIVE STRATEGIES AGAINST POISONING ATTACKS

One of the easiest ways to address poisoning attacks is by implementing efficient methodologies to scan and prevent adversaries from infiltrating the system. Various approaches based on AI techniques can be employed for such screening. An example of such a technique is presented in article [20], which detects any changes in model behavior. Another method involves continuously evaluating the performance of global models during each iteration to easily identify any anomalies. Currently, most of the proposed methodologies are active approaches, with limited literature on passive approaches.

### 9.6.2 DEFENSIVE STRATEGIES AGAINST BACKDOOR ATTACKS

One of the major approaches to reduce the susceptibility of a model to backdoor attacks is to make the model less expressive, thus making it highly difficult to launch a successful backdoor attack. One commonly employed method for this is referred to as pruning, where the total size of the model is drastically reduced in an attempt to reduce the complexity of the model without sacrificing accuracy. In addition to providing robustness against backdoor attacks, pruning also helps in reducing the total number

of parameters, thus lowering communication bandwidth requirements and the probability of message interception by a third party. Shifting the system architecture from a centralized to a decentralized framework can also reduce the vulnerability of the system, as a single point of failure can be eradicated. However, such a solution comes at the cost of reduced model performance [21]. Additionally, this also requires client devices to perform complex computational procedures, which may not be ideal when dealing with edge-based systems and lightweight IoT devices.

### **9.6.3 DEFENSIVE MEASURES FOR PRIVACY PRESERVATION**

To make the system robust against previously mentioned adversarial conditions, certain defensive strategies have been devised that also ensure the benefits of added privacy preservation. Some of the methodologies can be shortlisted as follows:

#### **9.6.3.1 NOISE ADDITION AND COMPRESSION-BASED APPROACHES**

Even though data leakage-based attacks perform efficiently in both federated learning and centralized learning scenarios, Zhu et al. [22] proposed that by adding a certain level of noise to the generated gradients, the efficiency of such inference-generating attacks can be considerably decreased. In the article, the authors demonstrated the drastic drop in accuracy and the inability of the adversary to infer initial data from the obtained gradients post-noise addition. Another widely employed methodology for the same is based on data compression. Also referred to as sparsification, the approach works by reducing the total volume of gradients to be transferred by a considerable margin. Researchers have demonstrated that gradients can be compressed up to very high extents before the accuracy starts getting affected [23]. Similar to compression, partial data transfer-based approaches are also employed, which function on a similar rationale.

#### **9.6.3.2 INCREASING BATCH SIZE AND DATA SIZE**

The DLG algorithm is considered one of the benchmark approaches through which federated learning privacy can be breached via gradient leakage attacks. Most of the recent privacy-preserving mechanisms have been proposed taking

into consideration various aspects of the DLG algorithm. A major peculiarity of the DLG algorithm is that it cannot perform efficiently with increasing batch size and data complexity. Similarly, when working with images of higher dimensions, the DLG algorithm fails to yield satisfactory results. Thus, it can be inferred that with increased system scale, more efforts are required from the adversary to launch an efficient leakage-based inference attack.

#### *9.6.3.3 SECURE MULTIPARTY COMPUTATION (SMC)*

Secure multiparty computation is one of the initial privacy-preserving methodologies proposed along with the proposal of federated learning. The methodology simply attempts to perform secure computation among a set of devices by making use of personal data. One of the major advantages of such an approach is that the privacy aspect can always be maintained as data will be secured through classical cryptographic techniques [24]. In the FL context, a different approach is used for employing SMC where only the transferred gradients are secured rather than securing the whole training data as in other instances. The main disadvantage of such an approach is the computational complexity of encryption, which may intensify with the increasing scale of the system. Thus, it can be said that the system attains security at the cost of efficiency and increased computational expenditure.

#### *9.6.3.4 DIFFERENTIAL PRIVACY (DP)*

Not just in federated learning, but in any given application, differential privacy is a widely employed privacy preservation technique. Differential privacy functions by adding a specific amount of noise to data as a measure of privacy preservation, attempting to distort the data. In the context of federated learning, the noise addition mechanism is mainly employed during the gradient transfer operation from the client to the server [19]. The idea behind this approach is that even if the gradients can be interpreted by an adversary, the distortion in the data makes it useless. Several noise addition mechanisms can be employed in differential privacy, such as Gaussian noise, Laplacian noise, exponential noise, and so on. The amount of noise being added plays a significant role in deciding the level of security offered and also determines the understandability of the model. Increased noise ensures better security with lower accuracy, and vice versa. Thus, an optimal range should be devised depending on the application domain.

#### **9.6.3.5 HIDING ITERATION DETAILS**

In typical federated learning (FL) systems, model iteration details are visible to all parties involved in the training process, including client devices and central servers. This poses a risk as any adversarial device in the network can potentially infer data and launch various types of attacks. To address this, hiding the iteration details allows each individual device to perform training in a secure environment. The concept of a Trusted Execution Environment (TEE) was initially proposed by Xu et al. [26]. TEE ensures that the same procedure can be executed on any machine with a specific trust level, regardless of the administrator's trust level. By employing TEE, overall communication within the network is significantly reduced as part of the security measures. Additionally, TEE provides benefits such as confidentiality, attestation, and integrity to the system. Confidentiality and integrity are maintained as no data related to the execution environment is shared, ensuring secrecy throughout the training process. However, there are certain limitations associated with TEE. It is not suitable for lightweight devices and incurs high computational costs, raising questions about its applicability, especially for small devices.

#### **9.6.3.6 VERIFICATION-BASED APPROACH**

Another possibility for ensuring security in federated learning is through the verification of individual clients and central servers before commencing training. This approach is presented in Verify Net [26], where the author employs a double masking protocol to prevent attackers from inferring training data. Although it is not a robust methodology, it can prevent inference attacks to a certain extent. One major issue associated with this approach is the need for higher rounds of communication with client devices, which in turn decreases the overall performance efficiency of the system.

#### **9.6.4 BLOCKCHAIN-BASED SECURITY ENHANCEMENT**

Even though blockchain and federated learning are two technologies working with entirely different aims, one of the similarities between both is their distributed architecture. Blockchain can be considered a distributed data storage system that has a ledger keeping track of every individual transaction within the network. Due to its distributed architecture and security enhancement

feature, it can be integrated with federated learning to further optimize the system and improve trust among participants. In the context of FL, integrating blockchain can help in consistently maintaining a ledger of various model parameter updates, thus reducing the chances of poisoning attacks. Based on these methodologies, several approaches have been proposed, such as the ones presented in [27, 28].

Apart from the security benefits, blockchain also helps improve privacy in federated learning systems at multiple levels. Blockchain makes the federated environment more transparent for client devices, thus increasing the trust of individual clients. Additionally, unlike classical FL, the shared gradients can be stored in the blockchain instead of being transferred to the central server. Blockchain also brings additional benefits of reliability and auditability of client devices, acting as a deterrent for adversarial participants.

## 9.7 DISCUSSION AND FUTURE RESEARCH DIRECTION

Federated Learning is undergoing extensive research, citing its benefits and continuous evolution. Various sectors have been employing FL for various applications over the last few years. In the context of the industrial sector, with the advent of Industry 4.0, attempts have been made to streamline traditional manufacturing procedures. As a result, federated learning-based systems have been integrated into production units to streamline production. Thus, in any given application, federated learning has proven to be a general solution that efficiently manages distributed architecture while ensuring privacy. Although it overcomes certain shortcomings of the classical machine learning approach, a whole set of other issues needs to be addressed when employing FL-based systems. The scope of this article is limited to the privacy and security aspects of FL only. Some of the other challenges, such as architectural frameworks and system bias, are beyond the scope of this study.

Encountering model bias is a major concern in any machine learning algorithm, including federated learning-based approaches. In federated settings, instances of system bias increase considerably compared to centralized scenarios. While most literature in federated learning (FL) focuses on IID data distribution settings, real-life use cases often involve non-IID data distribution. Bias-related issues are particularly evident in such systems, and the extent of bias varies based on the disparity in data distribution [29]. Although several bias eradication techniques exist for centralized settings, they are not well-suited for federated systems. In

federated learning-based systems, bias mitigation approaches need to be specific to the requirements of individual applications [30]. Model bias in federated learning is closely associated with privacy and security aspects. For example, an exhaustive search for identifying bias in a dataset or model parameters can potentially reveal user data and model information, leading to privacy breaches. Additionally, most poisoning attacks introduce bias into the system to mask the adversary's presence to a certain level. Therefore, addressing these problems presents opportunities for developing a trustworthy FL system with better fairness.

Currently, the research direction in FL is heading towards similarity-based community detection and client clustering approaches [25]. Through such a methodology, generalized models can be developed capable of integrating a larger population with lesser bias. Thus, adversarial presence in such networks can be easily detected as such devices usually vary from abiding standard performance protocols and appear as outliers, thus enhancing system security. Even though the proposed approach ensures better performance, community generation requires inferring certain models and client parameters, which again leads to certain privacy concerns. Also, the rationale for choosing a set of clients to form a cluster should be explained well, thus adding to the challenges.

Another prominent research direction for FL is related to the integration of other technologies. Articles based on the integration of blockchain with FL are being widely presented, as both technologies have found a stable coexistence, optimizing each other's performance. Similarly, other technologies related to IoT and other distributed computing applications can also employ FL for added privacy and introducing cognitive capabilities into lightweight systems.

## **9.8 CONCLUSION**

Even though the last few years have witnessed the explosive growth of federated learning research in both academia and industry, it is still in the development stage as many problems are yet to be identified and solved. Recent research has proposed federated architecture as an effective alternative for overcoming the issues associated with conventional learning approaches. While various factors of FL have been well-studied, the security perspective offers more scope for exploration as adversaries are constantly finding better ways to break down existing defenses and generate improved approaches for breaching systems. This article aims

to present a comprehensive analysis of federated learning from a security and privacy preservation perspective. Additionally, the article provides a detailed overview of various FL terminologies associated with network topology, system architecture, aggregation methods, and more. The current literature on privacy and security in FL systems is also discussed. System vulnerabilities and adversarial attack scenarios were presented in detail, followed by their existing defensive strategies. Finally, we summarize our work by discussing various open challenges associated with the technology and future research directions derived solely from our study. We infer that, although federated learning systems can offer a certain degree of security compared to conventional approaches, they are not to be considered fully secure as they face a wide array of vulnerabilities. In the current scenario, better defensive strategies, including integration with other technologies such as blockchain, need to be considered to advance the technology and optimize the system.

## KEYWORDS

- **backdoor attacks**
- **blockchain**
- **differential privacy**
- **noise addition**
- **poisoning attacks**
- **secure multiparty computation**
- **verification-based approach**

## REFERENCES

1. Yang, Q., Yang, L., Yong, C., Yan, K., Tianjian, C., & Han, Y., (2019). Federated learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 13(3), 1–207.
2. McMahan, B., Eider, M., Daniel, R., Seth, H., & Blaise, A. Y. A., (2017). Communication-efficient learning of deep networks from decentralized data. In: *Artificial Intelligence and Statistics* (pp. 1273–1282). PMLR.
3. Aledhari, M., Rehma, R., Reza, M. P., & Fahad, S., (2020). Federated learning: A survey on enabling technologies, protocols, and applications. *IEEE Access*, 8, 140699–140725.
4. Lalitha, A., Shubhangshu, S., Tara, J., & Farinaz, K., (2018). Fully decentralized federated learning. In: *Third Workshop on Bayesian Deep Learning (NeurIPS)*.

5. Li, X., Kaixuan, H., Wenhao, Y., Shusen, W., & Zhihua, Z., (2019). *On the Convergence of FedAvg on Non-IID Data*. arXiv preprint arXiv:1907.02189.
6. Li, T., Anit, K. S., Manzil, Z., Maziar, S., Ameet, T., & Virginia, S., (2020). Federated optimization in heterogeneous networks. *Proceedings of Machine Learning and Systems*, 2, 429–450.
7. Wang, H., Yurochkin, M., Sun, Y., Papailiopoulos, D., & Khazaeni, Y., (2020). *Federated Learning with Matched Averaging*. arXiv preprint arXiv:2002.06440.
8. Fung, C., Chris, J. M. Y., & Ivan, B., (2018). *Mitigating Sybils in Federated Learning Poisoning*. arXiv preprint arXiv:1808.04866.
9. Tolpegin, V., Stacey, T., Mehmet, E. G., & Ling, L., (2020). Data poisoning attacks against federated learning systems. In: *European Symposium on Research in Computer Security* (pp. 480–501). Springer, Cham.
10. Kairouz, P., McMahan, H. B., Brendan, A., Aurélien, B., Mehdi, B., Arjun, N. B., Kallista, B., et al., (2021). Advances and open problems in federated learning. *Foundations and Trends ®in Machine Learning*, 14(1, 2), 1–210.
11. Nasr, M., Reza, S., & Amir, H., (2019). Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning. In: *2019 IEEE Symposium on Security and Privacy (SP)* (pp. 739–753). IEEE.
12. Zhang, J., Jiale, Z., Junjun, C., & Shui, Y., (2020). Gan enhanced membership inference: A passive local attack in federated learning. In: *ICC 2020–2020 IEEE International Conference on Communications (ICC)* (pp. 1–6). IEEE.
13. Lyu, L., Han, Y., & Qiang, Y., (2020). *Threats to Federated Learning: A Survey*. arXiv preprint arXiv:2003.02133.
14. Bagdasaryan, E., Andreas, V., Yiqing, H., Deborah, E., & Vitaly, S., (2020). How to backdoor federated learning. In: *International Conference on Artificial Intelligence and Statistics* (pp. 2938–2948). PMLR.
15. Zhang, J., Bing, C., Xiang, C., Huynh, T. T. B., & Shui, Y., (2020). Poison GAN: Generative poisoning attacks against federated learning in edge computing systems. *IEEE Internet of Things Journal*, 8(5), 3310–3322.
16. Bhagoji, A. N., Supriyo, C., Prateek, M., & Seraphin, C., (2019). Analyzing federated learning through an adversarial lens. In: *International Conference on Machine Learning* (pp. 634–643). PMLR.
17. Ha, T., & Tran, K. D., (2022). Inference attacks based on GAN in federated learning. *International Journal of Web Information Systems* (Ahead-of-Print).
18. Zhao, B., Konda, R. M., & Hakan, B., (2020). *IDLG: Improved Deep Leakage from Gradients*. arXiv preprint arXiv:2001.02610.
19. Nair, A. K., Ebin, D. R., & Jayakrushna, S., (2023). A robust analysis of adversarial attacks on federated learning environments. *Computer Standards & Interfaces*, 103723.
20. Rodríguez-Barroso, N., Martínez-Cámara, E., Luzón, M., Gerardo, G. S., Miguel, Á. V., & Francisco, H., (2020). *Dynamic Federated Learning Model for Identifying Adversarial Clients*. arXiv preprint arXiv:2007.15030.
21. Wu, C., Xian, Y., Sencun, Z., & Prasenjit, M., (2020). *Mitigating Backdoor Attacks in Federated Learning*. arXiv preprint arXiv:2011.01767.
22. Zhu, L., Zhijian, L., & Song, H., (2019). Deep leakage from gradients. *Advances in Neural Information Processing Systems*, 32.

23. Lin, Y., Song, H., Huizi, M., Yu, W., & William, J. D., (2017). *Deep Gradient Compression: Reducing the Communication Bandwidth for Distributed Training*. arXiv preprint arXiv:1712.01887.
24. Li, Y., Yipeng, Z., Alireza, J., Dongjin, Y., Gaochao, X., & Xi, Z., (2020). Privacy-preserving federated learning framework based on chained secure multiparty computing. *IEEE Internet of Things Journal*, 8(8), 6178–6186.
25. Briggs, C., Zhong, F., & Peter, A., (2020). Federated learning with hierarchical clustering of local updates to improve training on non-IID data. In: *2020 International Joint Conference on Neural Networks (IJCNN)* (pp. 1–9). IEEE.
26. Xu, G., Hongwei, L., Sen, L., Kan, Y., & Xiaodong, L., (2019). Verify Net: Secure and verifiable federated learning. *IEEE Transactions on Information Forensics and Security*, 15, 911–926.
27. Kim, H., Jihong, P., Mehdi, B., & Seong-Lyun, K., (2019). Block chained on-device federated learning. *IEEE Communications Letters*, 24(6), 1279–1283.
28. Kalapaaking, A. P., Ibrahim, K., Mohammad, S. R., Mohammed, A., Xun, Y., & Mahathir, A., (2022). Blockchain-based federated learning with secure aggregation in a trusted execution environment for the internet of things. *IEEE Transactions on Industrial Informatics*.
29. Fang, M., Xiaoyu, C., Jinyuan, J., & Neil, G., (2020). Local model poisoning attacks to {Byzantine-Robust} federated learning. In: *29<sup>th</sup> USENIX Security Symposium (USENIX Security 20)* (pp. 1605–1622).
30. Ferraguig, L., Yasmine, D., Sara, B., & Vania, M., (2021). Survey of bias mitigation in federated learning. In: *Conférence Francophone D'informatique en Parallelisme, Architecture et Système*.



Taylor & Francis

Taylor & Francis Group

<http://taylorandfrancis.com>

## CHAPTER 10

---

# Blockchain Integrated Federated Learning in Edge/Fog/Cloud Systems for IoT-Based Healthcare Applications: A Survey

SHINU M. RAJAGOPAL,<sup>1</sup> M. SUPRIYA,<sup>1</sup> and RAJKUMAR BUYYA<sup>2</sup>

<sup>1</sup>*Department of Computer Science and Engineering, Amrita School of Computing, Amrita Vishwa Vidyapeetham, Bangalore, Karnataka, India*

<sup>2</sup>*CloudsLab, School of Computing and Information Systems, The University of Melbourne, Australia*

---

### ABSTRACT

Modern Internet of Things (IoT) applications generate enormous amounts of data. To prototype the incoming data from such applications, data-driven machine learning has emerged as a viable method that can help develop precise and reliable statistical models. Existing data is not effectively utilized by machine learning as it is stored in data silos, and technology businesses are now required to treat user data carefully in accordance with user-privacy legislation in many regions of the world. It is self-evident that the samples in the typical machine learning centralized server paradigm have different probability distributions of data supplied. As a result, the common model fails to personalize. Without sufficient data, machine learning is unlikely to reach its full potential. Hence, the new distributed paradigm, federated learning, has gained popularity that supports collaborative learning while preserving privacy. IoT applications with cryptography characteristics will be capable of storing and transmitting data securely over networks by maintaining data consistency. Therefore, federated learning and blockchain integration are

---

Federated Learning: Principles, Paradigms, and Applications.

Jayakrushna Sahoo, Mariya Ouassa, & Akarsh K. Nair (Eds.)

© 2025 Apple Academic Press, Inc. Co-published with CRC Press (Taylor & Francis)

particularly beneficial for IoT applications that manage sensitive data such as healthcare. Although several studies have focused on various applications of blockchain technology and federated learning, a thorough examination of these technologies in edge-fog-cloud-based IoT computing systems and healthcare applications has not yet been performed. The basic architecture, structure, types, functions, and characteristics of federated learning and blockchain are addressed at the outset of this survey article. This article also considers the wide range of applications to which federated learning and blockchain have been implemented, including edge, fog, cloud, and IoT computing paradigms. Lastly, it evaluates the different implementations of federated learning in healthcare applications.

## **10.1 INTRODUCTION**

A centralized system is undesirable for both businesses and the environment in the long run, as it can lead to monopolization by a few large companies. This, in turn, can limit the participation of smaller enterprises. The next level of AI is built on the foundation of data privacy, addressing users' concerns about the privacy of various types of data, including personally identifiable information, payment data, protected health information, and confidential data. Two additional challenges associated with cloud/centralized-based techniques are latency and data transfer cost, as data is transmitted through a network to and from the cloud. One potential solution to this is federated learning, a machine learning technique that involves training an algorithm on distributed servers or edge devices that retain local data samples without transmitting them.

By removing the need to store data in the cloud, mobile devices can jointly develop a prediction model using federated learning while retaining all the training data on the device. Federated learning (FL) enables smart models, reduced latency, and reduced battery usage while protecting security and privacy. Training occurs only when the device is powered up, connected to a Wi-Fi network, and idle, ensuring that the phone's performance is unaffected. It works with the concept of training statistical models through distant devices or data centers, such as hospitals or smartphones, while maintaining local data. Defense, telecommunications, pharmaceutics, and IoT are just a few applications where it can be used. Federated learning involves tens to millions of distant devices combined into a single, global statistical model. Smartphones, wearable technologies, and self-driving vehicles are

a few examples of the currently distributed networks that generate massive amounts of data daily. Due to the rising computational capacity of these devices, as well as concerns about transferring confidential data, local data storage and pushing network computing to the edge are becoming more desirable. The term “device,” as used here, refers to nodes, clients, sensors, and organizations that are part of the communication network. Users may be unwilling to disclose their information in order to protect their data or conserve their phone’s limited bandwidth and battery capacity. This model has the limitation that data generated by devices is stored and processed locally, with only intermediate modifications being periodically transferred to a central server. Typically, only a limited subset of devices in federated networks participate in each round of training [25].

Federated learning utilizes an aggregator, which is a centralized server that combines locally trained models and their updates. However, the presence of an aggregator poses significant challenges. It can become a single point of failure, causing the entire training process to collapse if it fails. Unreliable aggregators hinder cooperation among parties. To tackle these issues, several decentralized federated learning architectures have been developed. One powerful decentralized architecture for addressing the security of a central aggregator is blockchain. Blockchain offers advanced features such as tamper-proofing, confidentiality, and traceability, making it highly appealing for enhancing security in areas like IoT. When combined with federated learning technology, blockchain can provide benefits such as low power consumption, low latency, and privacy [39].

A blockchain is a series of blocks that are linked together to store all committed transactions on a public ledger. These blocks contain the entire list of transactional data. The initial block is called the genesis block as it has no parent. Each subsequent block has a relation to the block before it, which is simply a hash value of the previous block called the parent block. The transaction size and block size together determine the maximum number of transactions that can be accommodated in a block [68]. Without the need for a regulatory body, blockchain allows multiple parties to reach an agreement on a specific action and record the agreement. None of the approved and registered activities can be altered without the participation of the other participating entities [5]. In a decentralized and distributed network, a blockchain is a collection of information-storage blocks with digital signatures, forming a multi-field infrastructure technology [30]. This is supported by several fundamental technologies, including distributed consensus methods, cryptographic hashes, and digital signatures. Decentralization, immutability, accountability,

and auditability are all features of blockchain that ensure highly secure and tamper-proof transactions [36]. Due to the public trust it fosters, blockchain enables the elimination of intermediary third parties [7]. Privacy protection, computation, power consumption, network traffic, storage, scalability, and technological abuse are all challenges in blockchain.

The decentralized digital currency, Bitcoin, is a version of blockchain's distributed ledger. It has a block size restricted to one megabyte and a total block formation duration of ten minutes, which limits the network's throughput [52]. Bitcoin's transaction processing speed ranges from 3.3 to 7 transactions per second. Various problems arise, including transaction delays, blockchain congestion, and increased transaction charges. Therefore, relying on the blockchain platform to build a business model may not be a viable option for the government or private sector. The most popular consensus mechanism currently used in blockchain technology is called Proof of Work (PoW). However, one of PoW's major drawbacks is the waste of computational power. Ethereum is working on a hybrid consensus framework that combines PoW and Proof of Stake (PoS) to address this challenge [26]. Numerous studies have focused on applying blockchain in various application areas, but there is no comprehensive study on the use of blockchain technology in edge/fog/cloud/IoT healthcare applications.

The contributions of this study are as follows:

- Overview and fundamentals of federated learning and blockchain concepts.
- I have identified and analyzed the survey papers on federated learning and blockchain.
- I have analyzed research papers on blockchain and federated learning in various application domains, utilizing edge/fog/cloud/IoT technologies. Based on this analysis, I have identified the next directions for blockchain research in the context of federated medical IoT applications.

The rest of this survey paper is organized as follows. Section 10.2 introduces the blockchain and federated learning architecture. Section 10.3 discusses the significance of integrating federated learning and blockchain. Section 10.4 examines survey papers on blockchain and federated learning. Section 10.5 presents the application areas of federated learning and blockchain. Section 10.6 summarizes the blockchain-based federated learning. Section 10.7 deals with blockchain and federated learning in healthcare applications. Section 8 concludes the paper.

## 10.2 PRELIMINARY-FEDERATED LEARNING AND BLOCKCHAIN ARCHITECTURE

### 10.2.1 FEDERATED LEARNING

This section begins with a general overview of federated learning technology, including its structure, types, and how they function.

#### 10.2.1.1 OVERVIEW

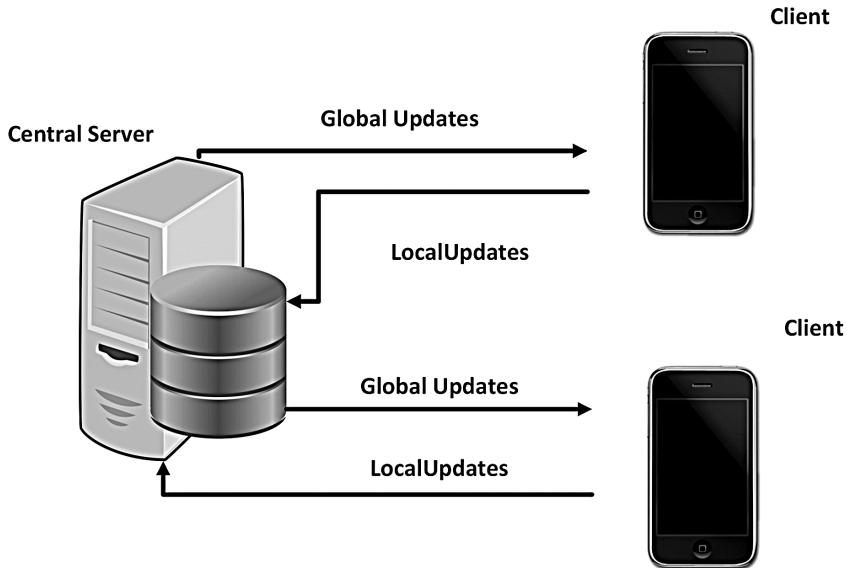
Federated learning facilitates machine learning algorithms to gain knowledge from a diverse collection of datasets stored in distributed locations. It is a decentralized, privacy-preserving strategy that preserves data on devices and uses local machine learning-based training to reduce communication overhead. Federated learning allows collective model training without the need to share training data explicitly with all others involved in the network. It requires an aggregator, which is a central system that combines locally trained models and updates. This allows various parties to train a model together without exchanging data or allowing the aggregator to infer original data, thus lowering the high cost of data gathering. Federated learning is based on an iterative mechanism of client-server interactions. Each iteration of this procedure entails sending the participating nodes the current global model state and training localized models on such nodes to generate a collection of possible model updates, aggregating and processing these local changes, and later processing these local updates [38]. The federated learning system is depicted in Figure 10.1.

#### 10.2.1.2 STRUCTURE

To train the machine learning model, there are three major components: clients, servers, and the communication-computation system.

- **Clients:** Organizations or smart devices that submit local model updates to a central server for aggregation to make practical real-time predictions.
- **Servers:** The aggregator incorporates local updates to create a global model iteratively in order to improve the predictions.
- **Communication-Computation System:** On the client and server, computation for model training and communication for exchanging

model parameters take place. Hence, an efficient communication-computation framework is needed for federated learning.



**FIGURE 10.1** Federated learning system.

#### 10.2.1.3 TYPES

Types of federated learning classes based on training data dispersed throughout the sample and feature spaces are vertical federated learning, horizontal federated learning, and federated transfer learning.

- **Vertical Federated Learning:** To train a global model, it incorporates datasets from different feature spaces.
- **Horizontal Federated Learning:** Across all platforms, it uses data-sets of the same feature space.
- **Federated Transfer Learning:** It involves learning with a pre-trained model to solve a different problem that was trained on a similar dataset.

#### 10.2.1.4 FUNCTIONS

- The server identifies a subset of clients to collaborate with. The usual conditions for device selection are that the device is charged, idle, and

connected to an unmetered network. A selected few participants receive the server's most recent model weights and use them to set up the local machine learning model. The global model is trained and optimized using the local training data of each chosen customer. To compute the update, the client uses the stochastic gradient descent technique.

- To improve model performance updates and minimize communication costs, several gradient descent steps over numerous epochs are processed in one round.
- The clients submit the optimized values to the server once training is over. Due to poor connection, limited computing resources, large amounts of training data, and other factors, some clients may drop out during the training or parameter transmission cycles. If the number of clients who report in time is insufficient, the current active round will be suspended.
- After weighing the updates depending on the scale of the dataset, the server aggregates them and develops a new shared model, which will be improved in consecutive iterations.

#### 10.2.1.5 FEATURES

- **Security and Privacy Benefits:** Personal data is stored locally on one's server, so there is no concern with encryption or privacy.
- **Smarter Realistic Predictions:** Since the data sets are available, generating the model does not require a centralized server. This allows for more realistic predictions to be made on our computer.
- **Lower Latency:** The modified model can be used to make predictions on the user's computer.

#### 10.2.1.6 CHALLENGES

- **Data that is not independent or non-identically distributed:** Each client creates its dataset based on individual actions and system use.
- **Massive Distributed Unbalanced Data:** A large amount of diverse training data is generated by the system's distributed heterogeneous clients.
- **Intermittent Device Connection:** The level of network access varies considerably from one client to the next, which sometimes causes intermittent behavior.

- **Device Memory and Processing Constraints:** The end devices used in the learning process have processing power and memory limitations.
- **Security Threats:** Due to the anonymity of the clients, an intruder may impersonate a regular user and be selected to participate in the learning process.

### **10.2.2 BLOCKCHAIN**

This section starts with a general overview of blockchain technology, including its structure, types of blockchain's, and how they function.

#### *10.2.2.1 OVERVIEW*

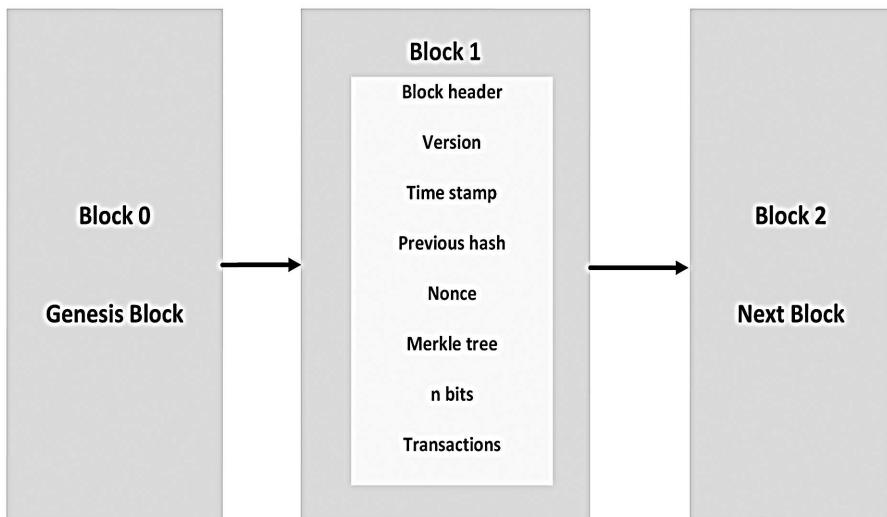
With the publication of the Bitcoin whitepaper in 2008, the blockchain idea became widely known [37]. Blockchain is a chain of connected data blocks, each one dependent on the one before it, creating a continuous data structure. A blockchain is a permanent and verifiable ledger of transactions. A block contains a hash of the previous block which influences the chain. The blockchain's size is decreased by using the Merkle tree structure by representing the transactions as leaf nodes, each of which is separately hashed. Until the root is reached, each group of child hashes is concatenated and hashed once more up the tree [17]. Transactions are broadcast across the network once they are generated. The timestamp confirms that the transactions occurred at the time the block was created. The nonce is a one-time-use number that must be calculated so that the current block's hash value satisfies some arbitrary conditions, such as starting with a predetermined number of zeros to impose the complexity of calculating a block's hash. When blocks are added to the blockchain, the nodes participating in transactions and block creation must be able to confirm their validity. Simplified Payment Verification (SPV) is a payment authentication technology that uses block header messages rather than complete blockchain information to verify payments. This has the potential to significantly reduce user storage in blockchain payment verification, as well as relieve user burden as transaction volumes grow massively [68].

#### *10.2.2.2 STRUCTURE*

The header block includes the following [7]:

- The block version specifies which set of block validation rules should be used.
- A 256-bit hash value, known as the “parent block hash,” is linked to the preceding block.
- The hash value computed for each transaction in the block is known as the “Merkle tree root hash.”
- The timestamp is the current timestamp represented by n bits, which is the current hashing objective in a concise form.
- The 4-byte field known as Nonce begins with 0 and increases with each hash computation.

The block body includes the transaction counter and transactions. The structure of the blockchain is depicted in Figure 10.2.



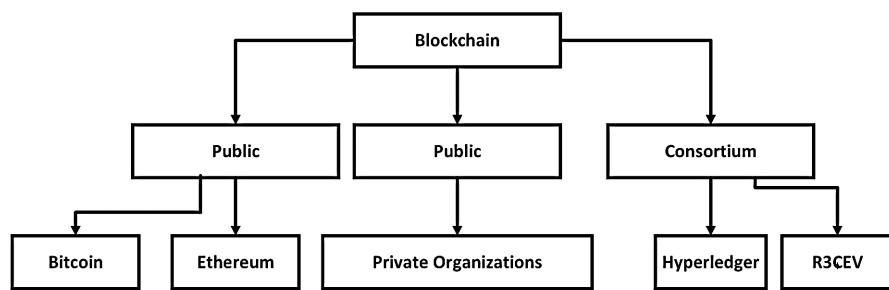
**FIGURE 10.2** Structure of blockchain.

#### 10.2.2.3 TYPES

Blockchain technology can be categorized as follows:

- **Private Blockchain:** This does not allow any node to participate. It has a rigid control management system in place for data access. Only approved nodes are permitted to participate in the private blockchain.

- **Public Blockchain:** All can check and validate the transaction, and they can also participate in the consensus process, as seen in the public blockchain's of Bitcoin and Ethereum.
- **Consortium Blockchains:** The node with authority can be chosen ahead of time. The blockchain's data can be either public or private, and there are partially decentralized consortium blockchain's like Hyperledger and R3CEV. Types of blockchain are described in Figure 10.3.



**FIGURE 10.3** Types of blockchain.

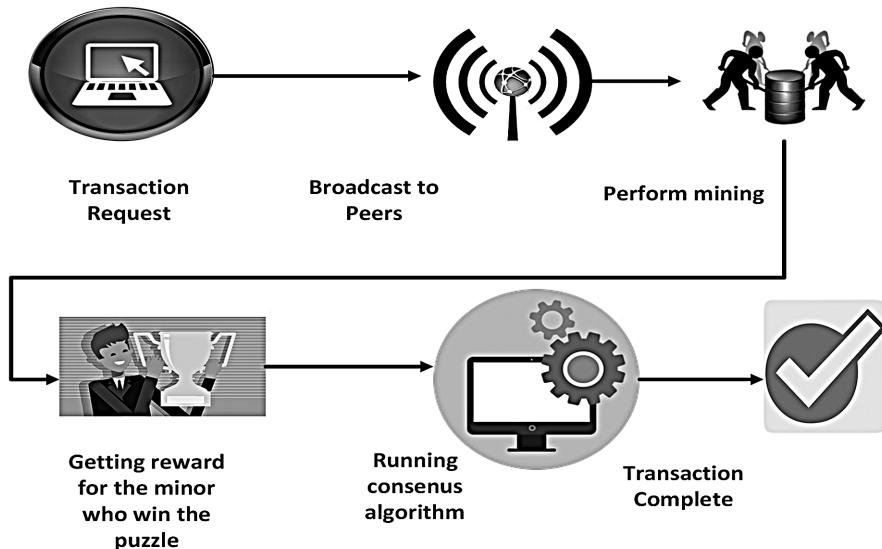
#### 10.2.2.4 FUNCTIONS

The main working processes of blockchain are as follows:

- The transmitting node reports new transactions and sends them to the rest of the network.
- Mining operations will be carried out by all mining nodes, and the winner of the puzzle will update the blockchain with a new block and be eligible to receive the reward.
- After the consensus algorithm has been run, the block will be added to the chain, which every node in the network will accept, allowing the chain to grow. The functioning of the blockchain is presented in Figure 10.4.

#### 10.2.2.5 FEATURES

- **System Transparency:** An application or organization may use blockchain to create a decentralized network that eliminates the need for a central system, thus improving system transparency.



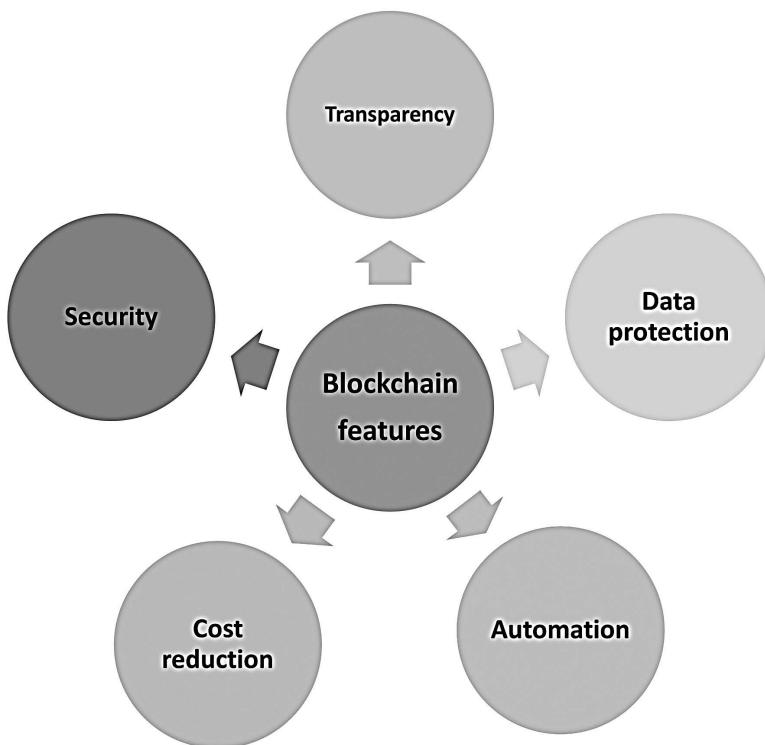
**FIGURE 10.4** Functioning of blockchain.

- **Improved Security:** Other nodes in the network would be aware of any malicious attacker attempting to alter the transaction. Transactions that have been published in the block cannot be modified in any way.
- **Cost Reduction:** There is no need to pay for third-party vendor costs because blockchain is a decentralized framework.
- **Automation:** This is programmable, and when the trigger's conditions are met, it can automatically execute a series of actions such as payments.
- **Data Protection:** In a public ledger, the chain records the entire non-reversible log of transactions, while new transactions are added in an immutable manner and old transactions are kept in the ledger permanently. The description of blockchain features is listed in Figure 10.5.

#### 10.2.2.6 CHALLENGES

- **Scalability:** The capacity of the blockchain platform to accommodate growing transaction loads, as well as the size of the network's nodes, is difficult to achieve.

- **Energy Consumption:** Mining for consensus algorithms would require machines to solve complex equations, which would consume more energy.
- **Memory Requirements:** The blockchain platform's memory capacity to handle growing transactions is challenging.



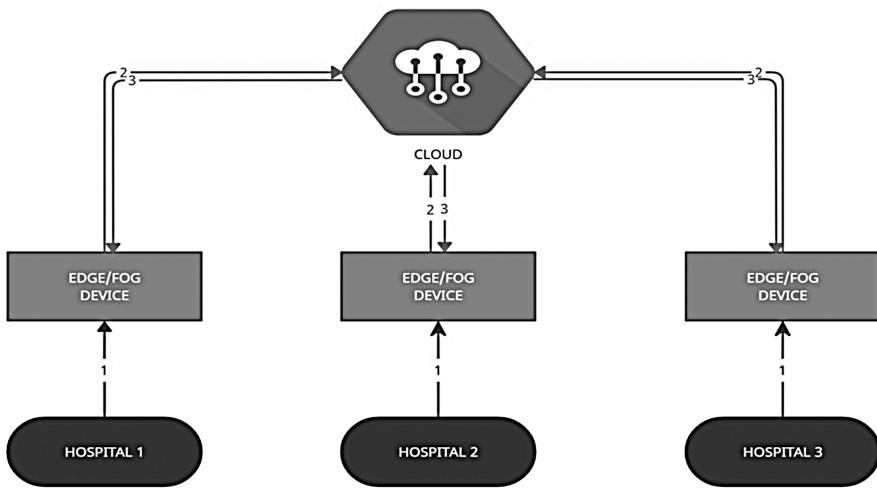
**FIGURE 10.5** Blockchain features.

### 10.3 BENEFITS OF INTEGRATING FEDERATED LEARNING WITH BLOCKCHAIN

To meet the demands of the modern world, IoT can handle huge amounts of data transmission, storage, and real-time processing. AI, fuzzy logic, and machine learning approaches are used to deal with IoT applications. However, these methods require a significant amount of computing power, which is impractical in a completely distributed system. A centralized design is inadequate for the accurate decision-making required by many real-time

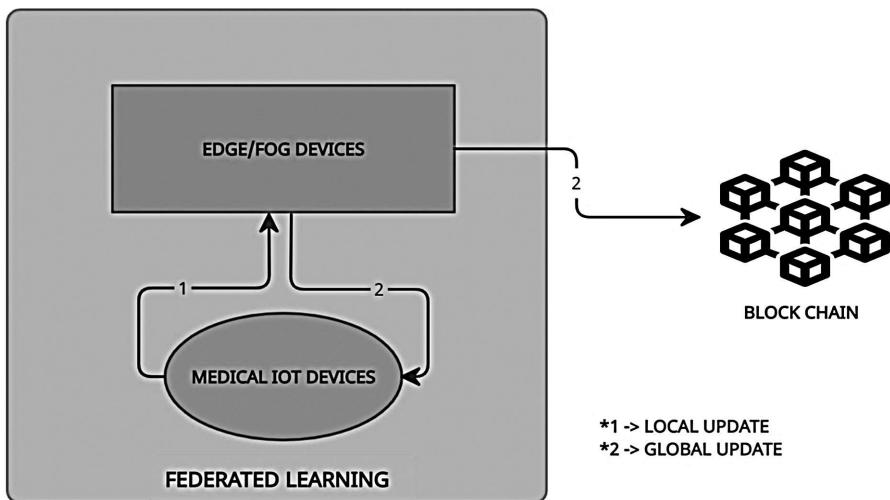
IoT applications within a given time frame. FL's distributed machine learning (DML) technique offers the best solution to address such issues. Mobile devices collect data, which is then used to create local models and train them. The local model data is then provided to the aggregator, which creates a global model by averaging the local models. Each global model is trained on mobile devices until the desired result is achieved.

However, FL models have a few issues related to the security of data. An attacker can access sensitive data on an end device through model updates that are already on the primary server. Additionally, the malicious end device can alter the current local model. Furthermore, attackers can execute an injection attack against the central server to deliver an inaccurate model update to connected end devices during execution. Due to all the factors mentioned above, FL security is crucial. The combination of FL and blockchain can address the security issues with an existing system, which is especially appropriate for applications in the delicate field of healthcare. Compared to centralized networks, the decentralized approach maximizes the utilization of computing resources and offers balanced workloads. Additionally, the blockchain guarantees data consistency by utilizing cryptographic properties. IoT apps with cryptography capabilities will be able to store and transmit data securely over networks.



**FIGURE 10.6** Federated learning for edge computing-based medical application.

Blockchain saves details of medical data, and each patient's identity is assured, ensuring data authentication and authorization. Blockchain performs transaction validation and locally saves trained data of various transactions for medical data, maintaining secure data transfer. The most challenging problem facing crucial infrastructures like the healthcare system is data breaches. Blockchain provides data encryption methods to decrease the risk of data breaches. By providing an FL solution based on blockchain, communication of medical data using encrypted identities without a central server is possible. After adding the transaction to the block, the transaction is equally dispersed among all network nodes. The issues of security and privacy of IoT applications can be solved by integrating FL and blockchain [3, 12, 39]. Figure 10.6 depicts federated learning for edge computing-based medical applications, illustrating the application of federated learning concepts in the medical sector. Figure 10.7 presents blockchain-based federated learning.



**FIGURE 10.7** Blockchain-based federated learning.

#### 10.4 RELATED WORKS

This section elaborates on the existing literature of survey papers on blockchain and federated learning. A thorough overview of the most important federated learning (FL) hardware, software platforms, and protocols helps researchers quickly learn about FL. Aledhari et al. [4] present practical examples of FL applications and use cases to demonstrate how different FL architectures can be

applied in various scenarios, particularly in healthcare and other areas. Liu et al. discuss frequently used federated systems based on functional architecture and explain federated learning systems from four perspectives: parallelism, aggregation techniques, data transfer, and security [31]. Abdulrahman et al. categorize research directions into four categories: system model and design, application areas, privacy and security, and resource management in various fields such as Gboard, smart healthcare, IoT, edge computing, networking, robotics, grid-world, models, recommender systems, cybersecurity, online retailers, wireless communications, and electric vehicles [1]. Zhang et al. present a federated learning research exploration from five perspectives: data partitioning, privacy mechanisms, machine learning models, communication architectures, and systems heterogeneity. To examine the existing practical use of federated learning, it is important to address current difficulties and potential research areas, including communication costs, statistical heterogeneity, system heterogeneity, privacy concerns, and other risks [63] [64]. An analysis of system design issues for large-scale FL implementation considers difficulties with resource allocation, communication costs, privacy, and security concerns [29].

In addition to the present research on federated learning (FL), Rahman et al. survey a new categorization of FL concepts and study topics [1]. The literature also presents a thorough examination of FL's increasing applications in IoT networks, current developments in FL, IoT integration, and FL's potential for supporting IoT services such as data offloading, IoT data sharing, localization, caching, mobile crowd sensing, attack detection, privacy, and security. It is also used in major IoT applications like unmanned aerial vehicles (UAVs), smart transportation, smart healthcare, smart cities, and smart industries [39]. Furthermore, various implementations of FL incentive systems and blockchain-based federated learning (BCFL) industrial application scenarios have been presented to leverage blockchain and improve FL performance [24].

Blockchain technologies are divided into layers such as data, network, consensus, and application, with different blockchain applications being classified according to their domains. A survey conducted a thorough investigation into blockchain consensus strategies, network architecture, and applications [58]. The advantages of adopting blockchain technology in hospital records management, verifiable end-to-end electronic voting, identity management systems, decentralized notary, supply chain management, and access control systems are presented by Maesa et al. [35]. A systematic organizational vision of blockchain networks focusing on the characteristics of blockchain networks, the use of decentralized consensus, an analysis of

distributed consensus system design, and the reward mechanism perspectives have also been discussed [56]. The main objective of Zhou et al. is to categorize and address all existing blockchain scaling solutions. It analyzes various methods and suggests a few possible approaches to solve the blockchain scalability problem [69].

The usage of blockchain as a service to safeguard and manage current information systems is thoroughly examined by Berdik et al. The survey offers comprehensive details on numerous blockchain research and implementations suggested by the researchers, as well as their individual effects on blockchain and its use in other application scenarios [7]. Feng et al. detail how privacy concerns pertain to the blockchain. It examines blockchain's privacy risks and discusses current cryptographic security mechanisms including anonymity and transaction privacy protection. It also summarizes common blockchain implementations and examines potential research issues that must be tackled to maintain privacy as blockchain is being used [15]. By examining well-known blockchain systems like Ethereum, Bitcoin, and Monero, Li et al. investigate common blockchain systems to perform a thorough analysis of the security threats, attacks to the blockchain, vulnerabilities exploited in attacks, and the causes or potential consequences of each risk or vulnerability [25]. The blockchain classification, prominent blockchain consensus algorithms, blockchain implementations, technological problems, current developments in overcoming those challenges, and future perspectives in blockchain technology are presented [68]. Blockchain security concerns are divided into two categories: algorithm-based security and hashing operations-based security. Threats to application blockchain protection have been examined, including network design vulnerabilities and privacy concerns. Potential vulnerabilities in blockchain have been identified, such as cryptographic operation vulnerabilities, identity vulnerabilities, manipulation-based attacks, quantum vulnerabilities, reputation-based attacks, service vulnerabilities, malware attacks, device vulnerabilities, and threats to application blockchain protection [11].

By describing the taxonomy and design of the blockchain, comparing various consensus mechanisms, and analyzing aspects such as scalability, anonymity, interoperability, energy consumption, and regulatory concerns, Monrat et al. present a comparative analysis of blockchain trade-offs [36]. This comparison focuses on node identity management, energy conservation, the adversary's power that can be tolerated, and consensus methods like PoW, PoS, and delegated PoS. Practical Byzantine fault tolerance and Tendermint are considered for domains such as healthcare, voting, governance, identity

management, resources, supply chain, stock exchange, education, digital records, and asset monitoring.

According to Gao et al., the blockchain system is a comprehensive definition of the blockchain architecture, consisting of three layers: network, data, and application. The data layer encompasses data structures and algorithms such as hash and hash pointer, digital signature, and Merkle tree. The network layer includes a distributed agreement consensus mechanism for validating blocks and enables users to update and distribute the blockchain. The application layer demonstrates the utilization of blockchain for leveraging the ledger, achieving consensus among nodes, incorporating cryptographic elements, and implementing smart contracts [17].

## 10.5 FEDERATED LEARNING AND BLOCKCHAIN APPLICATION AREAS

This section discusses the related literature on blockchain and federated learning in edge/fog/cloud/IoT applications. For customized federated learning, Wu et al. propose PerFit, a synergistic cloud-edge architecture that addresses system, statistical, and model heterogeneity in IoT applications. It explores emerging personalized federated learning approaches in a case study of IoT human behavior recognition to demonstrate the efficiency of personalized federated learning for smart IoT applications, which can mitigate the negative effects of heterogeneities in various ways [59]. The most recent developments in federated learning, enabled by IoT applications, consider a variety of factors including sparsification, robustness, quantization, scalability, security, and privacy [71]. The analysis of FL chain applications in well-known MEC fields such as edge data sharing, edge content caching, and edge crowd sensing, including the study of difficulties in FL chain design, communication cost, resource allocation, incentive mechanism, security, and privacy protection, has been discussed [39]. It is suggested to use a neural-structure-aware approach for resource management in federated learning, which is module-based. In this approach, mobile clients are assigned to various sub-networks of the global model based on the status of their local resources. This allows for elastic and effective resource utilization [62]. To make the outcomes of deep reinforcement learning practical and to reduce transmission costs between IoT devices and edge nodes, the proposed work suggests deploying multiple deep reinforcement learning agents on various edge nodes to communicate the decisions

of the IoT devices [48]. To ensure the privacy of updated local models through randomly distributed updates and to eliminate security vulnerabilities through a central curator, the work proposes the incorporation of local differential privacy (LDP) into federated learning. By using weighted aggregation and update verification, it achieves quick convergence.

To address the issue of the lack of control over the posted ledgers, Zhu et al. propose controllable blockchain data management. The proposed approach includes a trust authority node, which allows for the elimination of any suspicious acts. The security and efficiency of the proposed approach have also been analyzed [70]. The integration of blockchain technology with already-in-use cloud technologies, enabling cloud data center reengineering, and the integration of blockchain technology with present cloud systems for increased performance and security have been discussed in Gai et al. [16]. Zhao et al. suggest using hierarchical crowd sensing to evaluate the test accuracy of federated learning in systems using blockchain's with and without differential privacy [72].

Zhang et al. propose the BCPay outsourcing services, a blockchain-based fair payment system in which the service fee is exchanged directly between the customer and the server, eliminating any third party while maintaining payment fairness against intruders and outsourcing service providers. This implementation employs an all-or-nothing checking proof protocol and assesses the efficiency of the proposed BCPay in terms of transactions and computations involved. In terms of transaction size and cost of computation, BCPay is very efficient [65]. ProvChain, a blockchain data provenance architecture that provides a cloud storage application with real-time monitoring of every data access, while improving privacy, availability, transparency, and data accountability as a decentralized and trusted cloud data provenance system, has been proposed. It uses a file as a unit of data, hence, all cloud object operations are reviewed and registered using blockchain to capture and track cloud data access events. The computing complexity overhead increases as the file size grows [28]. A reliable blockchain-based decentralized resource management system uses a smart contract-embedded reinforcement learning approach to reduce the energy used by the request scheduler. It does not require a scheduler, which adds to the cost of energy and reduces the robustness of data centers. Analytical findings on Google traces of cluster and real-world power prices indicate that the proposed solution can dramatically reduce data center costs compared to other standard algorithms [60].

A system that utilizes blockchain technology to ensure data integrity during the offloading process of a multimedia workflow, as well as to manage

cloudlets based on blockchain for multimedia workflow, has been developed using the CloudSim framework. In order to assess the advantages of this proposal, the results are compared to benchmark approaches, namely best fit decreasing, first fit decreasing, and hybrid computation offloading algorithms [61]. The NutBaaS platform provides blockchain services such as network implementation and device management, as well as smart contract analysis and testing, within cloud computing environments. Developers should focus on the code and explore how to adapt blockchain technology to better suit their business needs using these services, rather than worrying about code maintenance [67]. Blockchain technology introduces the first certificate-less public authentication scheme for cloud storage, which provides better security assurance against delaying auditors as presented in the literature. The key concept is that auditors must report each verification result as a transaction in a blockchain, and the verification can always be time-stamped immediately after the transaction is recorded in the blockchain. In comparison to current systems, it has the drawback of continuous communication and computation overhead [66]. Li et al. propose the CK shared network, which is built on private cloud and blockchain technology, and promises to be a reliable platform for mold redesign knowledge-sharing. Confidentiality, integrity, and availability are addressed through symmetric encryption, hash-based connection, and SHA256 encryption algorithms. However, there is no consensus process and smart contract included [27].

The parameters addressed in the above-discussed survey papers have been analyzed and presented in Table 10.1. The table presents a summary of the parameters addressed by the existing literature.

Baniata et al. explore and review published papers that combine blockchain and fog computing, categorizing them based on their type, domain, publication year, blockchain function, consensus algorithm, and level of configuration for IoT, smart mobile devices applications, internet of vehicles applications, e-health applications, and industrial internet of things applications [6]. They discuss the interrelationship between blockchain and edge/fog computing, focusing on blockchain-based solutions. This discussion also covers privacy/security issues in the fog paradigm, as well as security requirements in fog-enabled IoT frameworks using the blockchain [53]. Wu et al.'s proposed work customizes the blockchain for fog node clusters to reduce necessary computing storage space and power consumption. Their paper also proposes a novel approach to recover access control lists (ACLs) in blockchain-based fog node clusters. They suggest the time-aware computing set allocation algorithm as a heuristic method for allocating computing power

**TABLE 10.1** Comparison Analysis of Existing Blockchain Survey Papers

Paper	Applications	Domains	Consensus Algorithm	Network Architecture	Reward Mechanisms	Scaling Solutions	Services	Privacy Issues/ Attacks	Challenges and Opportunities
[10]	✓	✓	✓	✓				✓	✓
[11]	✓								
[12]		✓			✓				
[13]						✓			
[3]							✓		
[14]								✓	
[7]								✓	
[5]	✓			✓					✓
[15]								✓	
[2]	✓			✓					✓
[9]				✓					✓
[25]	✓	✓	✓	✓					
[26]							✓		

to calculate block's hash values, which can reduce the overall processing time in obtaining the block's hash value. Each ACL update is treated as a single blockchain transaction. Each timeslot's ACL hash value is used to synchronize all the ACLs stored in a single fog node cluster based on blockchain at the beginning of the following timeslot [57].

The proposal of the Blockchain and Fog-based Architecture Network (BFAN) aims to reduce the average energy consumption of fog nodes, increase scalability, enable effective communications and computing, and provide enhanced security in smart city applications. In this proposal, data from smart sensors is treated as a blockchain transaction. The performance evaluation was conducted using the iFogSim simulator on a real dataset [50]. To address scalability issues, Kai Lei et al. propose a Group chain, a public blockchain with a two-chain structure that is suitable for fog-based IoT applications. This approach improves protection against attacks such as double spend and selfish mining, while also enhancing transaction throughput and optimizing approval latency [23]. The FogBus architecture, proposed in the literature, enables end-to-end integration of IoT/fog/edge/cloud and provides platform-independent execution of IoT applications. It facilitates app development for developers, allows users to run multiple apps simultaneously, and assists service providers in managing their services. Different FogBus configurations can adapt the computing environment to meet specific needs. The architecture is easily deployable, scalable, cost-efficient, comparatively lightweight, and responsive [73].

A distributed architecture of cloud-based on blockchain and utilizing Software Defined Networking is proposed by Sharma et al. This architecture allows controller fog nodes at the network's edge to fulfill the necessary design principles by providing lower cost, increased reliability, and on-demand access to the compute resources of the IoT network. When compared to conventional IoT architecture, the suggested system enhances the effectiveness of the IoT network by improving its capacity to identify real-time attacks, reducing response times, increasing throughput, and minimizing end-to-end latency [49]. Additionally, a suitable permission blockchain based on the fog system and a novel blockchain-based fog architecture for the industrial IoT have been proposed. The proposed system structure is divided into edge, fog, and cloud. This proposal evaluates the transactions per second, active threads, and the response time of the Hyperledger-based fog architecture for elapsed time [19].

Bouachir et al. propose an industrial cyber-physical system that utilizes both blockchain and edge/fog approaches to address challenges related to protection, quality of service, and data storage, while ensuring compliance with general

data protection regulations (GDPR) for privacy. The decentralized computing resources of fog enable better management of the distributed function of the blockchain, resulting in improved scalability and system availability. However, this approach has drawbacks in terms of energy consumption, regulation, and standards [8]. A lightweight hybrid federated learning architecture, incorporating blockchain, is used to test the reliability of distributed datasets in deep learning (DL) applications developed for COVID patients in clinical trials. Smart contracts handle edge trust, training, and authentication of federated nodes, as well as the credibility of distributed datasets. Each edge federated node performs additive encryption, while the blockchain aggregates the modified model parameters using multiplicative encryption. The raw data from the differentially private Internet of Health Things (IoHT) is stored in the repository for authenticity and shared model training. The position of the training data hash in the repository is stored in the blockchain [46].

## **10.6 BLOCKCHAIN-BASED FEDERATED LEARNING**

Blockchains have the potential to decentralize the coordinating process for model generation in federated learning [41]. Convergence is aided by blockchain-enabled federated learning, which allows for enhanced confirmations and participant selection [12]. FLchain is a blockchain-based federated learning architecture proposed by Nguyen et al. that incorporates design issues and application cases [39]. BlockFL, a blockchain BAFF-based FL architecture, verifies and rewards local model improvements to exchange local model updates [20]. On a public blockchain network, Toyoda et al. propose a mechanism-design-oriented FL protocol in which mechanism design is utilized to create a rule with a specific objective. It has the advantage of rewarding participants who have made significant contributions, which automatically discourages participants from deviating from the protocol [54]. Blockchain-based federated learning without aggregator (BAFFLE), an FL ecosystem based on blockchain technology, is fundamentally decentralized and uses smart contracts to manage round segmentation, aggregation of models, and updates in FL. After segmenting the world's parameter space into discrete parts, it uses a score and bid technique to speed up computation [47].

When compared to non-FL equivalents, Chain FL proposed by Korkmaz et al. uses the private blockchain to transfer the burden of storing the model to the network's nodes, producing promising outcomes with federated learning. To perform the federated learning update phase, miners run the smart contract code that was previously distributed on the network. Because the contract

applies to all miners, the system will continue to function even if one or more miners controlled by various parties fail. The smart contract installed in the private blockchain network allows any party to see both the most recent model and the past [21]. A decentralized paradigm for cognitive computing powered by big data combines federated learning and blockchain to boost industry manufacturing 4.0 performance [45]. Two types of weights are proposed for selecting a subgroup of clients to update the global model using blockchain-based federated learning. One type is based on the accuracy of local learning for each client, and the other is based on each client's participation frequency. To compare the performance of the proposed system with existing schemes, important performance metrics such as learning speed and standard deviation were used [74]. Kim et al. employ a consensus mechanism in blockchain and on-device machine learning without the need for centralized data for training or coordination. This encourages the federation of additional devices with a larger number of training samples by providing rewards proportional to the training sample sizes. The BlockFL end-to-end latency model is developed by analyzing communication, computation, and PoW delays, and the subsequent latency is reduced by adjusting the block production rate to accommodate the additional delay caused by the blockchain network [20].

The incorporation of privacy and security-preserving federated learning into a blockchain-based architecture for secure data sharing between distributed parties has also been discussed. It incorporates federated learning within the permissioned blockchain consensus process so that the computing power required for consensus can also be used for FL training. For privacy-conscious and effective vehicular communication networking, on-vehicle machine learning model updates are communicated and verified in a distributed format utilizing an autonomous blockchain-based federated learning architecture [43]. Aich et al. train the manufacturer's initial model using both the mobile edge server and the smartphone. Manufacturers designate consumers or organizations as second-stage miners to compute the average model utilizing the models obtained from consumers. After the crowd sourcing project, one of the miners chosen as the temporary leader transfers the model to the blockchain to protect individuals' privacy, improve the accuracy of the test, provide differential privacy on the gathered characteristics, and introduce a novel normalization process [3].

An asynchronous federated learning scheme proposed uses the edge data for learning and chooses the participating nodes carefully to reduce overall costs and integrate the learned parameters into the blockchain to increase the reliability of learned models and verify the properties of these parameters through the two-stage process [34]. FL-Block provides

decentralized preservation of privacy while avoiding single points of failure with hybrid identity generation, full verification, access restriction, and data storage in off-chain and retrieval [44]. Ma et al. studied FL's shortcomings and further investigated blockchain-assisted decentralized federated learning (BLADE-FL), a decentralized FL supported by blockchain, and demonstrated how effectively the suggested solution can handle possible problems, particularly the single point of failure issue, present in the conventional FL system [22]. To protect the model parameters of B5G networks edge devices, the proposal of combining blockchain-based FL with Wasserstein generative adversarial network enabled differential privacy. Blockchain makes it possible for decentralized FL to lower communication costs between the edge and cloud while resolving the problem of data falsification. It also offers a method for incentives to resolve the problem of data islands beyond fifth-generation (B5G)-driven edge computing [55].

## **10.7 BLOCK CHAIN AND FEDERATED LEARNING FOR HEALTHCARE**

Certain cognitive diseases can be revealed through everyday behaviors. Wearable devices have made it simple to access people's health records, thanks to rapid advancements in computing technology. However, the ability to learn from health data is constrained due to various technical, ethical, legal, and medical data privacy issues [42]. It is crucial to secure the privacy and confidentiality of healthcare data from outside attackers, as cybercriminals may be interested in obtaining confidential and sensitive information. In some cases, third-party vendors may utilize data analysis to categorize individuals who may be deemed uninsurable based on their medical history or genetic abnormalities. Cybercriminals may even sell stolen data to these vendors for financial gain. To protect patient privacy, hospitals and related organizations are often reluctant to share data in real-world scenarios. Since the introduction of Bitcoin, researchers have been exploring the expansion of blockchain technology beyond financial use cases. Integrating blockchain technology into e-health systems has the potential to enhance service quality [2]. Machine learning models have achieved great success in smart healthcare by training on vast amounts of consumer personal data. However, traditional machine learning models raise privacy concerns as original data is sent for model training. Therefore, incorporating federated learning with enhanced privacy can be beneficial in the context of smart healthcare. In the

field of smart healthcare, blockchain-based federated learning can be highly effective.

The advantages of deploying a blockchain in the healthcare sector approach are as follows:

- Without the use of a trusted mediator, an agreement may be achieved, preventing a performance issue and a single point of failure.
- Patients have the right to preserve their personal information.
- As blockchain data, medical history is accurate, secure, reliable, authentic, and easily distributed.
- Every member of the patient network can monitor changes to the blockchain data, and unauthorized modifications are easily detectable.

Blockchain technology has the potential to be disruptive, necessitating a complete rethink and substantial investment in the ecosystem [14]. Cao et al. propose a secure cloud-assisted eHealth framework using blockchain technology to defend outsourced Electronic Health Records (EHRs) from unauthorized alteration. According to the suggested method, only authenticated participants are permitted to outsource EHRs, and each such activity generates a transaction on a public blockchain [9]. A comparison has been made between a smart remote healthcare system with and without blockchain in terms of power consumption, latency, network use, and load balancing [13]. Islam et al. propose a system for activity tracking and identification based on a multi-class cooperative procedure of categorization, deployable either online or offline, to improve the activity classification's accuracy with the aid of a blockchain architecture based on fog or cloud computing. A method is proposed that uses the frame-based prominent characteristics of videos of various human behaviors, which are then processed using a support vector machine built on the idea of error correction output codes for efficiency and accuracy [18].

Chen et al. propose Fed Health, a system that uses FL to aggregate data and then uses transfer learning to create personalized models. Transfer learning is employed to adjust the model to the user due to the high distribution divergence between the server's data and the user's data. All parameter exchange processes utilize Homomorphic encryption to ensure the protection of user data. The design of a new secure aggregation protocol includes a secure hardware component and an Ethereum-native encryption toolkit [10]. A system proposed in the literature collects a modest amount of data from different hospitals and trains a global model based on blockchain using DL-enabled federated learning [22]. A secure architecture for smart healthcare, supported by blockchain and federated learning, has been proposed to protect privacy

by utilizing blockchain-based IoT cloud platforms [51]. Recent FL designs for smart healthcare include resource-aware FL, secure and privacy-aware FL, incentive FL, and personalized FL. A state-of-the-art review of the new FL applications in vital areas of healthcare, such as COVID-19 detection, medical imaging, and remote health monitoring, has been presented [40].

The principal advantages of federated learning in the healthcare sector approach are as follows:

- Individual hospitals can benefit from the vast datasets of numerous hospitals using federated learning instead of centralizing the data in one location.
- Auto-scaling is enabled by the federated learning healthcare systems at absolutely no additional cost.
- This approach helps to solve the challenge of healthcare data regulation and privacy by collectively training algorithms without disclosing the data.
- It enables precision medicine by allowing models to make balanced decisions that reflect a person's physiology while considering governance and security/privacy concerns.

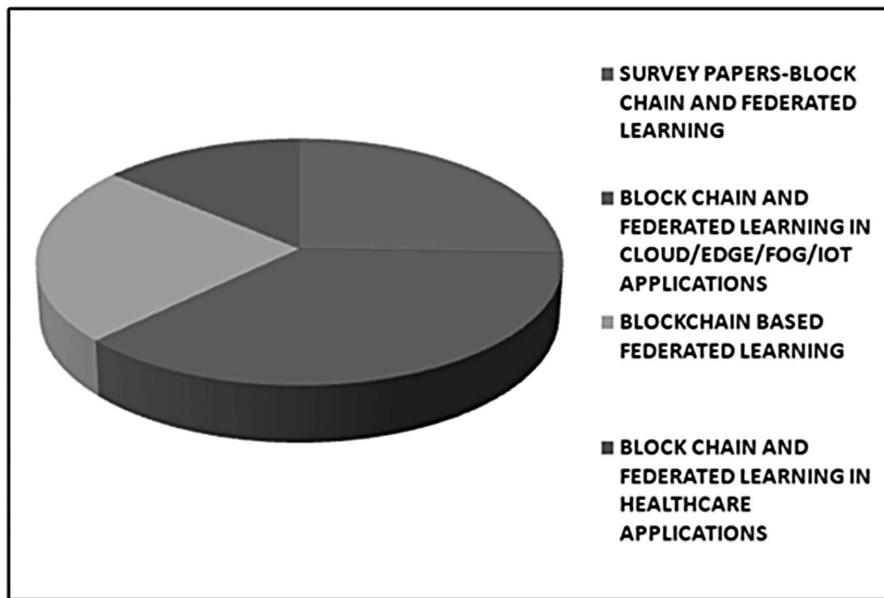
A classification of the articles that were reviewed for the survey is presented in Figure 10.8. Comparison of parameters in survey papers of edge/fog/cloud/IoT healthcare applications analyzed in sections 10.5-10.7 is described in Table 10.2.

## **10.8 CONCLUSION**

This paper explores the significance of federated learning with blockchain integration for IoT healthcare solutions in Edge/Fog/Cloud computing environments. Our survey is divided into different sections: architecture, benefits of federated learning with blockchain integration, blockchain-based federated learning, blockchain and federated learning application areas, blockchain and federated learning in healthcare applications touching upon numerous aspects of the relationship between blockchain/federated learning and edge/fog and IoT systems. The extensive review of the blockchain/FL architecture, as well as its features, is discussed in the first section. In the subsequent sections, the discussion turns to focus more on blockchain/FL in edge/fog/cloud computing in IoT and healthcare applications. For future smart implementations, the research gaps in blockchain-based federated learning have been identified.

**TABLE 10.2** Comparison of Parameters in Edge/Fog/Cloud/IoT Systems

Paper	Applications	Consensus Algorithm	Transaction	Type	Reward Mechanisms	Smart Contracts	Advantages
[27]	Fog computing applications	PoW	Each access control list entry.	Public	No	No	Manage computing power.
[28]	Smart city	Not mentioned	The information obtained by smart sensors.	Public	No	No	Improved security features are ensured while the latency and energy are reduced.
[29]	Fog computing of IoT services.	PoW	Data from IoT nodes	Public	Yes	Yes	Optimization on transaction throughput and confirmation latency.
[30]	Framework for executing and deploying apps for various IoT-enabled systems.	PoW	Data from different IoT-enabled systems.	Not mentioned	No	No	Enables the deployment, management, and monitoring of IoT applications and resources.
[31]	Big data producing IoT applications.	Proof of service	Task completion, data management, and server provision.	Private	Yes	Yes	Enables economical highly effective computing.
[32]	Industrial IoT	Not mentioned	Data from the recently registered edge device is queried and exchanged for resources.	Public	No	Yes	Guarantee fast performance.
[33]	Cyber-physical systems	Not mentioned	An IIoT device requests a transaction, after which the device's metadata are added to a block.	Public or private	No	No	Increased scalability and system availability.
[36]	Healthcare	PoS and PoW	Electronic health records.	Not mentioned	No	No	Guarantees confidentiality, correctness, and integrity of EHRs.



**FIGURE 10.8** A classification of the articles that were reviewed for the survey.

To the best of our knowledge, there is currently no work that offers a detailed and systematic evaluation of blockchain and federated learning use in cloud/edge/fog IoT networks and applications, even though blockchain and federated learning have been extensively discussed in the literature.

## KEYWORDS

- **blockchain**
- **cloud computing**
- **edge computing**
- **federated learning**
- **fog computing**
- **healthcare applications**
- **internet of things**

## REFERENCES

1. Abdul Rahman, S., Tout, H., Ould-Slimane, H., Mourad, A., Talhi, C., & Guizani, M., (2020). A survey on federated learning: The journey from centralized to distributed on-site learning and beyond. *IEEE Internet of Things Journal*, 8(7), 5476–5497.
2. Agbo, C. C., Mahmoud, Q. H., & Eklund, J. M., (2019). Blockchain technology in healthcare: A systematic review. In: *Healthcare* (Vol. 7, No. 2, p. 56). MDPI.
3. Aich, S., Sinai, N. K., Kumar, S., Ali, M., Choi, Y. R., Joo, M. I., & Kim, H. C., (2022). Protecting personal healthcare records using blockchain & federated learning technologies. In: *2022 24<sup>th</sup> International Conference on Advanced Communication Technology (ICACT)* (pp. 109–112).
4. Aledhari, M., Razzak, R., Parizi, R. M., & Saeed, F., (2020). Federated learning: A survey on enabling technologies, protocols, and applications. *IEEE Access*, 8, 140699–140725.
5. Al-Jaroodi, J., & Mohamed, N., (2019). Blockchain in industries: A survey. *IEEE Access*, 7, 36500–36515.
6. Baniata, H., & Kertesz, A., (2020). A survey on blockchain fog integration approaches. *IEEE Access*, 8, 102657–102668.
7. Berdik, D., Otoum, S., Schmidt, N., Porter, D., & Jararweh, Y., (2021). A survey on blockchain for information systems management and security. *Information Processing & Management*, 58(1), 102397.
8. Bouachir, O., Aloqaily, M., Tseng, L., & Boukerche, A., (2020). Blockchain and fog computing for cyber-physical systems: The case of smart industry. *Computer*, 53(9), 36–45.
9. Cao, S., Zhang, G., Liu, P., Zhang, X., & Neri, F., (2019). Cloud-assisted secure e-health systems for tamper-proofing EHR via blockchain. *Information Sciences*, 485, 427–440.
10. Chen, Y., Qin, X., Wang, J., Yu, C., & Gao, W., (2020). Fed Health: A federated transfer learning framework for wearable healthcare. *IEEE Intelligent Systems*, 35(4), 83–93.
11. Dasgupta, D., Shrein, J. M., & Gupta, K. D., (2019). A survey of blockchain from a security perspective. *Journal of Banking and Financial Technology*, 3(1), 1–17.
12. Drungilas, V., Vaičiukynas, E., Jurgelaitis, M., Butkienė, R., & Ceponiene, L., (2021). Towards blockchain-based federated machine learning: Smart contract for model inference. *Applied Sciences*, 11(3), 1010.
13. Ejaz, M., Kumar, T., Kovacevic, I., Yliantila, M., & Harjula, E., (2021). Health-block edge: Blockchain-edge framework for reliable low-latency digital healthcare applications. *Sensors*, 21(7), 2502.
14. Esposito, C., De Santis, A., Tortora, G., Chang, H., & Choo, K. K. R., (2018). Blockchain: A panacea for healthcare cloud-based data security and privacy. *IEEE Cloud Computing*, 5(1), 31–37.
15. Feng, Q., He, D., Zeadally, S., Khan, M. K., & Kumar, N., (2019). A survey on privacy protection in the blockchain system. *Journal of Network and Computer Applications*, 126, 45–58.
16. Gai, K., Guo, J., Zhu, L., & Yu, S., (2020). Blockchain meets cloud computing: A survey. *IEEE Communications Surveys & Tutorials*, 22(3), 2009–2030.
17. Gao, W., Hatcher, W. G., & Yu, W., (2018). A survey of blockchain: Techniques, applications, and challenges. In: *2018 27<sup>th</sup> International Conference on Computer Communication and Networks (ICCCN)* (pp. 1–11).

18. Islam, N., Faheem, Y., Din, I. U., Talha, M., Guizani, M., & Khalil, M., (2019). A blockchain-based fog computing framework for activity recognition as an application to e-healthcare services. *Future Generation Computer Systems*, 100, 569–578.
19. Jang, S. H., Guejung, J., Jeong, J., & Sangmin, B., (2019). Fog computing architecture-based blockchain for industrial IoT. In: *International Conference on Computational Science* (pp. 593–606).
20. Kim, H., Park, J., Bennis, M., & Kim, S. L., (2019). Block chained on-device federated learning. *IEEE Communications Letters*, 24(6), 1279–1283.
21. Korkmaz, C., Kocas, H. E., Uysal, A., Masry, A., Ozkasap, O., & Akgun, B., (2020). Chain FL: Decentralized federated machine learning via blockchain. In: *2020 Second International Conference on Blockchain Computing and Applications (BCCA)* (pp. 140–146).
22. Kumar, R., Khan, A. A., Kumar, J., Golilarz, N. A., Zhang, S., Ting, Y., et al., (2021). Blockchain-federated learning and deep learning models for COVID-19 detection using CT imaging. *IEEE Sensors Journal*, 21(14), 16301–16314.
23. Lei, K., Du, M., Huang, J., & Jin, T., (2020). Group chain: Towards a scalable public blockchain in fog computing of IoT services computing. *IEEE Transactions on Services Computing*, 13(2), 252–262.
24. Li, D., Han, D., Weng, T. H., Zheng, Z., Li, H., Liu, H., & Li, K. C., (2022). Blockchain for federated learning toward secure distributed machine learning systems: A systemic survey. *Soft Computing*, 26(9), 4423–4440.
25. Li, T., Sahu, A. K., Talwalkar, A., & Smith, V., (2019). *Federated Learning: Challenges, Methods, and Future Directions*. arXiv preprint arXiv:1908.07873.
26. Li, X., Jiang, P., Chen, T., Luo, X., & Wen, Q., (2020). A survey on the security of blockchain systems. *Future Generation Computer Systems*, 107, 841–853.
27. Li, Z., Liu, X., Wang, W. M., Vatankhah, B. A., & Huang, G. Q., (2019). CK share: Secured cloud-based knowledge-sharing blockchain for injection mold redesign. *Enterprise Information Systems*, 13(1), 1–33.
28. Liang, X., Shetty, S., Tosh, D., Kamhoua, C., Kwiat, K., & Njilla, L., (2017). ProvChain: A blockchain-based data provenance architecture in a cloud environment with enhanced privacy and availability. In: *2017 17<sup>th</sup> IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing (CCGRID)* (pp. 468–477).
29. Lim, W. Y. B., Luong, N. C., Hoang, D. T., Jiao, Y., Liang, Y. C., Yang, Q., & Miao, C., (2020). Federated learning in mobile edge networks: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, 22(3), 2031–2063.
30. Lin, I. C., & Liao, T. C., (2017). A survey of blockchain security issues and challenges. *Int. J. Netw. Secur.*, 19(5), 653–659.
31. Liu, J., Huang, J., Zhou, Y., Li, X., Ji, S., Xiong, H., & Dou, D., (2022). From distributed machine learning to federated learning: A survey. *Knowledge and Information Systems*, 1–33.
32. Lu, Y., Huang, X., Dai, Y., Maharjan, S., & Zhang, Y., (2019a). Blockchain and federated learning for privacy-preserved data sharing in industrial IoT. *IEEE Transactions on Industrial Informatics*, 16(6), 4177–4186.
33. Lu, Y., Huang, X., Dai, Y., Maharjan, S., & Zhang, Y., (2019b). Differentially private asynchronous federated learning for mobile edge computing in urban informatics. *IEEE Transactions on Industrial Informatics*, 16(3), 2134–2143.
34. Lu, Y., Huang, X., Zhang, K., Maharjan, S., & Zhang, Y., (2020). Blockchain-empowered asynchronous federated learning for secure data sharing in the internet of vehicles. *IEEE Transactions on Vehicular Technology*, 69(4), 4298–4311.

35. Maesa, D. D. F., & Mori, P., (2020). Blockchain 3.0 applications survey. *Journal of Parallel and Distributed Computing*, 138, 99–114.
36. Monrat, A. A., Schel'en, O., & Andersson, K., (2019). A survey of blockchain from the perspectives of applications, challenges, and opportunities. *IEEE Access*, 7, 117134–117151.
37. Nakamoto, S., (2008). Bitcoin: A peer-to-peer electronic cash system. *Decentralized Business Review*, 21260.
38. Ng, D., Lan, X., Yao, M. M. S., Chan, W. P., & Feng, M., (2021). Federated learning: A collaborative effort to achieve better medical imaging models for individual sites that have small labeled datasets. *Quantitative Imaging in Medicine and Surgery*, 11(2), 852.
39. Nguyen, D. C., Ding, M., Pathirana, P. N., Seneviratne, A., Li, J., & Poor, H. V., (2021). Federated learning for Internet of things: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, 23(3), 1622–1658.
40. Nguyen, D. C., Pham, Q. V., Pathirana, P. N., Ding, M., Seneviratne, A., Lin, Z., & Hwang, W. J., (2022). Federated learning for smart healthcare: A survey. *ACM Computing Surveys (CSUR)*, 55(3), 1–37.
41. Passerat-Palmbach, J., Farnan, T., McCoy, M., Harris, J. D., Manion, S. T., Flannery, H. L., & Gleim, B., (2020). Blockchain-orchestrated machine learning for privacy-preserving federated learning in electronic health data. In: *2020 IEEE International Conference on Blockchain (Blockchain)* (pp. 550–555).
42. Passerat-Palmbach, J., Farnan, T., Miller, R., Gross, M. S., Flannery, H. L., & Gleim, B., (2019). *A Blockchain Orchestrated Federated Learning Architecture for Healthcare Consortia*. arXiv preprint arXiv:1910.12603.
43. Pokhrel, S. R., & Choi, J., (2020). Federated learning with blockchain for autonomous vehicles: Analysis and design challenges. *IEEE Transactions on Communications*, 68(8), 4734–4746.
44. Qu, Y., Gao, L., Luan, T. H., Xiang, Y., Yu, S., Li, B., & Zheng, G., (2020). Decentralized privacy using blockchain-enabled federated learning in fog computing. *IEEE Internet of Things Journal*, 7(6), 5171–5183.
45. Qu, Y., Pokhrel, S. R., Garg, S., Gao, L., & Xiang, Y., (2020). A blockchain federated learning framework for cognitive computing in industry 4.0 networks. *IEEE Transactions on Industrial Informatics*, 17(4), 2964–2973.
46. Rahman, M. A., Hossain, M. S., Islam, M. S., Alrajeh, N. A., & Muhammad, G., (2020). Secure and provenance enhanced Internet of health things framework: A blockchain managed federated learning approach. *IEEE Access*, 8, 205071–205087.
47. Ramanan, P., & Nakayama, K., (2020). Baffle: Blockchain-based aggregator free federated learning. In: *2020 IEEE International Conference on Blockchain (Blockchain)* (pp. 72–81).
48. Ren, J., Wang, H., Hou, T., Zheng, S., & Tang, C., (2019). Federated learning-based computation offloading optimization in edge computing-supported internet of things. *IEEE Access*, 7, 69194–69201.
49. Sharma, P. K., Chen, M. Y., & Park, J. H., (2017). A software-defined fog node-based distributed blockchain cloud architecture for IoT. *IEEE Access*, 6, 115–124.
50. Singh, P., Nayyar, A., Kaur, A., & Ghosh, U., (2020). Blockchain and fog-based architecture for the internet of everything in smart cities. *Future Internet*, 12(4), 61.
51. Singh, S., Rathore, S., Alfarraj, O., Tolba, A., & Yoon, B., (2022). A framework for privacy-preservation of IoT healthcare data using federated learning and blockchain technology. *Future Generation Computer Systems*, 129, 380–388.

52. Tang, J., Huang, C., Liu, H., & Al-Nabhan, N., (2020). Cloud storage strategy of blockchain-based on genetic prediction dynamic files. *Electronics*, 9(3), 398.
53. Tariq, N., Asim, M., Al-Obeidat, F., Zubair, F. M., Baker, T., Hammoudeh, M., & Ghafir, I., (2019). The security of big data in fog-enabled IoT applications including blockchain: A survey. *Sensors*, 19(8), 1788.
54. Toyoda, K., Zhao, J., Zhang, A. N. S., & Mathiopoulos, P. T., (2020). Blockchain-enabled federated learning with mechanism design. *IEEE Access*, 8, 219744–219756.
55. Wan, Y., Qu, Y., Gao, L., & Xiang, Y., (2022). Privacy-preserving blockchain-enabled federated learning for b5g-driven edge computing. *Computer Networks*, 204, 108671.
56. Wang, W., Hoang, D. T., Hu, P., Xiong, Z., Niyato, D., Wang, P., & Kim, D. I., (2019). A survey on consensus mechanisms and mining strategy management in blockchain networks. *IEEE Access*, 7, 22328–22370.
57. Wu, D., & Ansari, N., (2020). A cooperative computing strategy for blockchain-secured fog computing. *IEEE Internet of Things Journal*, 7(7), 6603–6609.
58. Wu, M., Wang, K., Cai, X., Guo, S., Guo, M., & Rong, C., (2019). A comprehensive survey of blockchain: From theory to IoT applications and beyond. *IEEE Internet of Things Journal*, 6(5), 8114–8154.
59. Wu, Q., He, K., & Chen, X., (2020). Personalized federated learning for intelligent IoT applications: A cloud-edge based framework. *IEEE Open Journal of the Computer Society*, 1, 35–44.
60. Xu, C., Wang, K., & Guo, M., (2017). Intelligent resource management in blockchain-based cloud data centers. *IEEE Cloud Computing*, 4(6), 50–59.
61. Xu, X., Chen, Y., Yuan, Y., Huang, T., Zhang, X., & Qi, L., (2020). Blockchain-based cloudlet management for multimedia workflow in mobile cloud computing. *Multimedia Tools and Applications*, 79(15), 9819–9844.
62. Yu, R., & Li, P., (2021). Toward resource-efficient federated learning in mobile edge computing. *IEEE Network*, 35(1), 148–155.
63. Zhang, C., Xie, Y., Bai, H., Yu, B., Li, W., & Gao, Y., (2021). A survey on federated learning. *Knowledge-Based Systems*, 216, 106775.
64. Zhang, K., Song, X., Zhang, C., & Yu, S., (2022). Challenges and future directions of secure federated learning: A survey. *Frontiers of Computer Science*, 16(5), 1–8.
65. Zhang, Y., Deng, R. H., Liu, X., & Zheng, D., (2018). Blockchain-based efficient and robust fair payment for outsourcing services in cloud computing. *Information Sciences*, 462, 262–277.
66. Zhang, Y., Xu, C., Lin, X., & Shen, X., (2019). Blockchain-based public integrity verification for cloud storage against procrastinating auditors. *IEEE Transactions on Cloud Computing*, 9(3), 923–937.
67. Zheng, W., Zheng, Z., Chen, X., Dai, K., Li, P., & Chen, R., (2019). Nut BaaS: A blockchain-as-a-service platform. *IEEE Access*, 7, 134422–134433.
68. Zheng, Z., Xie, S., Dai, H. N., Chen, X., & Wang, H., (2018). Blockchain challenges and opportunities: A survey. *International Journal of Web and Grid Services*, 14(4), 352–375.
69. Zhou, Q., Huang, H., Zheng, Z., & Bian, J., (2020). Solutions to the scalability of blockchain: A survey. *IEEE Access*, 8, 16440–16455.
70. Zhu, L., Wu, Y., Gai, K., & Choo, K. K. R., (2019). Controllable and trustworthy blockchain-based cloud data management. *Future Generation Computer Systems*, 91, 527–535.

71. Khan, L. U., Saad, W., Han, Z., Hossain, E., & Hong, C. S., (2021). Federated learning for the Internet of things: Recent advances, taxonomy, and open challenges. *IEEE Communications Surveys & Tutorials*.
72. Zhao, Y., Zhao, J., Jiang, L., Tan, R., Niyato, D., Li, Z., & Liu, Y., (2020). Privacy-preserving blockchain-based federated learning for IoT devices. *IEEE Internet of Things Journal*, 8(3), 1817–1829.
73. Tuli, S., Mahmud, R., Tuli, S., & Buyya, R., (2019). FogBus: A blockchain-based lightweight framework for edge and fog computing. *Journal of Systems and Software*, 154, 22–36.
74. Kim, Y. J., & Hong, C. S., (2019). Blockchain-based node-aware dynamic weighting methods for improving federated learning performance. In: *2019 20<sup>th</sup> Asia Pacific Network Operations and Management Symposium (APNOMS)* (pp. 1–4).



Taylor & Francis

Taylor & Francis Group

<http://taylorandfrancis.com>

## CHAPTER 11

---

# Incentive Mechanism for Federated Learning

LEKHA C. WARRIER,<sup>1</sup> G. K. RAGESH,<sup>2</sup> and PAO-ANN HSIUNG<sup>3</sup>

<sup>1</sup>*Department of Computer Science, Indian Institute of Information Technology, Kottayam, Kerala, India*

<sup>2</sup>*Department of Cyber Security, Indian Institute of Information Technology, Kottayam, Kerala, India*

<sup>3</sup>*Department of Computer Science and Information Engineering, National Chung Cheng University, Taiwan*

---

## ABSTRACT

The exponential growth of data necessitates the development of novel technologies such as artificial intelligence (AI), machine learning, and data mining. The data analytics industry has expanded, and machine learning has found applications in everyday life. As data becomes more dispersed, researchers must focus on the scalability of machine learning algorithms. The data from local machines was brought into the centralized machine learning paradigm, which trained the models. Because the data must be transferred to the central server for training, this violates the clients' privacy and security. It resulted in a new distributed machine learning (DML) method in which the local models were trained on the device where the data was generated. This approach necessitated extensive computational and communication resources, which the clients must bear. Because clients have a significant impact on the overall performance of the global model, mechanisms that encourage their participation in the long run with quality data are critical. Clients should receive incentives that are proportional to their contribution

to the global model's training. A well-designed incentive mechanism can thus result in a highly accurate global machine-learning model. Different approaches, such as game-theoretical-based, contract-theory-based, auction-based, deep reinforcement-based, or blockchain-based methods, can be used to design appropriate incentive mechanisms. This chapter delves into the various types of incentive mechanism designs and their applications in cross-device and cross-silo federated environments.

## **11.1 INTRODUCTION**

As the amount of data produced by all digital devices grows exponentially, technologies that can make use of massive amounts of data are in high demand. Intelligent systems learn from experience, efficiently train the model, and predict the behavior of previously unseen instances. Machine learning algorithms are rapidly evolving but with only one actor. In traditional machine learning algorithms, clients must share their private data with a central server, and the model is trained at the central server. This resulted in a breach of privacy and security. It also reduces the server's ability to learn in real-time. The server may sell the user's private data or be intercepted, making it vulnerable to various types of attacks. According to GDPR rules, intelligent systems must be designed in such a way that the privacy of user data is preserved [28]. Federated learning ensures that models are trained at the edge and only the parameters of all local models are transferred to the central server. The central server is accountable for securely aggregating all local models and creating the global model. The global model is then fetched by all of the clients. This procedure is repeated until the model arrives at the desired result. The main advantage of federated learning is that user data is never sent away from the devices [3]; instead, the trained model parameters are sent back to the server. The quality and quantity of data provided for model building are also important considerations. High-quality data can be used to build precise systems. Edge devices that participate in the learning process should have ample computational and communication resources [2]. Federated learning is also known as cooperative learning (collaborative learning) because the training is a collaborative effort by all clients who participate in the training process. The federated adds privacy preservation to the joint learning process. We have used all these terms in this chapter for explanation purposes.

The machine learning model will become more efficient after being trained with enough data. In our data-driven world, data is distributed. When

a substantial amount of information is accumulated and used for model development, privacy concerns will arise. To avoid the use of user-sensitive data, on-device training in the form of federated learning is encouraged. The machine learning task is created, and as a result, a global model with hyperparameters is initialized. Clients interested in participating in the federated learning process are given access to the global model. The initial model is given to the devices, who are then instructed to train it using their own data and computational resources [1]. Before being returned to the central server for aggregation, the updated models are encrypted. The server collects complete client updates and securely aggregates global models. The updated global model is then distributed to the devices, and the process is repeated until the model is accurate enough.

As artificially intelligent systems become more popular, on-device training of machine learning models is becoming extremely prevalent. Making the machine learn on its own and only transferring model updates to the central server protects the privacy of the systems participating in collaborative learning. Edge computing is important in data science because complex computations will take place at edge devices. In a centralized machine learning paradigm, the devices must share sensitive data with the central server, which performs all training and distributes updates to the devices. The central server should have plenty of resources to collect all of the client's data and train the model with it. In this scenario, the central server serves as a single army and is vulnerable to poison attacks. Privacy concerns arise when the client's private data must be shared with the central server for the model updates to be built.

Clients in the federated learning paradigm are typically diverse. In a cross-device setting, millions of clients can participate in the learning task. Mobile phones, laptop computers, and tablets with varying specifications can be used. Due to the varying speeds of the various devices, the server may have to wait for the aggregation process in this case. Dealing with a diverse and large number of clients in the learning task is one of the major challenges that federated learning researchers face. The clients who take part in the federated learning task have limited resources. The unreliability of federated learning clients is another issue. Client participation is critical to the task's success. The proper selection of client devices for a specific task, as well as motivating them for long-term involvement in the federated learning task, is a critical idea for improving the global model's performance [4]. Several end devices may be rich in data quantity but inadequate in computing capability. Client dropouts during the learning period have a significant impact on the model's performance. The socioeconomic behavior of the device owners

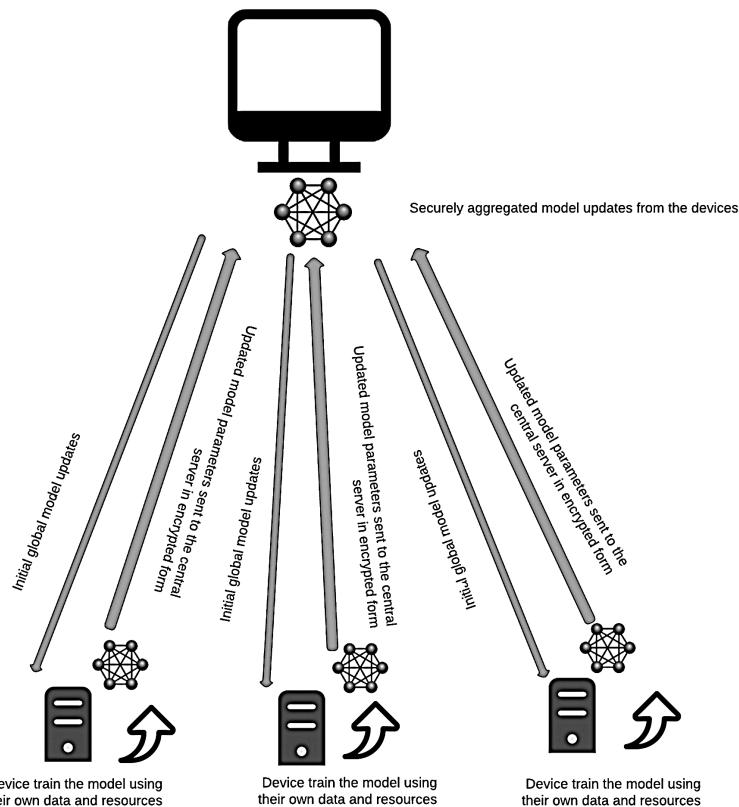
may affect the processing capability of the central server. Robustness should be added to the federated learning scenarios to overcome the aforementioned issues. To address the aforementioned issues, federated learning, a privacy-preserving DML framework, was initiated. Clients are the federated network's backbone. Quality data distribution and long-term device participation result in highly accurate models. A suitable incentive mechanism must be devised to encourage clients to actively engage in the collaborative process of learning [6]. A fair distribution of incentives is required for long-term federated learning. On-device training incurs communication and computation costs. Fair rewards should be provided to participants to compensate for expenses and to inspire clients to participate actively in the cooperation training.

## 11.2 FEDERATED LEARNING ARCHITECTURE

Mobile devices carry sufficient data that, when properly trained, can contribute to high-performing machine learning models. Federated Learning makes use of the edge computing concept, which encourages training at edge devices. Federated learning applications can take advantage of the performance of local machine learning models, and the use of cutting-edge technologies in federated learning can broaden its applicability. The data locality feature has also become an important feature in the DML paradigm [13]. The energy efficiency of machine learning techniques can be increased by training data locally, where it was generated. Google introduced federated learning to prioritize user privacy while keeping training data at the edge devices. The architecture of federated learning consists of the following phases. The architecture of federated learning is shown in Figure 11.1.

1. **Task Publication:** According to the requirements, the central server must publish the task on which the ML/DL model will be built. It is possible to use any supervised (classification) or unsupervised learning task (clustering) for the model building.
2. **Client Selection:** The central server selects clients from a pool of available devices, including idle devices, plugged-in devices, and Wi-Fi enabled devices.
3. **Sharing the Global Model with the Participating Devices:** The hyperparameters of the model have been tuned, and the global model is constructed by the central authority's task. All selected clients utilize the initial global model.

4. **Local Device Training:** Clients download the initial model and train it on their devices using their own resources and data. After several iterations, the model is trained, and the local update is sent to the central server in the form of model parameters. The model parameters are encrypted before being sent.
5. **Secure Aggregation of Local Model Updates:** The central server collects all model parameters and securely aggregates the model for the next training round. The problem of determining a multi-party weighted sum in which no participant discloses its update to any other participant is referred to as secure aggregation.
6. **Global Model Transfer:** The revised model is transmitted to all selected clients, and the subsequent round process begins with this phase. Clients download the most recent model, and the preceding steps are repeated.

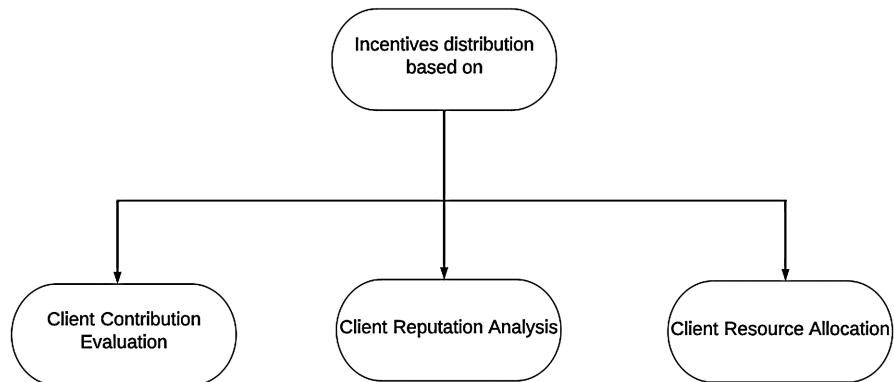


**FIGURE 11.1** Federated learning architecture.

### 11.3 INCENTIVE SCHEME FOR FEDERATED LEARNING

#### 11.3.1 NEED FOR AN INCENTIVE MECHANISM IN A FEDERATED LEARNING SCENARIO

The federated learning paradigm heavily relies on edge devices. Devices must contribute positively to the group learning process to enhance system reliability [5, 29]. The accuracy of the final model created by federated training is highly reliant on the response of each client device. The server initiates a machine learning task, creating a global model with hyperparameters and distributing it to all available clients. The client devices properly train the model using their data and assets. Because they use their own data and computation power, the accuracy of the local model is heavily influenced by their input. Some interested clients may show agnosticism toward model training by refusing to go through the additional training process.



**FIGURE 11.2** Criteria for designing incentives.

It encourages participants in federated learning to actively participate in the long run. The goal of an optimal incentive design is to reward clients based on their contribution to the learning task. If a client provides accurate and high-quality data, a positive reward should be given. Conversely, the reward should be zero or negative if the client is malicious or engages in adversarial behavior [16]. Therefore, an effective incentive mechanism must assess client contributions and the accuracy of the data they provide. In real-world applications, federated clients can be categorized into two types. The

first category includes systems that genuinely prefer participation and can play a role in the development of the model. The second category consists of malicious clients who attempt to impersonate the server as clients, pretending to be extremely valuable contributors. A fair incentive mechanism will undoubtedly empower clients to be fully involved in the federated learning environment, especially those who contribute high-quality data [9]. Due to the lack of access to each client's local data, it is challenging to determine the quality of individual data and design appropriate incentives for participants.

The federated learning incentive scheme encourages data owners to generate a sufficient amount of high-quality data for local training from the start, and as a result, the global model's accuracy improves. Offering incentives to participants encourages users to engage in desirable behavior. By offering incentives, clients are urged to join cooperative learning with the best quality input [8]. A reasonable incentive scheme enables clients to partake in the productive model-building process while also continuing to improve model accuracy. People who participated lacked the motivation to update data in real-time and contribute actively to the federation in the absence of an authentic incentive mechanism. As a result, the model's accuracy will suffer immensely, and the model will be incapable of proficiently predicting unknown data. It will be a waste of resources and time for everyone associated. The final trained model is swayed by each participant's local model. Clients of the federation are constantly striving to enhance the final model's quality.

As a result of the following factors, a credible incentive architecture scheme has been devised. To incentivize attendees to participate in more federated learning, clients must first be reimbursed for the use of their resources [11]. Secondly, in this learning paradigm, the server has no jurisdiction over the clients, and the server is uninformed of the information that the clients are gathering and transferring to the local model development [3]. The issue of client dropout must also be addressed for federated learning settings to be sustainable. An efficient strategy for incentive mechanism design must be chosen to empower participants to be active in the entire learning process [6]. To assign the appropriate reward, each user's honest user participation depth must be identified. The client's data contribution, resource allocation, or client reputation can all be used to determine reward allocation.

Participants typically earn money when they engage in federated learning with their local datasets. Therefore, evaluating the contributions of various

data sources is crucial for ensuring that the profits generated by the learning program are divided up appropriately. To ensure the privacy of the participating clients in a federated learning scenario, model updates rather than raw data are sent to the central server. Since the parameters are in the form of mapped features, local updates do not directly reveal the information when the model is built using deep learning (DL) models.

The Shapley value method of cooperative game theory was used to assess the quality of the data provided by the clients. It is a metric used to evaluate the marginal contribution of each client to the shared model training framework. Data quantity is another criterion for evaluating input for local model training. The model that trains with more data is more likely to generalize previously unseen data. Several game-theoretical, as well as deep reinforcement learning approaches, have been widely adopted for assessing the data quantity of contributing clients.

The prestige of the clients can be considered when selecting clients for the federated learning ecosystem. The reputation is updated based on the performance of client devices on the specific task. The system was launched with a uniform reward rate, and when model updates are genuine, the reputation is positively updated. However, if clients contribute malicious updates, the reputation is negatively updated. Figure 11.2 illustrates the criteria used to design incentive schemes. In conjunction with reverse auction mechanisms, reputation-based contribution evaluation has been considered in the horizontal federated learning setup.

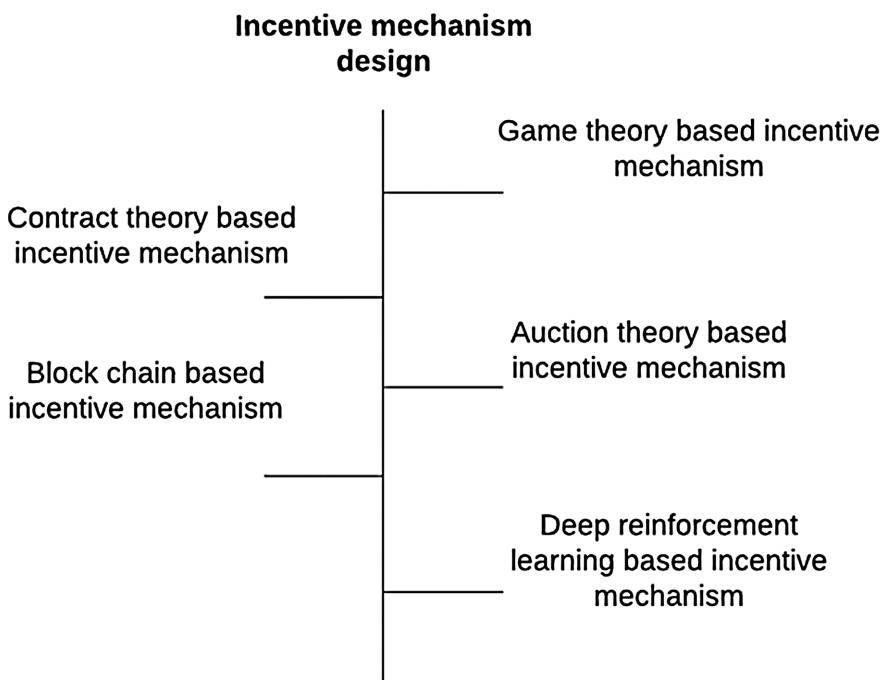
The central server and various clients are the main components of a federated learning setup. Due to the decentralized training scenario, learning is subject to various resource constraints such as energy, bandwidth, and others. The process involves computational and communication resources. Local model training on individual devices is an iterative process, and the heterogeneity of different client devices may cause delays in local model training. In reference [6], a Stackelberg game for CPU power allocation and budget allocation has been developed. A deep reinforcement learning approach is used to evaluate and assign incentives in a dynamic model environment [6]. A two-stage Stackelberg game perspective is employed to achieve communication efficiency in model parameter exchange within a federated learning framework. A crowd-sourcing framework was utilized for the successful execution of this approach [27].

Machine learning tasks are widely distributed among the participants in the federated learning (FL) environment. The model is trained using rich data sources and computational resources, which improves the efficiency

of the ecosystem through task distribution. However, client trustworthiness has become a challenging issue due to factors such as data availability, network resources, and computational power of the participating entities. It is difficult to determine the dependability of the participating clients during the collaborative learning process. To address this, the creation of an effective incentive mechanism is crucial to encourage users to contribute useful data. The accuracy of the global model is strongly influenced by the quality of the data and the number of participants.

#### 11.4 TYPES OF INCENTIVE MECHANISM DESIGN

The incentive mechanism is essential for the survival of the Federated Learning community, and various design principles have been introduced as shown in Figure 11.3. Designs based on contract theory, auction theory, game theory, deep reinforcement learning, and blockchain have been identified based on incentive mechanism design principles [4, 6, 16].



**FIGURE 11.3** Types of incentive mechanism design methods.

### 11.4.1 GAME THEORY-BASED INCENTIVE MECHANISM

An incentive mechanism for federated learning has been designed based on game theory principles from mathematics and economics. The game's participants are the central server and the clients. The study of how the interactions of economic agents produce outcomes based on those agents' choices even when none of the agents intended those outcomes is renowned as game theory [7]. Game theory is used to model rational player conflict and cooperation. The term game has been used to describe a decision-making process involving two or more people. The final result is impacted by the tactical plan of each person involved in the game [19]. The following are game theory's most commonly used terms [7]:

- **Game:** A game is any collection of contexts where the output is determined by the actions of other players.
- **Player:** A player is a logical policy decision-maker who always chooses actions that boost their dividend.
- **Strategy:** The sequence of actions undertaken by the gamers to achieve the intended outcome.
- **Payoff:** The positive or negative reward received as a result of achieving the goal.
- **Equilibrium:** The point at which the ultimate goal has been accomplished.

First, each player will be assigned an objective function. It is encouraged for each player to maximize their utility function when participating in federated training. The utility function can be based on resource usage or data donation. Incentive mechanisms based on game theory primarily focus on optimizing computational and communication resources. To some extent, the game theory approach has addressed the issue of communication round delay. Shapley value-based games, Stackelberg games, Non-cooperative games, and others are the most common incentive mechanisms for federated game-theoretic learning [7, 16, 21].

#### 11.4.1.1 SHAPELY VALUE-BASED MECHANISM

A cooperative game is one in which the strategic approach of each contributor influences the policy of the other. The Shapley value is used to evaluate each client's commitment to the cooperative game. In federated learning, the Shapley value computes each client's marginal score [14, 15]. The sequence in which contestants join does not affect the Shapley value. It distributes a

total surplus produced by the alliance of all members among the groups in a distinct way. Furthermore, the Shapley value enables the contribution of a participant to be defined by a mixture of desirable traits. Shapley values are determined by assessing a player's marginal contribution to all potential player coalitions. In addition to that, each such evaluation includes the training and evaluation of a model. So when dealing with large datasets, the Shapley value index computations become complex. The Shapley value, which implements the marginal contribution, is a widely used index for maneuvering local and global attributes [22].

#### ***11.4.1.2 STACKELBERG GAME-BASED MECHANISM***

The Stackelberg game is a leader-follower game in which participants take turns acting in a precise sequence. In the Stackelberg game, the model holder is the purchaser, as it purchases the learning facility offered by mobile devices. The mobile devices that are content providers then operate as vendors. Stackelberg games, which are considered hierarchical optimization problems, are preferred for both stochastic and dynamic game designs. In this paradigm, the central server (Cloud server) will be the leader and the client devices will be the followers. The game's leader is expected to make the first move, and the followers must act in accordance with the leader's actions. Interactions between the central server and the data owners are critical for joint model training. A Stackelberg game has been designed to demonstrate effective player interactions. Stackelberg's game-based incentive mechanisms are widely used to persuade clients to participate in the group learning process [19]. The reward is proportional to the local model's accuracy. Ref. [26] presented a Stackelberg game model to examine multiple workers' CPU disbursement techniques as well as the model host's funding sources in a synchronous Stochastic gradient descent pass by the model holder. It particularly looked into the impact of the amount available and target margin of error on workers' CPU vitality and training convergence time.

#### ***11.4.1.3 NON-COOPERATIVE GAME-BASED MECHANISM***

In non-cooperative game-based approaches, the concept of leaders and followers does not exist. A player's strategy is determined by the expected payoff, as well as the anticipated actions of rival participants. Each player

is considered selfish, aiming to maximize their own payoff rather than welfare benefits. Cooperation among participants does not affect system performance, and one player's decisions have no impact on the others. Nash equilibrium is a reliable game outcome where players in equilibrium have no incentive to change their strategies unilaterally. Numerous studies have been conducted on non-cooperative incentive mechanisms. In Ref. [25], a two-level dynamic game is proposed to model the interactions between participants in the federated learning service market. To evaluate the dynamic behavior of portable devices with rational behavior, an evolutionary game model [26] is developed.

#### **11.4.2 CONTRACT THEORY-BASED INCENTIVE MECHANISM**

The incentive design based on contract theory overcomes the problem of asymmetric information [16] between the server and clients. Before determining rewards, these schemes generally take into account the computational and communication resources of the devices. They establish a criterion for local data uniformity, and the client selects an agreement item to maximize profits. The devices are initially grouped based on the reliability of their information and the computational hardware they use. Higher payoffs are achieved by devices with greater data value and computation power. The device selects one of the presented agreements and completes the training process. The established incentive mechanism motivates devices by ensuring access to more data, thereby boosting federated learning performance and enhancing incentives for both parties. Contract theory-based mechanisms are particularly effective in environments characterized by comprehensive information imbalance [4]. Since the majority of device costs are associated with data training and model transmission, this method proves to be an efficient way of designing incentive schemes. Contract theory-based mechanisms outperform game theory-based mechanisms in the presence of information asymmetry [35]. To address malicious clients, a reputation measure is introduced, and agreements are made based on the participants' reputation [42].

#### **11.4.3 AUCTION THEORY-BASED INCENTIVE MECHANISM**

Auction theory is the section of economic theory concerned with bid forms and auction participants' conduct. An auction is a type of sales event in

which prospective buyers place price quotes on assets or services in an open or closed format. Buyers and sellers attend auctions because they believe they will get a good deal when purchasing or selling assets. It follows a mathematical model for task and resource allocation and is widely applicable to node selection strategies in distributed computing paradigms [4]. In bidding, there are two types of participants: the auctioneer and the bidders. The auction is governed by an auctioneer, who is represented by the global model owner, and buyers are participants who react to the bidder with numerous local assets and tenders [6]. The most prevalent bid templates are as follows.

#### 11.4.3.1 SEALED-BID AUCTIONS

Sealed-bid auctions are one of the most popular auction formats in economic applications. A sealed bid auction is distinguished by the fact that all bidders submit their bids at the same time. The auctioneer then evaluates the bids and determines the winning bidder and the amount they will pay. Below are two well-known templates of sealed-bid auctions [16]:

1. **First-Price Auction:** In a first-price auction, the highest bidder earns the item and pays their target cost. Other bidders make no payments at all. This is most likely the commonly used sealed-bid format of an auction.
2. **Second-Price Auction:** In this type of auction, the highest bidder earns the asset and pays the second-highest tender. Other buyers do not make any payment.

#### 11.4.3.2 OPEN CRY AUCTIONS

It follows an iterative procedure for the auctions. In each iteration, the supplier declares a cost and prompts tenderers to submit bids at that price. In accordance with demand, the price is raised or lowered. It is divided into two categories:

1. **English Auction:** In this type of auction, bids are placed in ascending order. The auctioneer starts with lower prices and gradually raises them. A buyer who is unwilling to participate in the process or is uninterested in the product may withdraw at any time. The item is awarded to the last remaining bidder in the bidding, who pays the price level where the last bidder dropped out [4].

2. **Dutch Auction:** Investors submit bids for the cost and quantity of items they are interested in purchasing. In a Dutch auction, the price with the most interested parties is chosen as the opening price, so that the entire suggested amount is sold at a single rate [16].

In federated learning, participants are provided with compensation to motivate their effective involvement in the learning process. Recently, an incentive mechanism based on auction theory has emerged. The central server publishes the auctions, with the buyers acting as the bidders. Various machine learning tasks, such as classification or regression algorithms, are published in the federated learning framework [6] to enable participatory learning experience. An instinctual greedy auction mechanism is utilized to assess the bid finalists, making auction theory the preferred incentive mechanism. The server acts as the auctioneer, while the edge devices serve as the sellers. Each edge device accepts a bid based on the cost of power, aiming to select a champion and maximize people's welfare. The server first publishes the auction rules and the federated learning task, after which each edge device determines the required resources. Auction theory-based schemes aim to attract high-quality data owners while reducing latency, computational, and communication resources. Strategies based on the design principle can guarantee critical parameters such as trustworthiness, independent objectivity, and computational efficiency.

#### **11.4.4 BLOCKCHAIN-BASED INCENTIVE MECHANISM**

This section explains incentive schemes based on blockchain technology. The job inventor discloses the federated learning activity in the suggested environment, in addition to a model summary, initial time, the number of involved parties in each round, reward in each round, and overall score incentive [4]. In each round, arbitrary edge devices are designated, and a secret key is conveyed to the particular data stakeholders to obtain the models from the previous update. The compensation is dispersed according to the votes put by each edge device in the preceding round for the top models. Because there are no data holders to vote in the ultimate round, the reward is spread evenly among them. However, this equal division is a concern in and of itself because data owners can request the same update again and obtain the reward through fraud. In most cases, blockchain-based incentive schemes aim to enhance the safety and confidentiality of federated learning systems [16]. Blockchain enables the participants to securely submit transaction records.

It also allows for smart agreement systems, which are extremely efficient for creating and distributing benefits.

#### **11.4.5 DEEP REINFORCEMENT LEARNING-BASED INCENTIVE MECHANISM**

The clients are unable to make a feasible decision since they have inadequate evidence about the policy choices of the other participants. Furthermore, determining the contribution of clients to the accuracy of the global model is difficult [8]. As a result of these complaints, it is challenging to create an optimal incentive scheme, resulting in low client willingness to participate in the training. To address the above issues, a deep reinforcement learning-based incentive mechanism was proposed [2]. It was mainly designed to create an optimal strategy for the model owners and an optimal strategy for the data owners. The incentives are provided based on the contribution of the clients. The primary goal of deep reinforcement learning mechanisms is typically to minimize training time. It has been associated with the game theoretic approach as well as contract theory schemes for incentive designs [39, 43]. These mechanisms facilitate well in cases of information asymmetry because the practices developed by this methodology are based on experience rather than current data.

### **11.5 TYPES OF FEDERATED LEARNING PARADIGM AND ITS ASSOCIATED INCENTIVE POLICIES**

Federated learning focuses on collaborative model training without jeopardizing the user's privacy. The computation overhead is also shared among the various edge devices. Federated learning is divided into horizontal and vertical federated learning based on the distribution of data among devices for model building. Based on the application of federated learning, it is further divided into cross-device and cross-silo federated learning.

#### **11.5.1 HORIZONTAL FEDERATED LEARNING**

When a small number of clients share similar characteristics, the horizontal federated learning technique can be useful. For example, a machine learning model must be built to predict a specific disease. Google proposed

a horizontally fed learning approach [17] for managing Android phone updates. Horizontal federated learning primarily focuses on data security.

Let's consider the task of developing an accurate diabetic prediction model. Clients A and B are the hospitals where training data must be collected. In this case, the sample space is different, but the model is built using the same feature space. The characteristics that require the model to be built will be similar, but the samples will be collected from two different health centers.

The incentive mechanism associated with horizontal federated learning mainly deals with the shapely value mechanism. Each client's contribution is analyzed, and based on the data contribution and resource allocation, the incentives are designed. Game theoretical approaches have been widely used for sample-based federated learning [5]. Reputation and reverse auction-based incentive schemes have been designed for horizontal federated learning settings [40].

### **11.5.2 VERTICAL FEDERATED LEARNING**

Vertical federated learning can be considered when the clients in the network have the same or different datasets in different feature spaces. For example, let's consider a commercial company and a bank located in the same area. They have the same sample space but different attribute spaces. The bank stores customer information related to their accounts, while the company stores information about the customer's demographics and personal interests. A task is published to train a machine-learning model that predicts customer behavior. Since the bank and the company are in the same neighborhood, their customers are mostly the same. The model is trained by collecting characteristics from both organizations. The feature space we consider is large, and training on these characteristics enables the model to predict customer behavior. Vertical federated learning scenarios have utilized auction theory and contract theory-based mechanisms for the appropriate design of incentives [10].

### **11.5.3 CROSS-DEVICE FEDERATED SETTINGS**

Cross-device federated learning (FL), originally developed by Google [17], is a classic scenario where efforts have been dedicated to end users, such as

mobile keyboard assistants and audio keyword recognition [1]. The devices used for cross-device FL support massive parallel processing, allowing millions of devices to participate in the machine learning task and efficiently produce results. In cross-device settings, devices primarily process heterogeneous data, which consists of independent data from various distributions. The computations in cross-device federated learning use Wi-Fi or slower connections. It mainly follows a stateless client strategy, as each client is likely to participate only once in a task, and a new sample of never-before-seen clients is assumed in each round of computation. The devices involved in this process are highly unreliable, as it is expected that 5% or more of the clients participating in a round of computation will fail or drop out [41].

#### 11.5.3.1 CHALLENGES TO CROSS-DEVICE FEDERATED LEARNING

1. **Privacy Leakage:** The model parameters and output vectors, which are transmission items in federated learning systems, may cause sensitive information leakage. Model inversion attacks and ensemble learning framework membership attacks may reveal model parameters and training data. Participants must send the updated gradients or model parameters of each iteration to the server during the federated learning training process, which may expose private training data [1].
2. **Resource Limitations of Devices:** The devices train the model with their data for federated learning. End devices use their data, computational, and communication resources for training. The devices could be constructed from untrustworthy resources with limited computing power [26]. It is a challenge in a federated cross-device learning environment.
3. **Incentive Mechanism:** Clients play a crucial role in the federated learning scenario. However, they may be hesitant to participate in collaborative learning due to privacy concerns and the need to allocate their resources for the purpose of federated learning. As a result, they expect to receive compensation to cover their model training expenses. It is expected that the server will distribute incentives fairly among all participating devices in the collaborative learning ecosystem. Clients should be appropriately rewarded based on their contribution to the federated learning framework, the reputation of

each device gained from previous tasks, or the resource allocation strategy of each device [3].

#### *11.5.3.2 APPLICATIONS OF INCENTIVE MECHANISM IN CROSS-DEVICE FEDERATED LEARNING*

The central server can provide incentives in the form of monetary rewards or model updates. Without incentive design, the federated learning ecosystem may treat all entities the same and distribute the same global model to all, resulting in inequity in the environment. Each client can receive model updates based on their contribution or reputation to the collaborative learning environment. The device that contributes the most data and resources to the federated learning process will receive the most up-to-date global model. In the case of monetary rewards, edge devices that donate more quality and quantity data may be rewarded more [9]. Owners who manipulate trained models for their benefit may receive a negative reward as punishment.

Federated learning has the potential to be incredibly advantageous for self-driving cars. The data collected by the vehicles will be insufficient for model training. The model will be inadequate for predicting unknown situations and criteria. The collaborative learning of multiple vehicles at the same time may improve training results. Autonomous vehicles may encounter various situations while driving and may train the model using this heterogeneous data. The server will only collect the model parameters in an encrypted format, and those updated models will be shared with the cars, who will begin the next round of training with their data. The training results outperform traditional DL algorithms when federated learning is used in autonomous driving cars. The components in the federated learning scenario of autonomous cars can be described in the following way.

1. **Participants:** We assume that the participants (autonomous vehicles) are honest but curious. They do not provide false model parameters dishonestly, but they may try to uncover confidential information about other competitors.
2. **Aggregate Server:** A trustworthy third party collects all of the updated parameters from all of the cars, securely aggregates them, and creates the global model for the next round.
3. **Incentive Mechanism:** Determine the design method based on the evaluation scheme of the client's data or resources. The incentives

could include updated model parameters or a monetary reward using the most popular blockchain application, cryptocurrency.

#### **11.5.4 CROSS-SILO FEDERATED LEARNING**

In this setting, clients are various organizations or geo-distributed data centers. Data is produced locally and kept federated. Each client maintains its own records and cannot access the data of other clients. The data is not uniformly distributed or completely independent. The primary bottleneck in cross-silo settings is the communication and computation process. The cross-silo framework is considered stateful, which means that every client can participate in each cycle of calculation, carrying the state from one session to the next [42].

##### **11.5.4.1 CHALLENGES IN CROSS-SILO SETTING**

Below are a few practical challenges faced in the cross-silo settings of federated learning:

1. **Efficient and Effective Implementation:** One significant challenge is implementing cross-silo federated learning successfully and efficiently. Effectiveness is the capacity to normalize global (and local) models while taking into account various client heterogeneity, whereas efficiency refers to the ability to obtain an effective model quickly and at a minimal cost [13].
2. **Security and Privacy:** The cross-silo environment is vulnerable to various types of attacks, including evasion and membership inference attacks. Maintaining security in the development and deployment of federated learning across multiple organizations has become difficult. The organizations are located in various countries and participate in the collaborative process of the machine learning task. GDPR and other privacy regulations limit the transfer of raw data to a central server for centralized training. Each organization may have its privacy policies that limit the global transfer of sensitive data [16].
3. **Long-Term Cooperation and Incentive Mechanism:** The organization that participates in the cross-silo setting has a long-term strategic plan and development goals. The issue of reluctance to participate in cooperative learning can be resolved by utilizing the incentive mechanism [42].

#### ***11.5.4.2 APPLICATIONS OF INCENTIVE MECHANISM IN CROSS-SILO FEDERATED LEARNING***

The cross-silo federated learning is mostly applicable to the healthcare sector or the fin-tech sector. The central authority first publishes the machine learning task, which is then distributed to a group of clients who can contribute to the task and have demonstrated a willingness to actively participate in the long run. The organizations that participate in the collaborative learning mechanism develop long-term strategic plans and development goals. The organizations that participate in the process also anticipate a long-term relationship with the ecosystem. The cross-silo federated setting may have two or more participants. Clients are mostly large corporations with high-quality data. Due to privacy concerns, institutions are not permitted to share sensitive data with a third party or with one another [42]. The restrictions for sharing data across organizations or to other countries may be included in the privacy policies.

##### ***11.5.4.2.1 Federated Learning Application in the Health Care Sector and the Associated Incentive Mechanism***

A large dataset with high-quality data is required for machine learning tasks, such as cancer prediction. Each hospital must have a dedicated cancer section, and the authorities must maintain records of patients diagnosed with cancer. The patient's personal information, symptoms, and diagnostic methods are highly sensitive and should not be shared with anyone outside the department. In this scenario, cross-silo federated learning becomes crucial. A trusted third party acts as the central server, inviting hospitals to participate in collaborative learning by utilizing their resources and private data to build accurate deep learning or machine learning models. The hospitals' development goals and strategic plans are long-term, so participants naturally engage in the learning process when appropriate incentives are provided [1]. The central server publishes the task, such as building a machine learning model to predict cancer, and shares the initial global model with all hyperparameters with the participants. Hospital departments download the initial model and train it using their private data and resources. The model is iteratively built, and the updated model parameters are encrypted and sent to the central server [11]. The central server securely aggregates all model updates from participating hospital departments in the learning program and updates the global model. These updated model parameters are shared with the learning

participants for the next round of model training, and the process is repeated until the model achieves convergence or the desired accuracy of the machine learning model is attained.

#### ***11.5.4.2.2 Federated Learning Application in the Finance Sector and the Associated Incentive Mechanism***

Every second, millions of transactions are processed by financial institutions, and the data generated by these transactions, when properly mined, can yield meaningful insights. However, there is a problem with data islands. There is a wealth of data available, but it is spread across multiple organizations. The privacy regulations of these institutions prohibit collecting all data in a single location and conducting training in a single location. In this case, federated model training can be used. For example, if a fraudulent transaction occurs at several institutions simultaneously, a machine learning or deep learning model can be trained to predict the classification of fraudulent transactions. The global model can then be shared with all participating institutions, enabling real-time identification of fraud transactions for accurate results. To encourage active participation in shared model training, institutions are rewarded. These rewards can be in the form of monetary incentives or model updates. The rewards are also intended to compensate institutions for the loss of privacy and resource consumption. Various methods, such as game theory, contract theory, or auction mechanisms, can be used to evaluate and assign incentives [10, 20, 43].

## **11.6 CONCLUSION**

Scalable machine learning algorithms are in high demand due to the widespread use of AI technologies. Federated learning with privacy protection has emerged as a future technology in the distributed learning platform. Clients contribute user-sensitive data and build machine learning or DL models with their resources. Each device's active and long-term participation in this collaborative learning process is significant. Effective client involvement is required to ensure the smooth operation of the federated learning process. The compensation scheme for the use of data and resources encourages the client to participate in the training procedure. The various incentive schemes described in this chapter address client heterogeneity, information asymmetry, and resource constraint issues. The federated learning application scenario,

as well as the incentive mechanism associated with cross-device and cross-silo applications, are characterized. Real-time federated learning use cases can generate new types of incentive designs, and privacy-enabled incentive distribution can be future research directions.

## KEYWORDS

- **cross-device setting**
- **cross-silo setting**
- **federated learning**
- **game theory**
- **healthcare sector**
- **incentive mechanism**
- **learning platform**
- **machine learning**

## REFERENCES

1. Li, T., Sahu, A. K., Talwalkar, A., & Smith, V., (2020). Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3), 50–60.
2. Ye, Y., Li, S., Liu, F., Tang, Y., & Hu, W., (2020). EdgeFed: Optimized federated learning based on edge computing. *IEEE Access*, 8, 209191–209198.
3. Nair, A. K., Ebin, D. R., & Jayakrushna, S., (2023). A robust analysis of adversarial attacks on federated learning environments. *Computer Standards & Interfaces*, 103723.
4. Zhan, Y., Zhang, J., Hong, Z., Wu, L., Li, P., & Guo, S., (2021). A survey of incentive mechanism design for federated learning. *IEEE Transactions on Emerging Topics in Computing*.
5. Zhao, J., Zhu, X., Wang, J., & Xiao, J., (2021). Efficient client contribution evaluation for horizontal federated learning. In: *ICASSP 2021–2021 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)* (pp. 3060–3064).
6. Zhan, Y., Li, P., Qu, Z., Zeng, D., & Guo, S., (2020). A learning-based incentive mechanism for federated learning. *IEEE Internet of Things Journal*, 7(7), 6360–6368.
7. Tu, X., Zhu, K., Luong, N. C., Niyato, D., Zhang, Y., & Li, J., (2022). Incentive mechanisms for federated learning: From economic and game theoretic perspective. *IEEE Transactions on Cognitive Communications and Networking*.
8. Zhan, Y., Li, P., Guo, S., & Qu, Z., (2021). Incentive mechanism design for federated learning: Challenges and opportunities. *IEEE Network*, 35(4), 310–317.

9. Khan, L. U., Pandey, S. R., Tran, N. H., Saad, W., Han, Z., Nguyen, M. N., & Hong, C. S., (2020). *Federated Learning for Edge Networks: Resource Optimization and Incentive Mechanism* (Vol. 58, No. 10, pp. 88–93). IEEE Communications Magazine.
10. Sarikaya, Y., & Ercetin, O., (2019). Motivating workers in federated learning: A Stackelberg game perspective. *IEEE Networking Letters*, 2(1), 23–27.
11. Yu, H., Liu, Z., Liu, Y., Chen, T., Cong, M., Weng, X., & Yang, Q., (2020). A sustainable incentive scheme for federated learning. *IEEE Intelligent Systems*, 35(4), 58–69.
12. Song, T., Tong, Y., & Wei, S., (2019). Profit allocation for federated learning. In: *2019 IEEE International Conference on Big Data (Big Data)* (pp. 2577–2586). IEEE.
13. Kairouz, P., McMahan, H. B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A. N., & Zhao, S., (2021). Advances and open problems in federated learning. *Foundations and Trends® in Machine Learning*, 14(1, 2), 1–210.
14. Yang, X., Tan, W., Peng, C., Xiang, S., & Niu, K., (2022). Federated learning incentive mechanism design via enhanced Shapley value method. *Wireless Communications and Mobile Computing*, 2022.
15. Ghorbani, A., & Zou, J., (2019). Data Shapley: Equitable valuation of data for machine learning. In: *International Conference on Machine Learning* (pp. 2242–2251). PMLR.
16. Ali, A., Ilahi, I., Qayyum, A., Mohammed, I., Al-Fuqaha, A., & Qadir, J., (2023). *Incentive-Driven Federated Learning and Associated Security Challenges: A Systematic Review*. Authorea Preprints.
17. Hard, A., Rao, K., Mathews, R., Ramaswamy, S., Beaufays, F., Augenstein, S., & Ramage, D., (2018). *Federated Learning for Mobile Keyboard Prediction*. arXiv preprint arXiv:1811.03604.
18. Liu, Y., Xu, C., Zhan, Y., Liu, Z., Guan, J., & Zhang, H., (2017). Incentive mechanism for computation offloading using edge computing: A Stackelberg game approach. *Computer Networks*, 129, 399–409.
19. Dong, L., & Zhang, Y., (2020). Federated learning service market: A game theoretic analysis. In: *2020 International Conference on Wireless Communications and Signal Processing (WCSP)* (pp. 227–232). IEEE.
20. Lee, J., Kim, D., & Niyato, D., (2020). Market analysis of distributed learning resource management for internet of things: A game-theoretic approach. *IEEE Internet of Things Journal*, 7(9), 8430–8439.
21. Cong, M., Yu, H., Weng, X., & Yiu, S. M., (2020). A game-theoretic framework for incentive mechanism design in federated learning. In: *Federated Learning* (pp. 205–222). Springer, Cham.
22. Jia, R., Dao, D., Wang, B., Hubis, F. A., Hynes, N., Gürel, N. M., & Spanos, C. J., (2019). Towards efficient data valuation based on the Shapley value. In: *The 22<sup>nd</sup> International Conference on Artificial Intelligence and Statistics* (pp. 1167–1176). PMLR.
23. Hu, R., & Gong, Y., (2020). Trading data for learning: Incentive mechanism for on-device federated learning. In: *GLOBECOM 2020–2020 IEEE Global Communications Conference* (pp. 1–6). IEEE.
24. Qu, X., Hu, Q., & Wang, S., (2020). Privacy-preserving model training architecture for intelligent edge computing. *Computer Communications*, 162, 94–101.
25. Zou, Y., Feng, S., Xu, J., Gong, S., Niyato, D., & Cheng, W., (2019). Dynamic games in federated learning training service market. In: *2019 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM)* (pp. 1–6). IEEE.

26. Zou, Y., Feng, S., Niyato, D., Jiao, Y., Gong, S., & Cheng, W., (2019). Mobile device training strategies in federated learning: An evolutionary game approach. In: *2019 International Conference on the Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) IEEE Cyber, Physical and Social Computing (CPSCom), and IEEE Smart Data (SmartData)* (pp. 874–879). IEEE.
27. Pandey, S. R., Tran, N. H., Bennis, M., Tun, Y. K., Manzoor, A., & Hong, C. S., (2020). A crowd sourcing framework for on-device federated learning. *IEEE Transactions on Wireless Communications*, 19(5), 3241–3256.
28. Mothukuri, V., Parizi, R. M., Pouriyeh, S., Huang, Y., Dehghantanha, A., & Srivastava, G., (2021). A survey on security and privacy of federated learning. *Future Generation Computer Systems*, 115, 619–640.
29. Nair, A. K., Jayakrushna, S., & Ebin, D. R., (2023). Privacy-preserving federated learning framework for IoMT-based big data analysis using edge computing. *Computer Standards & Interfaces*, 103720.
30. Rodrigues, T. K., Suto, K., Nishiyama, H., Liu, J., & Kato, N., (2019). Machine learning meets computation and communication control in evolving edge and cloud: Challenges and future perspective. *IEEE Communications Surveys & Tutorials*, 22(1), 38–67.
31. Khan, L. U., Saad, W., Han, Z., Hossain, E., & Hong, C. S., (2021). Federated learning for the Internet of things: Recent advances, taxonomy, and open challenges. *IEEE Communications Surveys & Tutorials*.
32. Konečný, J., McMahan, H. B., Ramage, D., & Richtárik, P., (2016). *Federated Optimization: Distributed Machine Learning for on-device Intelligence*. arXiv preprint arXiv:1610.02527.
33. Lim, W. Y. B., Xiong, Z., Kang, J., Niyato, D., Zhang, Y., Leung, C., & Miao, C., (2020). An incentive scheme for federated learning in the sky. In: *Proceedings of the 2<sup>nd</sup> ACM MobiCom Workshop on Drone Assisted Wireless Communications for 5G and Beyond* (pp. 55–60).
34. Wang, S., Tuor, T., Salonidis, T., Leung, K. K., Makaya, C., He, T., & Chan, K., (2018). When edge meets learning: Adaptive control for resource-constrained distributed machine learning. In: *IEEE INFOCOM 2018-IEEE Conference on Computer Communications* (pp. 63–71). IEEE.
35. Zhang, Y., Pan, M., Song, L., Dawy, Z., & Han, Z., (2017). A survey of contract theory-based incentive mechanism design in wireless networks. *IEEE Wireless Communications*, 24(3), 80–85.
36. Li, J., Chen, H., Chen, Y., Lin, Z., Vučetić, B., & Hanzo, L., (2016). Pricing and resource allocation via game theory for a small-cell video caching system. *IEEE Journal on Selected Areas in Communications*, 34(8), 2115–2129.
37. Huang, J., Talbi, R., Zhao, Z., Boucchenak, S., Chen, L. Y., & Roos, S., (2020). An exploratory analysis of users' contributions in federated learning. In: *2020 Second IEEE International Conference on Trust, Privacy, and Security in Intelligent Systems and Applications (TPS-ISA)* (pp. 20–29). IEEE.
38. Sim, R. H. L., Zhang, Y., Chan, M. C., & Low, B. K. H., (2020). Collaborative machine learning with incentive-aware model rewards. In: *International Conference on Machine Learning* (pp. 8927–8936). PMLR.
39. Cheng, W., Zou, Y., Xu, J., & Liu, W., (2021). Dynamic games for social model training service market via federated learning approach. *IEEE Transactions on Computational Social Systems*, 9(1), 64–75.

40. Zhang, J., Wu, Y., & Pan, R., (2021). Incentive mechanism for horizontal federated learning based on reputation and reverse auction. In: *Proceedings of the Web Conference 2021* (pp. 947–956).
41. Yang, W., Wang, N., Guan, Z., Wu, L., Du, X., & Guizani, M., (2022). A practical cross-device federated learning framework over 5G networks. *IEEE Wireless Communications*.
42. Huang, C., Huang, J., & Liu, X., (2022). *Cross-Silo Federated Learning: Challenges and Opportunities*. arXiv preprint arXiv:2206.12949.
43. Tian, M., Chen, Y., Liu, Y., Xiong, Z., Leung, C., & Miao, C., (2021). *A Contract Theory Based Incentive Mechanism for Federated Learning*. arXiv preprint arXiv:2108.05568.



Taylor & Francis

Taylor & Francis Group

<http://taylorandfrancis.com>

## CHAPTER 12

---

# Protected Shot-Based Federated Learning for Facial Expression Recognition

A. SHERLY ALPHONSE RAO<sup>1</sup> and J. V. BIBAL BENIFA<sup>2</sup>

<sup>1</sup>*Vellore Institute of Technology, Chennai, Tamil Nadu, India*

<sup>2</sup>*Indian Institute of Information Technology, Kottayam, Kerala, India*

---

### ABSTRACT

Numerous applications of human-centric computing rely on facial expression recognition (FER), which is crucial. Large-scale annotated datasets and developments in deep learning (DL)-aided computer vision have significantly improved FER performance. The centralization of user data into one computer or a shared database is a requirement of standard machine learning methodologies, which raises questions about data privacy and confidentiality. Facial expression data has become a significant research bottleneck in FER due to its privacy-sensitive nature. Strict data protection regulations are being implemented in an effort to end the voluntary collection of users' private data without obtaining explicit permission from the individual user. As a result of this non-consent-based facial data gathering, many publicly accessible face image datasets have been removed from the internet. Currently, Federated Learning (FL) is a successful method for training models on decentralized private data. FL enables several parties to cooperatively create a joint prediction model by leveraging parameters in the trained models. This chapter aims to provide privacy-preserving and unobtrusive facial emotion prediction for long-term AI-assisted emotion monitoring. This chapter proposes a novel Protected-Shot-based Federated Learning (PSFL) framework for facial emotion recognition. PSFL enables collaborative training of a deep

---

Federated Learning: Principles, Paradigms, and Applications.

Jayakrushna Sahoo, Mariya Ouassa, & Akarsh K. Nair (Eds.)

© 2025 Apple Academic Press, Inc. Co-published with CRC Press (Taylor & Francis)

model across various subjects without sharing their face photos. It utilizes a lightweight Convolutional Neural Network (CNN) architecture. Unlike standard FL, which shares all model parameters, PSFL keeps the last layer local. This adds an extra layer of data security, making it more difficult for an adversary to infer the target subject's emotional state. At the same time, it allows for local parameter customization to tailor the emotion estimation for each subject. Experimental results using publicly accessible datasets of facial expressions demonstrate that PSFL performs comparably to other centralized and distributed algorithms, while also enhancing data privacy. Consequently, traditional emotion monitoring in real-time applications, such as improving work environments in offices, can benefit from PSFL's improved protection, computational efficiency, and scalability to a large number of people.

## 12.1 INTRODUCTION

Facial Expression Recognition (FER), which is essential, is used in many human-centric computing applications. FER performance has greatly increased thanks to the emergence of large-scale annotated datasets and advances in deep learning (DL)-aided computer vision. Standard machine learning approaches call for the centralization of user data into a single computer or a public database, which creates concerns about data privacy and confidentiality. Due to its privacy-sensitive nature, facial expression data has become a major research bottleneck in FER. Strict data protection laws are being established to prevent the voluntary collection of users' private data. Numerous public face image datasets have been taken down from cyberspace as a result of non-consent-based facial data harvesting. The current successful technique for training models using decentralized private data is federated learning (FL) [1]. By utilizing the parameters of locally trained models and storing the actual training data locally, FL enables several participants to collaboratively develop a shared prediction model. The goal of this chapter is to offer unobtrusive and privacy-preserving facial expression prediction for ongoing AI-assisted facial emotion monitoring. This chapter proposes a novel Protected-Shot-based Federated Learning (PSFL) architecture for facial emotion recognition in order to achieve this. PSFL uses a lightweight Convolutional Neural Network (CNN) architecture to perform collaborative training of a deep model across several individuals, without disclosing the subjects' face images. Without sharing the participants' face photos, PSFL performs combined training of a deep model by a CNN. As opposed to standard FL, which shares every model parameter, PSFL keeps the last layer

local. This provides an additional layer of security, making it harder for an opponent to guess the target subject's emotional state while allowing for local parameter customization to tailor the emotion estimation for each subject. Using publicly accessible datasets of facial expressions, experiments show that PSFL performs on par with the standard centralized and FL algorithms while also improving privacy. Therefore, traditional emotion monitoring may be enhanced by being more protected, computationally effective, and scalable to a great number of users in a real-time application like improving work environments in offices. The argument is used by the server together with a global face feature extractor. The client initializes the mean feature in the class embedding and the server relays the current global settings to the client. According to the local data, the client updates the model parameters. The client then updates the server's parameters and sends them back. The server compiles the new parameters from each client. The server then uses the spread-out regularizer [2] to separate the class embeddings. These actions are repeated for  $T$  communication rounds, after which the global parameter is restored. The benefits of this work can be summarized as follows:

- This work enables the assessment of emotions while maintaining the secrecy of facial images. By utilizing a challenging dataset consisting of real-world and spontaneously expressed facial emotions, we demonstrate the effectiveness of custom FL with deep models. The main advantage of our approach is the confidential protection of user data. Data such as facial pictures and emotion labels are always stored locally.
- This paper presents federated personalization, a novel variation of the conventional FL algorithm that provides an additional layer of data security. In our method, the classification layer in the deep model for emotion estimation is retained rather than using all the model parameters for communication rounds between the user-specific models. This makes it difficult for an adversary to deduce the user's emotional expressions while enabling the model to tailor its estimates to the intended user. The general framework of our customized FL method for emotion assessment in a genuine environment is depicted in Figure 12.1.

## 12.2 LITERATURE SURVEY

Google originally implemented federated machine learning in 2016 [3]. In contrast to a centralized setting, an FL setting uses a variety of devices, such as end-user hardware, mobile phones, or business hardware like hospital

servers, to develop a classifier. The classifier could be a deep neural network, but it could also be a smaller model with fewer parameters than a logistic regression model or a support vector machine. The training data for FL models is always present on the specific local device that collected it, which makes them unique. Every device, also known as a client, preserves a copy of a similar model, which is reorganized each time an observation is made. A central server then finds the average of the model parameters from all active devices, using the model updates, i.e., the reorganized weights and biases of neurons in a deep network rather than the observations in our example, the face photos themselves. A fresh version of the model is distributed to all clients once it has undergone training. This cycle continues until the model converges or the training set of data runs out. The normal federated learning (FL) approach averages all model parameters, whereas in our method, the final (decision) layers of the deep models are not shared and are updated only locally. This is the main distinction between the PSFL approach described in this study and the standard FL. The method presented here is also connected to the recently suggested concept of split learning, where only a portion of the model or a smaller subset of its parameters is shared among several clients. Split learning [4] delays the development of the deep model's initial layers, not the last layer, as claimed above, as this is in the context of emotion. To enable effective model customization, estimation is essential.

### **12.2.1 APPLICATIONS OF FEDERATED LEARNING**

Although FL's potential applications [5] are numerous and greatly vary in vertical and particular use cases, they often have some characteristics recognized by people. Training datasets are large and challenging to gather in one place, and the training data is sensitive to confidentiality. When using FL, not all of these requirements must be met exactly, but in these situations, FL typically outperforms other machine learning methods. Here, a few recent FL model applications are discussed to demonstrate the possibilities of FL. For instance, the computer chip manufacturer Intel uses FL to demonstrate how different healthcare institutions [6] can cooperate while maintaining patient privacy by utilizing each institution's Electronic Health Information (EHR). While cooperation between institutions might help with the problem of gathering enough data required to train machine learning classifiers, the distribution of facial images is tightly controlled and constrained. Some publications demonstrate the first instance of a multi-institutional FL classifier in practice and discover that

they can develop a federated semantic model for brain tumor segmentation that performs better in comparison to a model trained on centralized data. Recent publications provide additional support for the advantages of using FL from health records under data privacy restrictions [7–9]. FL has been utilized for purposes other than health, such as enhancing search results in FL by utilizing information from 360K users. Every day, millions of URLs are entered into Firefox, significantly increasing user experience, and the auto-complete tool is useful in gaining knowledge and boosting retention. Similarly, Google describes one of the first extensive FL implementations developing a worldwide approach “to enhance the quality of virtual keyboard search suggestion” [10]. There are a few technical coordination issues while training the model across millions of devices globally, including difficulties with connectivity, the bias associated with training a model in different time zones, and minimizing the effect. The aforementioned research empirically shows the advantages of Federated Learning (FL) on practical issues. However, the FL technique has never been investigated in the context of emotion assessment from face photos. This is a difficult topic since the data is non-stationary, and the same person may exhibit various levels of emotions in different recording sessions. Furthermore, as expected in real-world applications, the underlying data distribution evolves in spaces with different camera orientations, lighting, etc., making it difficult to distinguish between the face-image data of seven emotion categories.

### **12.2.2 REAL-TIME CHALLENGES IN FEDERATED LEARNING**

FL is sensitive to some of the data attributes, just like traditional machine learning [11]. The sample can be considered representative of the entire population (independent and identically distributed, or IID) if a sufficiently large sample is randomly drawn from the overall data distribution. Client datasets in FL often differ significantly from those of other clients. Specifically, there are notable individual differences in the morphology and texture of the face, the dynamics of facial expression, and the expressiveness of emotions. The size and diversity of the clients’ data can also vary significantly. As mentioned earlier, some participants exhibit much more subtle emotions than others in the dataset used for emotion estimation in this study. Additionally, having a higher number of samples for each subject results in a greater number of samples classified in that specific category [12], which is a drawback of unbalanced classes. Furthermore, when dealing with real-world

subjects and longitudinal recording sessions, some participants may or may not participate in specific sessions. The majority of the subjects only had data for the first session. Despite the fact that this offers an environment for learning that is extremely challenging for both centralized learning and federated learning (FL), the latter scenario is more apparent. It is crucial for FL to have access to learned models before averaging their parameters, as global models are created by averaging the local model parameters. Therefore, any corrupted local models trained on highly unbalanced data could negatively impact the parameters of the global model. Mobile phones often have unstable or expensive internet connections, or they may be offline. Some organizations frequently experience poor internet connectivity or have a limited number of computers connected to the internet, especially in rural areas of affluent nations. Local updates are typically computationally inexpensive, but when the amount of training data is minimal, transmitting these local updates becomes time-consuming. This results in communication speed and averaging becoming the primary bottleneck in FL. In this study, the FL of different models for emotion estimates in the actual world is replicated using data of naturalistic facial expressions of pain made by several participants at various sessions. However, it should be noted that not all of the subjects in this dataset were present for all of the recorded sessions, and/or they may have had brief sessions without any expression of emotion (neutral). The data used to assess the robustness of our algorithms are under these difficult circumstances, even if it is expected to have a negative impact on the FL of the deep models for emotion estimate.

Another crucial feature of FL is that, because only local models have access to the raw data, the global model obtained by combining the parameters of the federated models is trained using only the parameters of locally tuned models for each client and not on the raw data of facial photos. Therefore, in contrast to a centralized machine learning approach, novel learning strategies are required to achieve comparable performance in a federated scenario. A model is fundamentally useless if it completely protects a client's anonymity but makes incorrect predictions. Therefore, a baseline global model trained on a subpopulation of subjects who consented to submit their face data is needed to initiate the learning of our FL models [1, 6]. A model is employed as a starting point for additional learning via FL after it has been instantiated and trained centrally. The purpose of the learning process at this point is to create model parameters that would allow improved generalization and prevent regression in the target subjects' performance when it comes to emotion estimation. In conclusion, the issues with FL that were previously discussed are very significant and create a significant

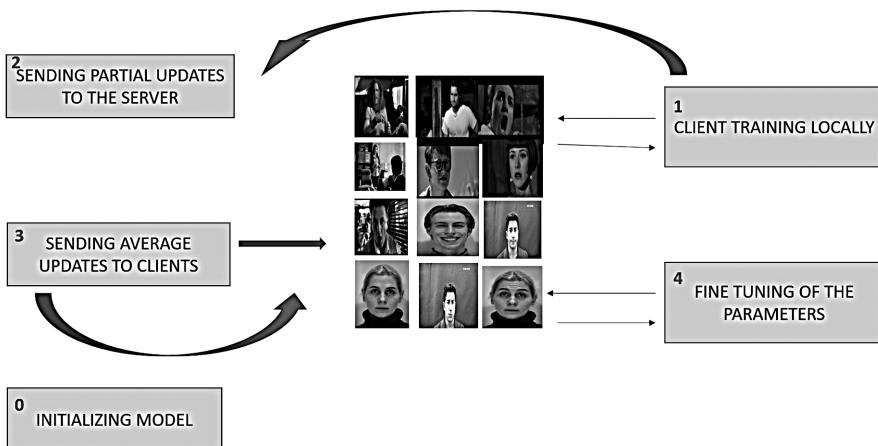
bottleneck for FL algorithms. This work addresses the difficulty of learning from non-IID and unbalanced data by using targeted picture pre-processing and class-stratified sampling of the client's data to mitigate such challenges. This proposed work develops a novel training and evaluation scheme that enables learning the model parameters one session at a time from the target client to address the challenges associated with limited communication and performance maintenance.

### 12.3 PROPOSED MODEL

A global face feature extractor is utilized by the server using the argument in the proposed PSFL. The client initializes the class embedding with the mean feature. Except for the first round, when the client initializes itself, the server relays the current global settings to the client. The client adjusts the model parameters using the local data. The client then provides the changed parameters to the server, and the server compiles all of the clients' updated parameters. The server employs the spread-out regularizer as the final technique to differentiate between the class embeddings. These aforementioned processes are repeated for  $T$  communication cycles. Afterward, the global parameter required for feature extraction is returned. The class embeddings are made more typical by confining the features to a  $d$ -dimensional hypersphere.

Federated Personalized Deep Learning (DL) is utilized in Privacy-Sensitive Federated Learning (PSFL) for Face Image-Based Emotion Assessment. The deep model is initially pre-trained based on data obtained from non-target participants. The target subject-specific models are then initialized using the parameters of this model and trained locally using the subject data. The local client models the parameters of the final two fully connected thick layers, but not the convolutional layers, to protect the privacy of the participants by utilizing the federated averaging method. The local sites receive the updates, and only the two thick layers are further adjusted to estimate the subject-specific facial expressions. The four actions, as seen in Figure 12.1, are spontaneously repeated for the fresh recording sessions. The features are extracted using a novel Fan beam method [15] of feature extraction while creating the models for the face images available from the datasets. On the server, there is a face extractor trained on a dataset that is easily accessible to the public or that has previously been parameterized by 0. The server can pre-train the system using certain datasets that are available to the public. Using the server, there is a face expression recognition system

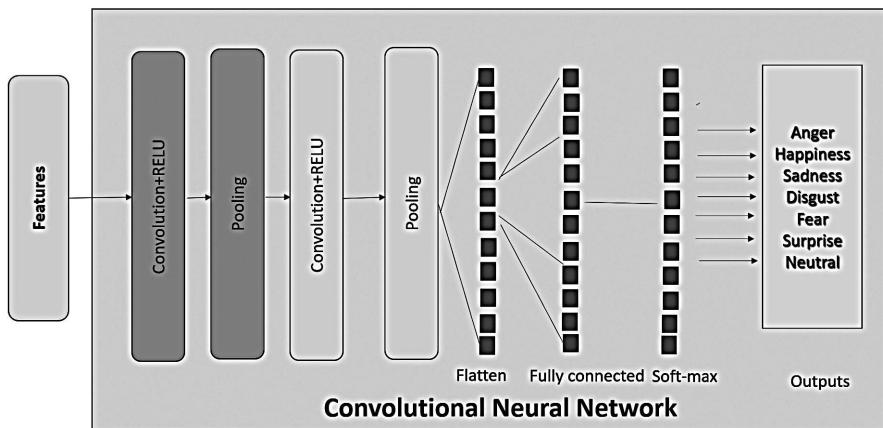
$f_{\theta_0}$  that has been trained on datasets that are open to the general public. The  $i^{\text{th}}$  client node contains all of one identity's  $n_i$  instances (the owner of the device). The dataset is thus available to the  $i^{\text{th}}$  client as a result. The routine of the emotion recognition system can be enhanced by joint learning using the data from each client without transferring face photos external to the device.



**FIGURE 12.1** The proposed architecture.

The video sequences from the databases have frames that are color images with a size of 250 by 250 pixels. All photographs are converted to grayscale because color is irrelevant to the FACS system's ability to detect emotion, which allowed us to reduce the number of input features sent to the network by half. This was followed by histogram equalization, which increased the contrast for each image. The emotion scale is determined by observing only a small number of features in a person's image. There is an imbalance in the number of samples in each class, which is also expected in a "real-world" scenario. The sample size is increased for balancing by using two data augmentation techniques: First, a flipped duplicate of all images is created. After that, a duplicate of each original image is made and flipped. These duplicates were randomly rotated 10 degrees to the left or right, depending on the copy. To remove any whitespace or false filling in the photos after rotation, each image was resized from 250 x 250 pixels to 215 x 215 pixels. By using two data augmentation strategies, the positive examples are up sampled. First, a flipped duplicate of all images is created, and then a duplicate of each original image is made and flipped. The copies are randomly rotated 10 degrees to the left or right, respectively. After rotating the images, each image

is resized from 250 x 250 to 215 x 215 pixels to remove any unnecessary whitespace or filler. A compact CNN-based model architecture, as shown in Figure 12.2, is created for mobile vision applications inspired by CNNs [16] for Mobile Vision Applications. It can be trained for a small number of epochs on hardware-limited computer apparatus. Similar structures have already shown success in recognizing facial expressions from face images, including painful facial expressions. This fundamental deep model for estimating emotion accepts a 215x215 rescaled face image. This representation is then input into three convolutional layers for feature extraction. The first layer has a kernel size of 5x5 and 32 filters. This is followed by a ReLU layer and a batch normalization (BN) layer. The following convolutional layers consist of two fully connected layers with 128 units and one unit plus the sigmoid activation, followed by 2x2 max-pooling (with stride 2) and convolutional layers. The binary cross-entropy-based loss function and the SGD optimizer with a learning rate of 0.1 were employed. If there is no improvement, early stopping is used. This helps in reducing overfitting.



**FIGURE 12.2** The convolutional neural network.

### 12.3.1 FEDERATED PROTECTION OF FACIAL IMAGES/SHOTS USING NOVEL FAN BEAM METHOD OF FEATURE EXTRACTION

The server transmits this global model to the respective clients after initializing a feature extractor with the values  $\theta_0$ . Each client must initialize the class embedding  $w_0^i$  for themselves during the initial communication round. This class is locally efficient using the local data and the parameters

$\theta_t$ . In most cases, the clients use a Gaussian distribution to randomly initialize the class embedding for themselves. The pre-trained feature extractor results in subpar performance on general emotion recognition tasks because the class embeddings may lie external to the feature space of the face extractor, trained earlier using the proposed fan beam method of feature extraction. To solve this problem, FedFace [17], an existing approach, uses the mean of the embeddings that are accessible at the client and were extracted using the already trained  $f_{\theta_0}$ , which is a fan beam extractor. This is used for the initialization of the class embeddings to address this issue.

The face detector developed by Viola Jones and Jones, which uses Haar characteristics, has been extensively used in previous research for face detection. It operates by employing a series of classifiers, with each classifier relying on the samples accepted by the preceding classifier. Once a general model has been built from the training samples in the Chehra [18] face detector, landmarks are initialized using a series of simultaneous regression methods. When new data is received, the generic model is incrementally trained using a cascade of regression methods. The shape parameters or landmarks are constrained by the Scale Invariant Feature Transform (SIFT) characteristics of the new image, using the cascade of the regression function, which is independent of the preceding levels. This makes it suitable for “in the wild” situations when photographs are taken in natural settings. A bounding box is created using the landmarks that have been fitted, and it is utilized to crop the face. The system can be adjusted to real-time environments thanks to the utilization of Chehra face detection. The detected faces undergo feature extraction based on fan-beam, which is then used to create a model for the proposed PSFL. In this approach, PSFL utilizes hypersphere-based normalization [18] to achieve improved performance. While patterns in the data can be easily observed by the naked eye, it becomes more challenging to identify them if the data dimension exceeds three. To identify patterns and perform classification using a classifier, feature extraction techniques are employed. The unique feature extraction method proposed in this paper focuses on fan-beam transforms and their projection data. The photos are converted to grayscale images after being cropped from the raw images in the datasets. Feature vectors can then be extracted using feature extraction techniques. Radon transforms are versions of fan-beam projections. The projections of an image can be obtained by utilizing fan-beam transformation functions [1].

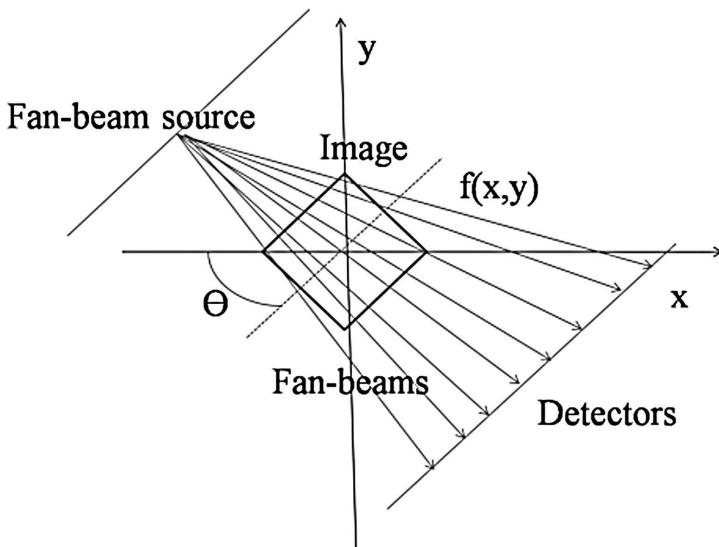
$$L = \sqrt{m^2 + n^2} \quad (1)$$

where the image’s height and width are denoted by m and n. After that, D is set to be higher than  $L/2$ . D is the separation between the rotational object’s

center and the vertex of the fan beam. The center pixel for this rotation is identified as:

$$\text{floor}((\text{size}(Facial\_image)+1)/2) \quad (2)$$

The rotation angle values are acquired sequentially in degrees. Here, the beams separate from the source like fans. In accordance with Figure 12.3, the source emits the beams, and the detector detects them. The coordinates are in the ranges of  $\pi/2 \leq \sigma \leq \pi/2$  and  $0 \leq \beta \leq \pi/2$ , and the angle represents the position of the source. As seen in Figures 12.3 and 12.4, the projection beams are indicated by the symbols  $\sigma$  and  $\beta$ .



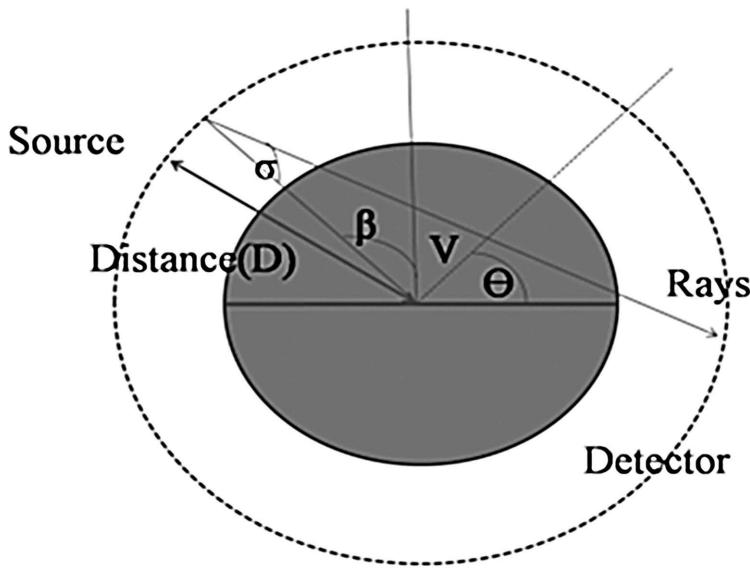
**FIGURE 12.3** Fan beam geometry.

The coordinates of the fan beams are  $V$  and  $\Theta$  as in Eqns. (3) and (4).

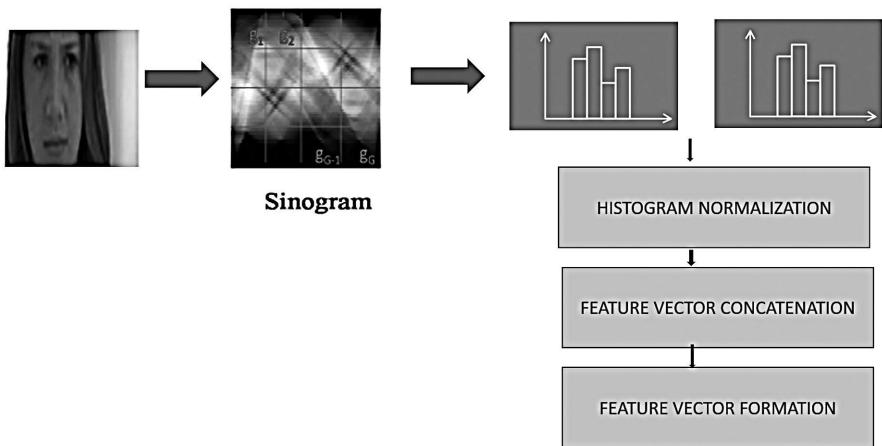
$$V = D \sin \sigma \quad (3)$$

$$\Theta = \sigma + \beta \quad (4)$$

One column of the sinogram is used to represent the projection data that was obtained at one rotation angle. As a result, the sinogram or fan-beam projection data is calculated from image I. This technique for gathering sinograms by fan-beam projections is quicker and captures a lot of information about each image. Each image in this sinogram has a different pattern, as shown in Figure 12.5.



**FIGURE 12.4** Fan beam projection at rotation angle  $\Theta$ .



**FIGURE 12.5** Feature vector extraction.

Next, each image's sinogram is split into grids, and the histograms for each grid are gathered to create a feature vector, or  $f_i$ , as in Eqn. (5).

$$f_i = \langle H_{i_1}, H_{i_2}, \dots, H_{i_G} \rangle \quad (5)$$

Where  $G$  is the total number of grids, and  $f_i$  is the feature vector of the image. Four different classifiers were used to categorize the feature vectors into the seven different emotional states: anger, disgust, happiness, fear, sadness, surprise, and neutrality. The experimental results section discusses the performance variations of these classifiers in terms of time and accuracy. The feature vector is denoted as  $f_{\theta_0}(x_j^i)$  in the Federated Emotion Recognition algorithm described in the subsequent section. Emotion recognition algorithms can be considered as embedding-based classifiers, as both the input instance and the classes (identities) are embedded in a high-dimensional Euclidean space. The similarity between the embedded input and the class serves as a representation of the likelihood that the instance belongs to a specific class. The feature extractor  $f_\theta: X \rightarrow R^d$ , specified as an input, maps the input to a feature vector  $f_\theta(x)$  of dimension  $d$ .

The Federated Emotion Recognition algorithm is based on a query suggestion model which was first used with the Google Keyboard. Each target subject from the dataset served as a representative of one user for our purposes. The fundamental flaw in this situation is that, instead of learning models centrally, models can be learned locally by averaging their parameters across clients, with the resulting parameters performing similarly to models learned centrally. In the suggested method, Algorithm 1 handles Federated Personalization, in which only the weights of the convolutional layers (not the feature maps) are directed to the central server aimed at averaging. The size of the local minibatch is  $B$ , the count of local epochs is  $E$ , the count of rounds that the local model is fine-tuned is  $F$ , and the learning rate is  $\eta$ . The clients are indexed by  $a$ . In the Client parameter tuning procedure, the learning rate is lowered. The model can learn common facial traits from a huge population by using the FL optimization technique on the lower convolutional layers. However, a deep network's last upper layers are primarily instructive for the final prediction task (seven emotions). It would still be very challenging to reconstruct the high-dimensional face images, given that the CNN feature maps are not shared during federated learning in our approach. Only the emotional expressions of the target subject are learned as subtle facial expressions of emotion, without necessarily learning from the entire population. The facial expressions may only generalize well for that subject, but not for the entire population. All clients initially receive a model, whose parameters are initialized with pre-trained parameters of a model that was centrally trained on the same task with a different subset of data, without personalization. The algorithm then follows the steps of the Federated Emotion Recognition algorithm. The Client Process procedure

concludes, and only the weights from the convolutional layers  $w_g$  are sent to the central server to find the average. The weights of the last fully-connected layers are  $w_l$ . The data remains locally with the client. As a result, only the averaged convolutional weights, denoted as  $W_{g_{t+1}}$  are used for each local client model update.

The images/shots of the individuals are protected using the proposed PSFL approach. Then, the procedure of Client parameter tuning engages in local fine-tuning. The convolutional layers  $w_g$  are frozen for each client to fine-tune and reduce the optimizer's learning rate by a factor for the respective client to prevent overrunning. The slower learning rate ensures that the local layers gradually "reconnect" to the globally averaged convolutional layers from which they were previously separated. The local models are then trained across several epochs, resulting in personalized models as a result of the Client fine-tuning stage, which only trains each client's dense layers on locally accessible data.

In FL, it is assumed that the dataset is distributed across  $a = 1, \dots, A$  clients rather than being centrally kept. The notation that is employed is as follows:

- A given client holds a set of indices ( $P_a$ ) of data points that represent each division.
- Here,  $n_a$  denotes the number of data points that the client currently holds, where  $n$  indicates the total count of data points acquired by all clients and  $n_a = |P_a|$ .

In such a federated scenario, assuming mini-batch stochastic gradient descent, the computational work for one complete update is governed by three parameters.

The formula for minimizing a loss function is as follows:

$$\min_{\theta \in R^d} f(\theta), \text{ where } f(\theta) \stackrel{\text{ghi}}{=} \sum_{i=1}^n f_i(\theta) \quad (6)$$

Then, each client receives a step of gradient descent and apprises its parameters, accordingly, dignified as

$$f(\theta) = \sum_{a=1}^A \frac{n_a}{n} F_a(\theta) \quad (7)$$

where;

$$F_a(\theta) = \frac{1}{n_a} \sum_{i \in P_a} f_i(\theta) \quad (8)$$

Thus, instead of computing loss (i.e., the binary cross-entropy) as an average over  $n$  number of samples taken from a centralized data set as:

$$\frac{1}{n} \sum_{i=1}^n f_i(\theta) \quad (9)$$

Calculate the average loss  $F_a(\theta)$  for a stated client as

$$\frac{1}{n_a} \sum_{i \in \rho_a} f_i(\theta) \quad (10)$$

And then collect the loss of all the clients A by calculating a weighted average loss based on the count of labeled data points  $n_a$  that each client possesses. Similar to the process of calculating the loss, the gradients of the model are calculated. In a federated location, each client calculates the average gradient  $g_a$  on its data locally as:

$$g_a = \nabla F_a(\theta) \quad (11)$$

$$v_a, \theta_a \leftarrow \theta \circ g_a \quad (12)$$

The weighted average of these gradients, which is comparable to the weighted average of the loss above, is then computed by a central server after this step is done numerous times, that is, for numerous epochs E.

$$\theta \leftarrow \sum_{a=1}^A \frac{n_a}{n} \theta_a \quad (13)$$

$f_i(\theta)$  is a loss function that may be reformulated to represent A clients in a federated environment, where  $F(\theta)$  represents the loss for a prediction of one observation  $(x_i, y_i)$  given model parameters.

- The percentage C of clients A that take part in a particular update round;
- The quantity of gradient descent epochs for every client;
- The size B of every batch is applied to every client update. Here, C affects the amount of processing power needed at the server.

More participating clients necessitate a greater data transfer to E and B puts greater effort into gathering information from the server. This impacts the amount of computation needed on the client side. A face extractor parametrized by  $\theta_0$  has already been trained on a publicly accessible dataset that is easily accessible at the server. The server can use certain datasets in the public domain to pre-train the system. A FER system  $f_{\theta_0}$  trained on publicly available datasets is present on the server. One identity's  $n_i$  instances are contained in the  $i^{th}$  client node (the owner of the device). As a result, the dataset is accessible to the  $i^{th}$  client. The performance of the face recognition system can be enhanced, by jointly learning the data on each client without transferring the training images outside the device. The discussed algorithms are given in Figure 12.6.

**Procedure Server-Emotion recognition**Set  $w_{g0}$ Set  $w_{l0}$ The server resets a fan-beam-based face feature extractor with the parameter  $\theta_0$  ;For each repetition  $t=1,2\dots T$  $m \leftarrow \max(C \times K, 1)$  $S_t \leftarrow (m \text{ clients chosen randomly})$ For every client  $p \in S_t$  which are parallel $W_{gt+1}^p \leftarrow ClientProcess(W_{pt}, W_{lt}^p)$ 

The server collects all the clients' altered parameters.

 $W_{pt+1} \leftarrow \sum_{p=1}^N \frac{n_p}{n} W_{gt+1}^p$ Where  $n$  is the total samples gathered from all of the clients.For each client  $p \in S_t$  in parallel doClientParameterTuning ( $W_{gt+1}, W_{lt+1}^p$ ) $\theta_{t+1} = \{W_{gt+1} \cup W_{lt+1}^{p=1\dots K}\}$  used for inference in this repetition

The spreadout regularizer is the server's final step in unscrambling the class embeddings from one another.

 $W_{t+1} \leftarrow W_{t+1} - \lambda W_{t+1} regsp(W_{t+1})$ **Procedure ClientProcess ( $w_g, w_l$ ) Runs on client  $p$**  $B \leftarrow (Gather P_p \text{ into groups of size } B)$ For every epoch  $i$  ranging from 1 to  $E$  continue $\{W_g \cup W_l\} \leftarrow \{W_g \cup W_l\} - \alpha \nabla l(\{W_g \cup W_l\}; b)$ The client initializes the  $W_0 = \frac{1}{n_i} \sum_{j=1}^{n_i} f_{\theta_0}(x_j)$ 

Modify all parameters which are send to server

**Procedure ClientParameterTuning ( $w_g, w_l$ ) Runs on client  $p$**  $B \leftarrow (Gather P_p \text{ into groups of size } B)$ For every epoch  $i$  ranging from 1 to  $E$  do $W_l \leftarrow W_l - \alpha \nabla l(W_l, b)$ **FIGURE 12.6** The algorithms employed.

The spread out regularizer, which is used as a final step, ensures that the class embeddings are evenly distributed throughout the embedding space and do not cluster in a single point. The spread out regularizer is optimized by the server after collecting the class embeddings from all clients.

$$regsp(W_t) = \sum_{c \in C} \sum_{\hat{c} \neq c} (\max\{0, v - d(w_t^c, w_t^{\hat{c}})\})^2 \quad (14)$$

where;  $v$  is the margin.

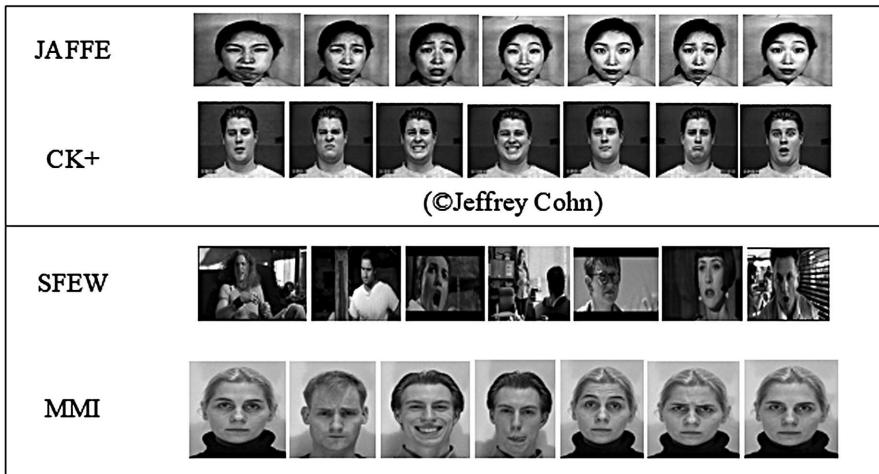
## 12.4 DATASETS

The investigations utilize six datasets: JAFFE [20], Cohn Kanade (CK+) [21, 22], Multimedia Understanding Group (MUG) [23], Static Facial Expressions

in the Wild (SFEW) [24–26], Oulu-CASIA NIR and VIS facial expression database (OULU-CASIA) [27], and Man Machine Interface (MMI) [28–30]. The JAFFE collection consists of 213 facial expression photos from 10 individuals. The photos have a size of  $256 \times 256$  and come with corresponding labels. The images in this dataset are categorized into seven groups based on emotions. The CK+ dataset contains two versions: CK and CK+. Compared to CK, CK+ offers more emotive facial representations. The facial expression photos in CK+ have a size of  $640 \times 490$ . This dataset includes 593 accessible image sequences, extracted from 123 individuals. Only the most expressive pictures are used in the experiments. The initial offering in each series is the neutral emotion. The experiments utilize 1281 photos. In the MUG dataset, there are specific image sequences where the first image is the neutral emotion image. Each sequence consists of 60 photos. The facial expression photos are depicted in  $896 \times 896$  pixels. Each sequence contains a total of 1462 photos, taken from 86 individuals. The experimental analysis uses 81 images from each category. The photographs were taken in various environmental settings, with a resolution of  $720 \times 576$  in an unrestricted environment. The Oulu-CASIA project captured images of 80 individuals (Oulu, the Machine Vision Group, Chinese Academy of Science, and Institute of Automation). For the experiments, five to six of the most emotive photographs were selected from each of the 480 sequences. 500 photos were chosen from each class. A total of 30 individuals contributed to the 312 series of photographs. From the series of images, three to four photographs with exceptional expressions were selected. The final picture in each series portrays a neutral emotion. The experiments conducted are independent of subjects. Sample images are provided in Figure 12.7.

## 12.5 EXPERIMENTAL EVALUATION

In essence, facial emotion recognition algorithms are embedded classifiers, where the input instance and the classes (identities) are both embedded in a high-dimensional Euclidean space. The chance of the instance belonging to a given class is represented by the resemblance between the embedded input and the class. Formally, the face feature extractor  $f_\theta: X \rightarrow R^d$  sets the input image as a d-dimensional feature vector  $f$  given an input face picture  $f_\theta(x)$ . Using the TensorFlow library, all the photos from the dataset are used to train the model on the training images without utilizing the labels in both centralized learning and federated learning settings.



**FIGURE 12.7** Sample images from the dataset used.

Source: Adapted from Ref. [20-26]

The publicly accessible face matcher CosFace [31] (64 layers) generates a feature vector of length 512 for the face feature extractor in the application of face recognition. This work employs an angular margin SoftMax loss with scale and margin hyper parameter values of 30 and 10, respectively, to pretrain the server-side global face recognition model. A squared hinge loss is utilized with cosine distance, and a boundary of 0.9 is set as the positive loss function for training in the FL configuration for the respective client. Both the class embeddings and the instance embeddings are normalized to have a norm of 1. There are 200 communication rounds to train the model. It updates the model on each of the servers with a learning rate of 0.001.

### 12.5.1 EFFECT OF NUMBER OF CLIENTS

This work divides our training set of 10,000 subjects from CASIA among multiple clients. Each client is provided with an equivalent count of subjects for training, allowing us to assess the impact and performance of the traditional algorithm used by the clients participating in cooperative learning. The traditional method of developing facial expression classification models involves having all the data available centrally, representing a single client scenario. We conduct tests on different numbers of client nodes and analyze the results. It is important to note that the existing algorithms do not require class embeddings to be properly segregated, as they train clients

without using the spread-out regularizer with the AM-SoftMax loss function. However, this approach leads to a simplistic solution when dealing with 10,000 clients, resulting in feature vectors clustering around a single point and significantly degrading the results. To overcome this problem, our proposed approach involves the server collecting class embeddings from all clients and optimizing the spread-out regularizer. Additionally, the features are normalized using hypersphere-based normalization.

### **12.5.2 BASELINE ADJUSTMENT IN A NON-FEDERATED WAY**

The centralized collection of faces from the 1,000 clients and non-federated use of them is analyzed for performance in the FER algorithm. As the face photographs are shared with the server, there are serious privacy concerns with the standard approach of building DNN-based recognition models.

### **12.5.3 RANDOMLY INITIALIZED CLASS EMBEDDINGS**

All class embeddings and face feature vectors may condense into a single point in the embedding space as a result of training using a positive loss function. One method of addressing this is by randomly initializing the class embeddings and correcting them during training. Although it is not technically required, doing so prevents the class embeddings from merging into a single point. This can enhance the effectiveness of the pre-trained face recognition system by collaboratively learning from supplementary face images accessible to each unique client. The privacy of the face photographs visible to each client was maintained while bridging the gap between the non-federated configuration and the federated setup of learning.

FedAffect [32] is compared to the aforementioned methods in Table 12.1. FedAffect utilizes the Fed Avg method to globally improve performance. By jointly learning from additional face images available for each individual customer, FedAffect can enhance the performance of the pre-trained FER system. The privacy of the face photographs visible to each client is maintained while bridging the gap between the non-federated configuration and the federated system of learning. Figure 12.8 illustrates the impact of the FedAffect algorithm's client count. During training, the subjects in CASIA are evenly distributed among multiple client nodes. Each client represents one identity and a client node represents the traditional or non-federated method of training AFR systems.

**TABLE 12.1** Performance Comparison (Accuracy in Percentage)

<b>Method</b>	<b>Training Data</b>	<b>JAFFE</b>	<b>CK</b>	<b>SFEW</b>	<b>MMI</b>	<b>MUG</b>	<b>OULU-CASIA</b>
E-Gabor [19]	Aggregated centrally.	98.8	99.0	78.2	89.9	90.2	93.2
Non-federated method of fine-tuning.	Aggregated centrally.	98.2	99.2	79.2	90.4	92.2	93.2
Class embedding with random initialization.	Distributed	66.78	72.23	56.73	75.36	75.83	74.32
FedAffect [32]	Decentralized private data	66.89	78.23	58.23	78.56	79.89	76.34
Proposed technique	Distributed	98.3	99.8	75.3	93.5	95.6	93.0

#### 12.5.4 ABLATION RESEARCH

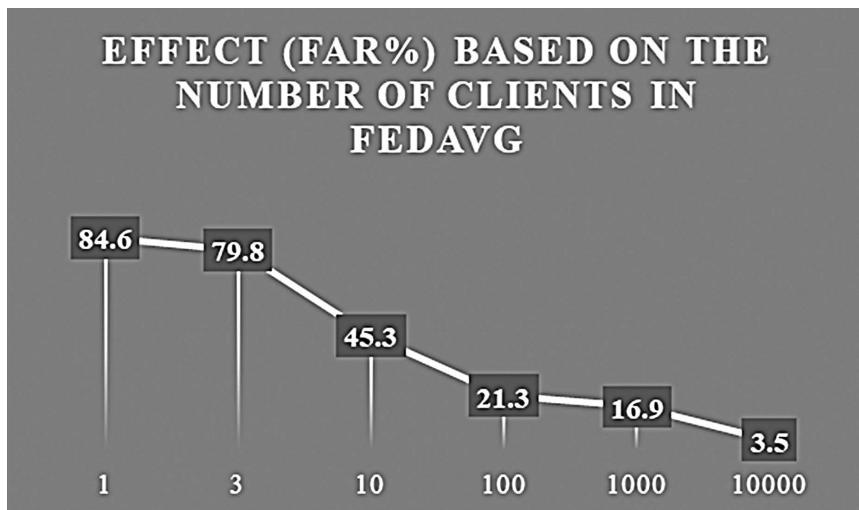
To assess the impact of each client on performance, we conducted ablation experiments on the spread-out regularizer and the mean feature initialization technique. The results in Table 10.2 offer an overview of the ablation trial. While mean feature initialization does not significantly enhance overall performance, it does expedite the process. Therefore, in a cooperative learning setting where there is a communication delay between the server and clients, it is crucial to address this bottleneck.

**TABLE 12.2** Analysis of the Proposed Method (Accuracy in Percentage)

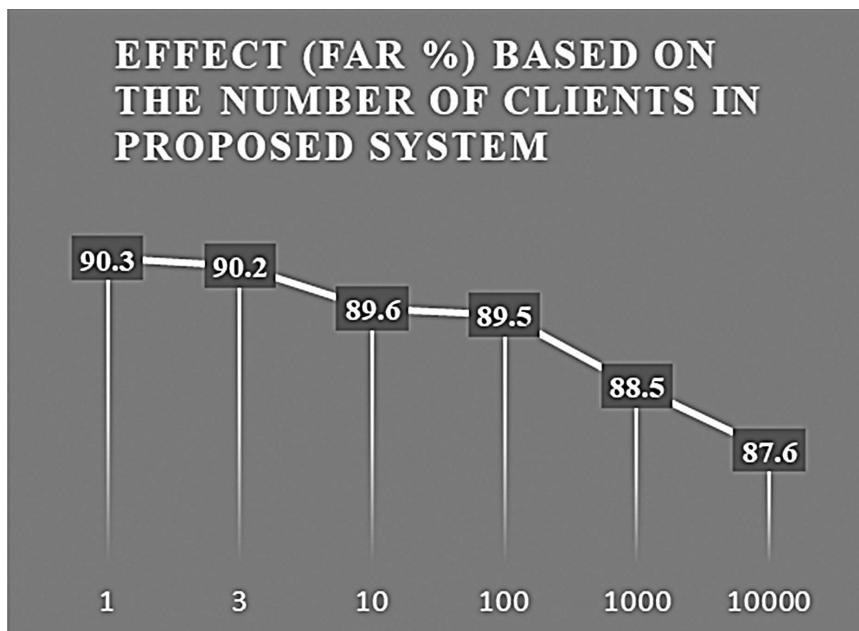
<b>Method</b>	<b>JAFFE</b>	<b>CK</b>	<b>SFEW</b>	<b>MMI</b>	<b>MUG</b>	<b>OULU-CASIA</b>
Without mean feature initialization.	95.34	95.3	63.4	70.8	88.5	85.6
Without spread out regularizer.	79.45	89.0	56.7	64.5	68.9	70.9

For global aggregation, the FedAvg [33] algorithm is considered. The FedAvg trains on the AM-SoftMax loss function. Without the widespread regularizer, clients are not enforced, and well-separated class embeddings are required. Consequently, in the instance of 10,000 clients, FedAvg provides a simple solution. Here, the feature vectors for each face image are grouped, impairing performance. This problem is overcome in the proposed approach, where the server totals the class embeddings of all the clients and

then optimizes the spread-out regularizer. This results in better performance, even for a larger number of clients as shown in Figures 12.8 and 12.9.



**FIGURE 12.8** Analysis of FAR based on the number of clients in FedAvg.



**FIGURE 12.9** Analysis of FAR based on the number of clients in the proposed system.

## 12.6 CONCLUSION

In this article, we address the challenge of developing a cooperative facial emotion recognition system from existing face photos. These applications are widely used in our daily lives on various mobile devices, with privacy being a key consideration. We propose a novel approach using a federated learning framework to enhance the effectiveness of an already trained facial recognition process. The suggested method utilizes a unique fan-based feature extraction algorithm. In this approach, the server employs a global face feature extractor, and the client uses the mean feature to initialize the class embedding. Except for the initial round where the client initializes itself, the server provides the client with the current global settings. Based on the local data, the client modifies the model's parameters. The client then updates the parameters and sends them back to the server. The server gathers all of the clients' modified parameters. To isolate the class embeddings from one another, the server uses the spread-out regularizer. After T communication cycles of the aforementioned procedures, the global parameter for the feature extractor is returned. This suggested strategy can improve the results by leveraging the extra data available on mobile devices, while ensuring that the photos/shots are always protected and remain on the device.

## KEYWORDS

- **ablation research**
- **AM-SoftMax loss function**
- **cooperative learning**
- **facial emotion recognition system**
- **FedAffect algorithm**
- **mean feature initialization**
- **regularizer**

## REFERENCES

1. Yang, Q., Liu, Y., Cheng, Y., Kang, Y., Chen, T., & Yu, H., (2019). Federated learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 13(3), 1–207.

2. Ning, L., Singhal, K., Zhou, E. X., & Prakash, S., (2021). *Learning Federated Representations and Recommendations with Limited Negatives*. arXiv preprint arXiv:2108.07931.
3. Yang, Q., Liu, Y., Chen, T., & Tong, Y., (2019). Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(2), 1–9.
4. Thapa, C., Arachchige, P. C., Camtepe, S., & Sun, L., (2022). Split Fed: When federated learning meets split learning. In: *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 36, No. 8, pp. 8485–8493).
5. Long, G., Tan, Y., Jiang, J., & Zhang, C., (2020). Federated learning for open banking. In: *Federated Learning* (pp. 240–254). Springer, Cham.
6. Zerka, F., Barakat, S., Walsh, S., Bogowicz, M., Leijenaar, R. T., Jochems, A., Miraglio, B., et al., (2020). A systematic review of privacy-preserving distributed machine learning from federated databases in health care. *JCO Clinical Cancer Informatics*, 4, 184–200.
7. Shyu, C. R., Putra, K. T., Chen, H. C., Tsai, Y. Y., Hossain, K. T., Jiang, W., & Shae, Z. Y., (2021). A systematic review of federated learning in the healthcare area: From the perspective of data properties and applications. *Applied Sciences*, 11(23), 11191.
8. Kassis, G. A., Makowski, M. R., Rückert, D., & Braren, R. F., (2020). Secure, privacy-preserving, and federated machine learning in medical imaging. *Nature Machine Intelligence*, 2(6), 305–311.
9. Sarma, K. V., Harmon, S., Sanford, T., Roth, H. R., Xu, Z., Tetreault, J., Xu, D., et al., (2021). Federated learning improves site performance in multicenter deep learning without data sharing. *Journal of the American Medical Informatics Association*, 28(6), 1259–1264.
10. Rudovic, O., Tobis, N., Kaltwang, S., Schuller, B., Rueckert, D., Cohn, J. F., & Picard, R. W., (2021). *Personalized Federated Deep Learning for Pain Estimation from Face Images*. arXiv preprint arXiv:2101.04800.
11. Ding, J., Tramel, E., Sahu, A. K., Wu, S., Avestimehr, S., & Zhang, T., (2022). Federated learning challenges and opportunities: An outlook. In: *ICASSP 2022–2022 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)* (pp. 8752–8756). IEEE.
12. Yang, Z., Chen, M., Wong, K. K., Poor, H. V., & Cui, S., (2021). Federated learning for 6G: Applications, challenges, and opportunities. *Engineering*.
13. Li, T., Sahu, A. K., Talwalkar, A., & Smith, V., (2020). Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3), 50–60.
14. Khan, L. U., Saad, W., Han, Z., Hossain, E., & Hong, C. S., (2021). Federated learning for the internet of things: Recent advances, taxonomy, and open challenges. *IEEE Communications Surveys & Tutorials*.
15. Zhuang, T., Leng, S., Nett, B. E., & Chen, G. H., (2004). Fan-beam and cone-beam image reconstruction via filtering the back projection image of differentiated projection data. *Physics in Medicine & Biology*, 49(24), 5489.
16. Vedaldi, A., & Lenc, K., (2015). MatConvNet: Convolutional neural networks for MATLAB. In: *Proceedings of the 23<sup>rd</sup> ACM International Conference on Multimedia* (pp. 689–692).
17. Aggarwal, D., Zhou, J., & Jain, A. K., (2021). FedFace: Collaborative learning of face recognition model. In: *2021 IEEE International Joint Conference on Biometrics (IJCB)* (pp. 1–8). IEEE.
18. Asthana, A., Zafeiriou, Cheng, S., & Pantic, M., (2014). Incremental face alignment in the wild. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 1859–1866).

19. Alphonse, A. S., & Dharma, D., (2017). Enhanced Gabor (E-Gabor), hypersphere-based normalization, and Pearson general kernel-based discriminant analysis for dimension reduction and classification of facial emotions. *Expert Systems with Applications*, 90, 127–145.
20. Lyons, M., Akamatsu, S., Kamachi, M., & Gyoba, J., (1998). Coding facial expressions with Gabor wavelets. *Proceedings of the Third IEEE International Conference on Automatic Face and Gesture Recognition* (pp. 200–205).
21. Kanade, T., Cohn, J. F., & Tian, Y., (2000). Comprehensive database for facial expression analysis. *Proceedings of the Fourth IEEE International Conference in Automatic Face and Gesture Recognition* (pp. 46–53).
22. Lucey, P., Cohn, J. F., Kanade, T., Saragih, J., Ambadar, Z., & Matthews, I., (2010). The extended Cohn-Kanade dataset (ck+): A complete dataset for action unit and emotion-specified expression. *proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*(pp. 94–101).
23. Aifanti, N., Papachristou, C., & Delopoulos, A., (2010). The MUG facial expression database. *Proceedings of the Eleventh International Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS)* (pp. 1–4).
24. Dhall, A., Asthana, A., Goecke, R., & Gedeon, T., (2011). Emotion recognition using PHOG and LPQ features. *Proceedings of the IEEE International Conference on Automatic Face & Gesture Recognition* (pp. 878–883).
25. Dhall, A., Goecke, R., Joshi, J., Sikka, K., & Gedeon, T., (2014). Emotion recognition in the wild challenge: Baseline, data, and protocol. *ACM ICMI*.
26. Dhall, A., Goecke, R., Lucey, S., & Gedeon, T., (2012). Collecting large, richly annotated facial expression databases from movies. *IEEE Multimedia* (Vol. 19, pp. 34–41).
27. Zhao, G., Huang, X., Taini, M., Li, S. Z., & Pietikäinen, M., (2011). Facial expression recognition from near-infrared videos. *Image and Vision Computing*, 29(9), 607–619.
28. Pantic, M., Valstar, M., Rademaker, R., & Maat, L., (2005). Web-based database for facial expression analysis. *Proceedings of the IEEE International Conference on Multimedia and Expo* (pp. 1–5).
29. Valstar, M., & Pantic, M., (2010). Induced disgust, happiness, and surprise: An addition to the MMI facial expression database. *Proceedings of the 3<sup>rd</sup> Intern. Workshop on Emotion (satellite of LREC): Corpora for Research on Emotion and Affect* (pp. 65–90).
30. Valstar, M. F., & Pantic, M., (2012). Fully automatic recognition of the temporal phases of facial actions. *IEEE Transactions on Systems, Man, and Cybernetics*, 42(1), 28–43.
31. Wang, H., Wang, Y., Zhou, Z., Ji, X., Gong, D., Zhou, J., Li, Z., & Liu, W., (2018). CosFace: Large margin cosine loss for deep face recognition. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 5265–5274).
32. Shome, D., & Kar, T., (2021). FedAffect: Few-shot federated learning for facial expression recognition. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision* (pp. 4168–4175).
33. Qu, Z., Lin, K., Kalagnanam, J., Li, Z., Zhou, J., & Zhou, Z., (2020). *Federated Learning's Blessing: FEDAVG Has Linear Speedup*. arXiv preprint arXiv:2007.05690.

# Index

---

## A

- Ablation  
research, 316, 318  
trial, 316
- Access control list (ACL), 257
- Account takeover (ATO), 17
- Adversarial, 96, 131, 132, 134–141, 143, 145, 148–150, 187, 189, 193, 194, 205, 213, 215, 220–223, 225–228, 230–233, 260, 276  
attack, 131, 134, 137, 138, 145, 148–150, 193, 215, 220–222, 225, 233  
intent, 136, 220, 222, 223, 225, 226  
network, 140, 205, 260  
scenario detection, 134
- Aggregation, 12, 24–26, 30, 31, 34, 36–38, 41–45, 49, 51, 55, 57–59, 67, 69, 71, 78, 81–83, 87–89, 92, 95–101, 140, 145, 146, 148, 154, 157, 158, 160, 165, 167, 169–171, 175, 197, 200, 202, 204, 215–221, 223, 224, 233, 241, 251, 254, 258, 261, 273, 275, 316  
algorithm, 57, 58, 219  
process, 36–38, 41, 158, 273  
protocols, 24, 98, 148, 175  
technique, 24, 25, 36, 41, 43, 45
- Algorithm, 3, 4, 7, 8, 18, 24, 32, 37–40, 45, 55, 57–59, 65, 68, 71, 89, 91, 98, 109, 118, 123, 134, 143, 160, 165, 167, 169, 170, 172–175, 177, 183, 185, 189, 192, 195, 196, 198, 203, 205, 214, 219, 220, 224, 228, 229, 231, 238, 241, 246, 248, 252–255, 262, 271, 272, 284, 288, 291, 298, 299, 302, 303, 309, 311–316, 318
- All model averaging (AMA), 24, 37, 38, 40, 45  
technique, 37, 40
- Allergic reactions, 122
- Amazon, 53, 59, 186
- AM-softmax loss function, 315, 316, 318
- Analog aggregation (AGA), 171
- Anomalies, 132, 134, 135, 137, 142, 147, 227
- Anomaly detection, 99, 102, 132, 133, 135, 137, 138, 143, 145  
information systems, 134  
model, 145
- Antibodies, 121
- Aphasia, 120
- Application  
intelligence, 60  
specific integrated circuits (ASICs), 7
- Artificial  
intelligence (AI), 2, 3, 6–9, 18, 23–25, 48, 64, 107, 109, 112, 117, 118, 126, 128, 132, 134, 136, 154, 183, 194, 205, 213, 214, 216, 227, 238, 248, 271, 291, 297, 298  
aided systems, 16  
applications, 67, 126  
based systems, 6  
chip, 48  
methodologies, 9, 126  
research, 48  
systems, 6  
XAI approaches, 126  
neural networks, 137
- Asynchronous aggregation, 175
- Auction, 90, 278, 279, 282–284, 286, 291  
theory, 90, 282, 284
- Automata theory, 2
- Automated infrastructure, 16
- Automation, 15, 52, 116  
solutions, 15
- Automotive applications, 18
- Autonomous  
cars, 288  
data, 23, 32  
vehicles, 68, 112, 288

## B

- Backdoor attacks, 205, 213, 224, 227, 233
- Backend network, 43

- Bandwidth, 26, 47, 48, 52, 55, 58–61, 109, 111, 133, 138–140, 143, 147, 153, 155, 157, 158, 169, 172–175, 220, 228, 239, 278
- Batch normalization (BN), 162, 305
- Benign, 135–137, 145, 148, 204, 224
- Best model averaging (BMA), 24, 37, 39, 40, 45
- Bias, 32, 57, 223, 231, 232, 301
- Bidder, 90, 91, 283, 284
- Binary, 4, 305, 310
- Biomedical devices, 118
- Biosensor, 121
- Bisection, 173
- Bitcoin, 240, 244, 246, 252, 260
- Blockchain, 18, 43, 45, 93, 96, 97, 146, 230–233, 237–240, 244–255, 257–262, 264, 272, 279, 284, 289
- aggregation, 24, 43, 45
- and fog-based architecture network (BFAN), 257
- based federated learning (BCFL), 251
- without aggregator (BAFFLE), 258
- classification, 252
- implementations, 252
- ledger, 43
- research, 240, 252
- technology, 97, 238, 240, 244, 245, 251, 252, 254, 255, 258, 260, 261, 284
- Bluetooth, 109
- Body guardian remote monitoring system, 116
- Bottleneck, 10, 16, 95, 135, 140, 146, 153, 155, 163, 171, 175–177, 214, 289, 297, 298, 302, 303, 316
- Breaching, 18, 221, 232
- Broadband analog aggregation (BAA), 170
- Byzantine attack, 204
- C**
- Caching data, 107
- California consumer privacy act (CCPA), 184, 188
- Carlini & Wagner attack (C&W), 193
- Central
- edge server, 68
  - server, 9, 23, 26, 27, 32–42, 49, 52–55, 57–59, 63, 64, 69, 70, 78, 80–82, 86, 88–90, 95, 96, 108, 133, 140, 154, 162, 197, 205, 216, 217, 221, 231, 239, 241, 249, 250, 271–275, 278, 280, 281, 284, 288–290, 300, 309–311
- Centralized
- and federated training, 63
  - architecture, 20, 70, 131, 142, 221, 226
  - data, 9, 78, 107, 108, 154, 259, 301, 310
  - federated learning (FL), 54
  - learning paradigms, 221
  - machine learning (ML), 9, 24, 25, 45, 52, 53, 58, 133, 221, 271, 273, 302
  - mechanism, 215
  - paradigm, 59
  - server, 31, 47, 49, 52, 55, 58, 63, 67, 196, 197, 237, 239, 243
- Channel state information details (CSI), 173
- Chatbot, 114
- Chronic
- disease, 126
  - management, 123
  - manifestations, 119
  - prophylaxis, 122
  - illnesses, 118, 122
- Cisco, 123
- Claims, 17, 18, 93, 170
- Classical
- aggregation techniques, 45
  - network architecture, 80
- Client
- device, 9–11, 19, 57, 71, 77–79, 81, 82, 86–92, 96, 98–101, 133, 143, 147, 153–155, 164–167, 169, 171–175, 177, 216, 217, 220, 228, 230, 231, 273, 276, 278, 281
  - edge communication, 58
  - instrumentation, 13
  - model, 26, 37–40, 45, 169, 226, 303
  - process procedure, 309
  - selection, 11, 12, 54, 97, 99–101, 147, 157, 158, 170, 172, 173
  - server
  - exchanges, 54
  - system, 1, 140
  - side parallel model, 49
  - training, 12
- Cloud
- computing, 47–49, 59–61, 78, 138, 255, 261, 262, 264

- data, 63, 254  
edge infrastructure, 67  
platform, 61  
resources, 60  
server, 47, 48, 58, 61, 63, 67, 69, 93, 109, 202, 203  
storage, 64, 120, 138, 254, 255  
Clustering, 5, 62, 134, 160, 168, 185, 189, 232, 274, 315  
approaches, 5, 232  
Clusters, 5, 65, 136, 168, 255  
Collaborative learning, 8, 66, 132, 183, 189, 194, 237, 272, 273, 279, 287, 288, 290, 291  
Collisions, 67  
Communication  
bottlenecks, 153  
budget, 158  
efficiency, 97, 99, 153, 155, 156, 159, 161, 163, 167, 176, 177, 278  
scenarios, 156, 169  
strategies, 176  
time, 169, 174  
Compressed  
analog distributed sgd (CADSGD), 171  
federated distillation (CFD), 164  
Compression sensing (CS), 159  
Computation, 36, 62, 64, 65, 67, 79, 93, 94, 96, 98, 101, 131, 145, 154, 161, 163, 167, 171–174, 202, 205, 229, 240, 241, 245, 254, 255, 258, 259, 274, 276, 282, 285, 287, 289, 311  
protocols, 65  
Computational  
ability, 67  
power, 7, 53, 126, 138, 172, 216, 240, 279  
resources, 7, 100, 175, 220, 273, 278  
Consensus  
node, 43  
protocol, 43  
Consortium blockchains, 246  
Containerization technology, 70  
Contract theory, 92, 279, 282, 285, 286, 291  
Contractual agreements, 92  
Convolutional  
layers, 303, 305, 309, 310  
neural network (CNN), 33, 34, 298, 305, 309  
Cooperative  
games, 280  
learning, 272, 277, 289, 314, 316, 318  
model, 91  
Coronavirus disease (COVID-19), 16, 66, 115, 217, 262  
Credit risk management, 56  
Cross  
clinical validation, 125  
device  
federated learning, 286  
setting, 273, 287, 292  
systems, 54  
silo  
federated learning (FL), 10, 11, 53, 139, 197, 285, 289  
setting, 289, 292  
systems, 55  
Cryptographic  
devices, 123  
elements, 253  
hash, 43  
protocols, 116  
techniques, 229  
Cryptography, 42, 94, 189, 237, 249  
Cybernetics, 2  
Cyber-physical systems (CPS), 137, 138
- ## D
- Data  
analysis, 4, 66, 123, 127, 136, 260  
analytics, 63, 122, 271  
anonymization, 183, 190, 191, 201  
breach, 17, 185–187, 250  
center, 49, 53, 108, 184, 254  
collection, 17, 107, 108, 119, 122, 123, 127, 140, 184, 190, 191  
resources, 71  
compression, 102, 153, 161, 174, 228  
distribution, 13, 14, 17, 84, 94, 98, 133, 142, 147, 149, 165, 166, 172, 214, 220, 226, 231, 274, 301  
flow diagram (DFD), 190  
generator devices, 47  
heterogeneity, 9, 14, 154, 155, 165, 166  
life cycle, 189, 190  
localization, 19, 47, 140  
mining, 63, 185, 271

- modification, 204, 206
- optimization, 60
- poisoning, 148, 204, 206, 221
- privacy, 9, 19, 53, 82, 133, 154, 185, 186, 189, 196–198, 201, 213, 214, 216, 238, 260, 297, 298, 301
- policies, 18
- processing, 2, 214
- protection, 17, 188, 195, 196, 206, 258, 297, 298
- redundancy, 5
- resources, 67
- scarcity, 142, 143
- security, 60, 79, 93, 108, 119, 126, 286, 298, 299
- silo, 10, 81, 154, 237
- sources, 48, 61, 132, 192, 278
- storage, 10, 60, 78, 80, 101, 139, 190, 191, 213, 214, 230, 239, 257, 260
- transfer, 9, 17, 133, 140, 145, 154, 163, 165, 168, 171, 213, 216, 222, 228, 238, 250, 251, 311
- cost, 17, 238
- transmission, 60, 64, 123, 156, 248
- utility optimization, 187, 206
- Databases, 15, 49, 83, 202, 304
- Databox platform, 69
- Decentralized, 10, 43, 48, 49, 54, 55, 57, 59, 96, 97, 108, 132, 139, 140, 146, 183, 184, 198, 205, 213, 217, 218, 221, 226, 228, 239–241, 246, 247, 249, 251, 254, 258–260, 278, 297, 298
- aggregation, 59
- data, 10, 54, 132
- federated learning (FL), 55
- Decision-making, 18, 24, 26, 52, 61, 64, 69, 108, 110, 123, 248, 280
- Decrypts, 30, 41, 44
- Deep
  - architectures, 62
  - fool attacks, 193
  - learning (DL), 2, 6, 25, 47, 48, 61–65, 71, 108, 131–133, 137, 140, 143, 154, 155, 216, 258, 261, 274, 278, 288, 290, 291, 297, 298, 303
  - frameworks, 65
  - models, 48, 63, 132, 140, 154, 291
  - neural network, 62
- Defensive strategies, 227, 228, 233
- Demographic prediction, 63
- Denial of service (DoS), 144, 190
- Density-based approaches, 5
- Deployment, 12, 13, 16, 48, 61, 64, 140, 289
- Deterministic quantization framework, 158
- Device
  - heterogeneity, 10, 154
  - homogeneity, 59
  - mobility, 19, 147
- Diagnosis, 66, 117, 118, 124, 127
- Diagnostic, 6, 15, 115, 121, 123, 128
  - applications, 6
  - equipment, 66
- Differential privacy (DP), 65, 93–96, 149, 165, 167, 183–185, 194, 195, 198, 201–206, 229, 233, 254, 259, 260
- Diffie-Hellman (DH), 201
- Dimensions, 5, 91, 94, 229
- Distillation, 153, 155, 156, 164, 166, 194
- Distributed
  - denial of service (DDoS), 143, 144, 150
  - learning, 1, 19, 23–25, 101, 132, 153, 159, 213–216, 291
  - machine learning (DML), 8, 9, 24, 25, 45, 48, 77, 81, 187, 189, 196, 197, 206, 249, 271, 274
  - mean estimation (DEM), 157, 158
  - systems, 8, 49, 143, 214
- Diverse business sectors, 49
- Drug management, 121, 124
- Dutch auction, 284
- Dynamic, 125, 301
  - environments, 148
  - pay-off, 91
- E**
- Edge/Fog (E/F), 47, 66, 237, 262
  - computing, 47, 48, 66, 71
  - environments, 47
  - framework, 47
  - paradigm, 47
  - systems, 47, 71
- Earthquake prediction, 63
- E-commerce, 18
- Ecosystem, 109, 123, 258, 261, 278, 279, 287, 288, 290
- Edge
  - cloud, 58
  - communication, 58

- computing, 9, 48, 53, 59–62, 64, 66, 67, 71, 89, 91, 101, 140, 141, 159, 214, 249–251, 260, 264, 273, 274  
devices, 48, 91  
network, 71  
technologies, 9
- device, 52, 57, 60, 61, 63, 68–70, 93, 118, 144, 153, 176, 183, 202, 205, 238, 260, 273, 274, 276, 284, 285, 288
- learning system, 68
- node, 60, 66, 253
- server, 57, 58, 68, 259
- Electroencephalography, 69
- Electronic health  
information (EHR), 117, 261, 300  
record (EHR), 66, 117, 120, 126, 261, 300
- Embedded systems, 118, 137
- Embedding, 5, 121, 170, 299, 303, 305, 306, 309, 312, 315, 316, 318
- E-medical services, 126
- Encoders, 123, 140
- Encrypted  
computation, 65  
models, 30, 41
- Encryption, 24, 41, 42, 45, 53, 65, 69, 83, 93, 95, 96, 149, 188, 227, 229, 243, 250, 255, 258, 261  
technique, 41
- Endeavors, 17
- End-user applications, 48, 121
- Energy, 11, 57, 60, 138, 139, 142, 173, 174, 177, 186, 220, 248, 252, 254, 257, 258, 274, 278  
management systems, 138  
status, 11, 220
- Entropy, 158, 164, 165, 305, 310
- Epoch, 37, 57, 58
- Epsilon, 68
- Equipment inspection, 121
- Ethereum, 186, 240, 246, 252, 261
- Extensive research, 19, 140, 147, 157
- F**
- Facial  
emotion recognition system, 318  
expression recognition (FER), 297, 298, 311, 315
- Far-reaching technology, 126
- Fast  
data processing, 60  
gradient sign method (FGSM), 193
- Faster convergence, 167, 168
- Feasibility, 24, 92, 167
- Fed rescue, 167
- Fedaffect algorithm, 315, 318
- Federated  
AI technology enabler (FATE), 65  
algorithm (FedAvg), 57, 58, 82, 98, 167, 172, 185, 197, 200, 219, 316, 317  
architecture, 112, 232  
averaging  
cloud computing (FC), 49  
computing systems, 49  
database systems (FDBSs), 49  
distillation (FD), 164, 165  
extended MNIST (FEMNIST), 66  
learning (FL) , 1, 8, 10, 14, 16, 17, 47, 48, 53, 57, 59, 66–69, 77, 80–83, 87, 89, 90, 92–97, 101, 107, 126, 127, 132, 133, 139, 143, 144, 147, 155, 156, 164, 169, 175, 183, 184, 187, 194, 197, 201, 203, 205, 215, 216, 220, 226, 297, 230, 231, 238, 250, 251, 278, 286, 298, 300–302  
aggregation algorithms, 57  
algorithm, 59, 65  
application, 47, 96, 110, 155, 221, 250, 262, 291  
architecture, 24, 31, 32, 43, 45, 50, 84, 85, 218, 219, 240, 258, 259  
artificial intelligence, 126  
dataset, 65  
environment, 15, 54  
forbids, 53  
frameworks, 64, 101  
IoT services, 107  
medical services, 126  
models, 13, 16, 25, 100, 176, 249, 300, 302  
network topology, 54, 71  
scenario, 80, 92, 93, 221  
server, 54, 79, 91, 219  
system, 10, 13–15, 19, 49, 50, 52, 53, 55, 65, 69, 70, 77–79, 83, 86, 88, 89, 91, 93, 96, 99, 101, 132, 139, 143, 146, 153, 155, 156, 158, 161–163, 165–169, 171–176, 215, 217–219, 221, 222, 230, 232, 233, 260

- technique, 31, 285
  - technology, 64
  - topology, 54
  - training process, 70
  - workflows, 49
  - machine learning, 45, 94, 220, 299
    - method, 69
  - scenario, 227, 302, 310
  - stochastic gradient descent (FedSGD), 57, 58, 185, 198, 200
  - swapping, 68
  - transfer learning, 13, 14, 23, 28, 30, 31, 45, 56, 57, 84, 133, 183, 198, 200, 201, 242
  - Federation concept, 49, 71
  - Financial sector, 17
  - Fintech, 17
    - based applications, 17
    - technology, 17
  - Flower (flwr), 65
  - Fog
    - computing, 47, 59–61, 64, 101, 255, 264
    - nodes, 93, 257
  - Fokker-plank kolmogorov (FPK), 68
  - Fully homomorphic encryption (FHE), 196
- G**
- Gadgets, 114, 118, 120
  - Game, 280, 281
    - design, 281
    - theoretic approach, 285
    - theory, 89–92, 278–280, 282, 291, 292
  - Gaussian noise, 94, 202, 229
  - General data protection regulation (GDPR), 154, 184, 187, 188, 191, 214, 258, 272, 289
  - Generalization, 45, 98, 164, 172, 185, 302
  - Generative adversarial network (GAN), 93, 193, 205, 213, 225, 226
  - Geographical area, 137, 141, 142
  - Geometric model, 67
  - Gigabytes (GB), 48
  - Global
    - differential privacy (GDP), 94, 183, 184, 201–203, 206
    - loss function, 51, 172, 173
    - LSTM model, 69
    - ML model, 52, 71
- model, 12, 23, 24, 26, 27, 30, 34–43, 45, 49–51, 54, 56–58, 65, 69, 78–80, 82, 83, 90, 92, 97, 98, 132, 133, 142, 154, 155, 157, 164–167, 169, 174, 177, 203, 204, 217, 218, 223–225, 227, 241–243, 249, 253, 259, 261, 271–274, 276, 277, 279, 283, 285, 288, 290, 291, 302, 305
  - aggregation, 51, 97, 98
  - platform, 43
  - population, 186
  - security measures, 80
  - trained model, 54
  - Google, 48, 59, 64, 65, 98, 108, 132, 184, 198, 216, 254, 274, 285, 286, 299, 301, 309
  - keyboard, 54, 309
  - team, 66
  - Gradient, 9, 10, 30, 36, 37, 51, 57–59, 63, 67, 68, 77, 78, 82, 87, 88, 92, 93, 95–99, 101, 132, 133, 137, 147, 153–157, 159–163, 167, 169, 171, 175, 189, 192, 198, 200, 204, 216, 217, 219, 223–226, 228, 229, 231, 243, 281, 287, 310, 311
  - based multiple access (GBMA), 171
  - calculation, 30, 198
  - data, 154, 160, 162
  - descent, 37, 58, 67, 219, 243, 281, 310, 311
    - algorithm, 37
  - transfer, 82, 99, 153–155, 157, 169, 229
  - Graphical processing unit (GPU), 7, 48, 62
- H**
- Hadamard product, 168
  - Hamilton-Jacobi-Bellman (HJB), 68
  - Hardware, 7, 48, 65, 70, 121, 138, 250, 261, 282, 299, 305
    - system, 138
  - Health
    - data, 120, 260
    - devices, 125, 187
    - insurance portability and accountability act, 66
    - IoT, 121
    - supervision, 118
  - Healthcare, 6, 15, 16, 64, 66–69, 79, 94, 107–110, 112–124, 126–128, 132, 133, 139, 154, 155, 185, 197, 214, 216, 217, 238, 240, 249–252, 260–262, 264, 290, 292
    - applications, 110, 238, 240, 262, 264
    - automation, 116

- chatbot, 114  
data, 16, 113, 118, 260, 262  
industry, 16, 83, 109, 139, 217  
institutions, 300  
insurance, 17  
organizations, 67  
scenarios, 114, 127, 128  
sector, 16, 114, 261, 262, 290, 292  
services, 108, 109, 114, 127  
solutions, 123  
systems, 66, 108, 155, 217, 262
- Heart  
monitoring system, 116  
rate monitors, 115, 120
- Heterogeneity, 12, 19, 66, 71, 142, 143, 154, 155, 164, 166, 167, 172, 176, 251, 253, 278, 289, 291
- Heterogeneous, 9, 14, 19, 45, 57, 59, 70, 133, 139, 142, 143, 148, 155, 157, 159, 166, 168, 169, 171, 176, 201, 205, 220, 243, 287, 288
- client  
architectures, 45  
devices, 9
- collection, 45
- devices, 59
- edge device, 65
- environments, 133, 143, 144, 148, 159, 166, 220
- federated learning, 28
- FL, 14, 70, 168
- resources, 70
- systems, 70, 139, 142, 168, 171
- Hierarchical  
agent-based systems, 138  
aggregation, 58  
computing architecture, 64  
relationship, 61
- HierFAVG, 57, 58
- High  
correlation filter, 5  
dimensional Euclidean space, 309  
economic costs, 7  
performing system, 7
- Homogeneous, 14, 24, 157  
federated learning, 14, 24, 28
- Homomorphic encryption (HE), 24, 41, 42, 45, 65, 69, 93, 95–98, 149, 194, 196, 198, 261
- based aggregation, 41, 45
- Horizontal, 13, 14, 28, 55, 56, 65, 83, 84, 133, 139, 172, 199, 242, 278, 285, 286  
federated learning (FL), 13, 14, 23, 28, 45, 56, 83, 139, 183, 198, 242, 278, 286
- Host intrusion detection systems (HIDS), 135, 141
- Hub-and-spoke topology, 54
- Human  
activity recognition (HAR), 69  
intervention, 132
- Hurdles, 19, 47
- Hybrid, 7, 95, 138, 175, 240, 255, 258, 260  
approaches, 8  
architecture, 138  
encryption techniques, 95
- Hyperparameter, 7, 50
- Hyperplane, 4, 32
- Hyper-sphere quantization (HSQ), 157
- I**
- Ideal location, 4
- Idealistic security strategy, 91
- Image  
description, 63  
recognition, 63
- Imbuing intelligence, 24
- Implantable chips, 118
- Implementation flaws, 189
- Incentive mechanism, 57, 78, 81, 89, 92, 93, 101, 253, 271, 272, 274, 276, 277, 279–282, 284–287, 289–292  
design, 277, 279
- Incentivization, 88, 89, 91, 92  
mechanism, 92
- Indentation, 197, 198, 201, 204
- Independent and identically distributed (IID), 8, 68, 98, 142, 143, 160, 165, 166, 170–172, 226, 231, 301, 303
- Individual  
algorithms, 45  
dimensions, 94  
methodologies, 153  
perceptron, 32  
system, 7  
users, 18, 83, 189, 196, 216
- Industrial  
control, 52

- sectors, 17
- specific applications, 16
- Inference attacks, 93, 223–225, 230, 289
- Information
  - analysis, 125
  - overload, 116
  - security system, 133
  - technology, 124, 185
  - theft, 125, 186
- Infrastructure, 109, 116, 119, 121, 126, 137, 186, 239, 250
- Innovations, 113, 114
- Integration, 77, 78, 81, 86–88, 96, 100, 101, 107, 112, 113, 117, 123, 126, 133, 140, 142, 145, 146, 149, 175, 177, 184, 232, 233, 237, 251, 254, 257, 262
- efforts, 77
- Intelligent
  - brokers, 69–71
  - devices, 125
  - services, 107, 121
- Inter-communications, 67
- Interconnected devices, 109, 123, 132
- Inter-device communication, 168
- Interdisciplinary research, 80
- Interfaces, 85, 86, 88
- Intermittent device connection, 243
- Internet of
  - internet of health things (IoHT), 258
  - medical things (IoMT), 107–110, 112–116, 118, 120, 121, 124, 126–128
  - applications, 123
  - devices, 108, 112–115, 120, 121, 124, 127
  - sector, 115
  - systems, 112, 113, 126, 127
  - technology, 109, 124
- things (IoT), 1, 9, 10, 18, 19, 47–49, 59–61, 63, 64, 68, 69, 78, 107–110, 112, 115, 117, 119, 121–128, 138, 139, 144, 145, 147–150, 167, 169, 174, 175, 177, 214, 228, 232, 237–240, 248–251, 253–255, 257, 262, 264
- applications, 63, 107, 237, 251, 253, 257
- architecture, 48, 122, 257
- devices, 19, 59, 61, 68, 70, 108, 109, 144, 148, 228, 253, 254
- ecosystems, 70
- networks, 107
- revolution, 59
- systems, 9, 48, 109, 112, 126, 169, 174, 262
- Interplanetary file system, 59
- Intersection, 14, 28
- Intrusion, 132
  - detection, 80, 86, 93, 97, 131, 133, 135–137, 139–141, 145, 147, 149, 150
  - system (IDS), 80, 93, 97, 131–150
- Iteration, 26, 40, 43, 82, 93, 100, 154, 158, 161, 166, 171–173, 216, 218, 227, 230, 241, 283, 287
- Iterative learning, 54
- J**
- Jacobian-based saliency map attack (JSMA), 193
- Jamming
  - areas, 68
  - attack, 68
- Jensen-Shannon divergence approach, 165
- K**
- Key management, 42
- Kiosks, 118
- Knowledge, 56, 115, 123, 134–136, 163, 166, 167, 185, 191, 194, 226, 241, 255, 264, 301
- distillation (KD), 163, 164, 166–168, 176
- L**
- Label flipping attacks, 148
- Large-scale
  - data management, 123
  - models, 77
  - real-world systems, 65
- Latency, 7, 9, 10, 48, 52, 55, 60, 61, 64, 109, 140, 141, 149, 172, 175, 176, 238, 239, 243, 257, 259, 261, 284
- Lattice, 158
  - quantization, 158
- Leakages, 213, 221, 225, 226
- Learned model parameters, 32
- Learning
  - architectures, 24, 31, 45, 239

environment, 143, 224, 277, 287  
models, 23, 31, 33, 44, 45, 108, 193, 251,  
260, 273, 274, 290, 309  
mood, 110  
paradigm, 66  
platform, 154, 291, 292  
process, 23, 24, 26, 30–33, 62, 197, 244,  
272, 273, 276, 277, 279, 281, 284, 288,  
290, 291, 302  
Legalization, 48  
Legislation, 237  
Leveraging, 49, 159, 189, 223, 224, 253,  
297, 318  
distributed resources, 49  
Lifestyle, 114, 118  
assessment, 118  
Lightweight devices, 17, 19, 97, 133, 153,  
214, 216, 230  
Limited-memory Broyden-Fletcher-Gold-  
farb-Shanno (LBFGS), 192  
Linear regression, 4  
Local  
data, 50, 58, 78, 80, 82, 88, 98, 133, 154,  
165, 166, 175, 217, 238, 239, 277, 282,  
299, 303, 305, 318  
differential privacy (LDP), 67, 94, 183,  
184, 194, 201–203, 206, 254  
epochs, 37, 309  
images, 67  
model, 26, 30, 35–38, 40–44, 51, 54, 56,  
58, 67, 69–71, 79, 80, 82, 83, 87, 92,  
93, 98, 101, 132, 155, 164–167, 175,  
200, 204, 217, 223, 241, 249, 254, 258,  
271, 272, 276–278, 281, 302, 309, 310  
training, 34, 36, 40, 54, 82, 83, 92, 98,  
132, 217, 278  
training process, 67  
Logistic scoring, 165  
Long term  
cooperation, 289  
evolution (LTE), 109  
Loopholes, 189  
Low  
level physical devices, 125  
network bandwidth, 60  
quality data, 142  
specification systems, 9  
variance filter, 5

## M

Machine learning (ML), 1–3, 5, 7–10, 15,  
17, 23–25, 31, 44, 45, 48, 49, 61, 67,  
77–79, 81, 94, 97, 102, 108, 109, 132,  
133, 137, 143, 153, 183, 184, 187, 193,  
214–216, 220, 221, 231, 237, 238, 241,  
243, 248, 249, 251, 259, 260, 271–274,  
276, 278, 284, 285, 287, 289–292,  
297–302  
algorithm, 2, 7, 8  
approaches, 8, 51–53, 56, 78, 132, 143  
models, 9, 51, 52, 58, 70, 82, 153,  
192–194, 196, 198  
systems, 4, 9, 19, 52, 189  
technology, 7  
Malicious, 80, 92, 96, 102, 132, 133, 136,  
137, 145, 149, 174, 200, 204, 215, 220,  
226, 247, 249, 276–278, 282  
aggregators, 204  
attacks, 132  
data, 204, 223  
Markov decision processes, 6  
Mask vector, 201  
Masquerade, 204  
Mean  
feature initialization, 316, 318  
field game (MFG), 67, 68  
Medical  
data, 66, 108, 123, 124, 154, 217, 250, 260  
device, 108, 109, 117, 120, 121  
diagnostics, 128  
professionals, 108, 113–116, 118, 122  
resources, 116  
services, 124–128  
systems, 121  
technology, 126  
Medicine vending robots, 6  
Medium access channel (MAC), 170  
Minimal transmission power, 68  
Ministry of Health, 67  
Minor perturbations, 192  
Mitigation, 19, 70, 190, 232  
module, 70  
schemes, 19  
Mobile  
computing  
devices, 10  
systems, 9

- devices, 49, 53, 71, 90, 108, 114, 120, 126, 169, 174, 216, 238, 249, 255, 281, 318
  - phones, 19, 47, 107, 111, 131, 299
  - systems, 19, 89
  - Mobility, 19, 52, 61, 109, 122, 125, 147, 169, 186
automation, 52
  - Model
aggregation, 12, 25, 26, 31, 34, 36–41, 44, 49, 51, 54, 55, 58, 67, 78, 82, 83, 87, 92, 96–99, 140, 145, 146, 157, 158, 167, 169, 215, 217, 221
 
techniques, 36
  - deployment, 2, 8
  - selection (OMS), 24
- Monitoring, 2, 49, 69, 107, 109, 110, 112, 113, 115–125, 127, 135, 186, 197, 253, 254, 262, 297–299
devices, 109, 115
- Motion blur, 67
- Multi-center systems, 19
- Multilayer
architectures, 62
 
perceptrons (MLP), 33
- Multi-party computation, 65
- Multiple
aggregation servers, 58
 
client devices, 77, 86, 153
 
devices, 77, 132, 169, 197, 216
 
edge devices, 52, 53
 
smaller systems, 7
- N**
- Nash equilibrium, 91, 174, 282
approach, 91
- Natural language procession (NLP), 205
- Near field communications (NFC), 109
- Network, 2, 9, 17, 18, 31, 55, 60, 67, 77, 86, 96, 107, 131, 132, 140–146, 149, 153, 160, 168–170, 176, 218, 220, 232, 237, 239, 249, 251, 253, 260, 264
architecture, 1, 8, 80, 220, 251
 
connectivity, 70, 71, 176
 
edge, 63, 64, 80, 176
 
devices, 63
- intrusion detection systems (NIDS), 135, 141
- parameters, 63
- topology, 54
- traffic, 134, 139, 141, 240
- Neural network (NN), 2, 31, 33, 36, 37, 61–63, 134, 137, 193, 204, 220, 300
model, 31, 33, 36, 37
- Neuroimaging, 116
- Neuron, 2, 37, 300
weights, 40
- Neurophysiologist, 2
- Noise addition, 94, 228, 229, 233
- Noisy fading channel, 170, 171
- Non-federated learning, 51, 71
- Non-imaging diagnostics, 115
- Nonlinear
control, 137
 
cyber-physical systems, 137
 
problems, 32
- Non-repudiation, 190
- Non-reversible process, 192
- Nonzero data index, 163
- Nucleic acids, 121
- NVIDIA federated learning application
runtime environment (NVIDIA FLARE), 65
- O**
- One model selection (OMS), 24, 37–39, 45
- On-site training, 18, 82, 213
- Open-source software development, 65
- Operating system, 43, 44, 65, 134, 191
- Operational data, 122, 123
- Optimal
defense, 68
 
strategy, 91, 285
 
vector quantizers, 158
- Optimization, 6, 47, 58, 60, 139, 142, 157–159, 162, 163, 167, 172–174, 177, 185, 192, 193, 198, 203, 219, 281, 309
- Oracle, 123
- Orchestrate, 55
- Orchestration, 49, 50, 69
- Output data, 3, 164
- Overlap, 163
- Overlapping features, 83
- Overloading, 143

**P**

Paddle federated learning (PFL), 65  
Paradigm, 1, 7, 10, 15, 19, 47, 59, 64, 66, 80, 81, 101, 126, 131, 132, 153, 213, 215, 221, 222, 237, 238, 255, 259, 271, 273, 274, 276, 277, 281, 283  
Parameter broadcasting, 54  
Parameterization, 168  
Partitioning approaches, 5, 172  
Pedestrian behavior, 110, 112  
Perceptron, 2, 31–33  
learning algorithm, 32  
Perceptual extraction network (PEN), 69  
Personal emergency response systems (PERS), 124, 125  
Perturbation, 192, 193, 196, 202, 203  
Pharmaceutical companies, 49, 121  
Pharmacodynamic dosage, 121  
Plex media server, 186  
Point-of-care (POC), 113, 121  
Poison GAN, 225  
Poisoning attacks, 96, 148, 201, 204, 213, 221, 222, 225, 227, 231–233  
Polling method, 198  
Post-acute care, 120, 127  
Prediction, 7, 18, 63, 66, 78, 110, 123, 124, 193, 216, 238, 286, 290, 297, 298, 309, 311  
Privacy  
enhancing technologies (PETs), 190  
issues, 9, 25, 64, 112, 150, 183, 205, 213, 260  
loss (PL), 94, 195  
preservation, 1, 19, 45, 78, 79, 93–96, 102, 132, 133, 154, 165, 176, 183–187, 189, 196, 203, 205, 206, 214, 228, 229, 233, 272  
preserving machine learning (PPML), 183, 184, 187, 194, 205  
sensitive federated learning (PSFL), 303  
vulnerabilities, 48  
Proactive methods, 227  
Prognosis, 66  
Prone, 17, 19, 148, 149, 220, 221  
Proof of stake (PoS), 240, 252  
Property inference attack (PIA), 205, 206, 224

Protected-shot-based federated learning (PSFL), 297–300, 303, 306, 310

Pseudonymization, 190–192

Python, 65

**Q**

Q-table, 68  
Qualitative biometric data, 123  
Quality, 2, 9, 64, 67, 77, 79, 82, 88, 91, 93, 99, 100, 109, 118, 119, 121, 123, 126, 127, 133, 142, 147, 149, 154, 183, 257, 260, 271, 272, 276–279, 284, 288, 290, 301  
control, 123  
Quantify, 67  
Quantization, 99, 156–160, 164, 167, 168, 253  
approach, 157–160, 164, 167  
result, 159  
Quantized stochastic gradient descent (QSGD), 156, 157  
Query suggestions, 54

**R**

Radio frequency identification (RFID), 121  
tag (RFID Tag), 109  
Radiology, 116  
Random  
mask vectors, 201  
transfer, 55  
Rapid  
evolution, 154  
pace, 24  
Rationale, 19, 90, 112, 131, 142, 161, 228, 232  
Raw data, 4, 9, 48, 49, 51–53, 64, 69, 109, 134, 154, 162, 197, 258, 278, 289, 302  
Real  
life applications, 5, 16, 98, 154, 160  
time  
analytics, 60  
applications, 64, 137  
business, 60  
data, 18  
media, 52  
world applications, 49, 173, 301

- Recognition, 2, 4, 63, 69, 110, 135, 253, 287, 297, 298, 303, 304, 306, 309, 311, 313–315, 318  
tasks, 4, 306
- Recurrent neural network (RNN), 33, 34, 220
- Regression, 3, 4, 31, 189, 284, 300, 302, 306
- Regularizer, 299, 303, 312, 315–318
- Reinforcement learning, 6, 68, 90, 148, 253, 254, 278, 279, 285
- Relays, 171, 299, 303
- Remote  
intervention, 116, 119  
monitoring, 116, 120  
patient monitoring (RPM), 125
- Replication, 8
- Reputation, 93, 98, 102, 148, 252, 277, 278, 282, 286–288
- Resource management, 102, 139, 251, 253, 254
- Restrictions, 188, 290, 301
- Rich execution environment (REE), 43, 44
- Robustness, 159, 164, 171, 221, 227, 253, 254, 302
- ## S
- Safeguard, 137, 187, 194, 252
- Satellite-terrestrial integrated networks (STIN), 142
- Scalability, 52, 60, 61, 64, 65, 87, 95, 101, 102, 107, 140, 240, 247, 252, 253, 257, 258, 271, 298  
issues, 102, 257
- Scale invariant feature transform (SIFT), 306
- Sealed-bid auctions, 283
- Secure  
aggregation (SA), 96, 197, 198, 200–203, 261, 275  
protocol, 41, 45, 261  
multiparty computation (SMC), 93–98, 149, 167, 196, 198, 201, 229, 233
- Security breaches, 19, 213, 220, 222, 226
- Semi-supervised learning, 68, 143
- Sensors, 63, 69, 70, 109, 112, 114, 118–120, 122, 124, 125, 127, 239, 257
- Sequential data, 33
- Server devices, 12, 147
- Server-side global model, 49
- Several edge devices, 52
- Shakespeare dataset, 66
- Shaky information transmission, 125
- Shapley value, 91, 278, 280, 281
- Sheer heterogeneity, 19
- Silos, 10, 81, 154, 197, 237
- Simple neural network, 62
- Simplified payment verification (SPV), 244
- Single  
cluster, 52  
server, 49, 54, 77, 146
- Small-scale devices, 16
- Smart  
appliances, 122  
device, 26  
grids, 138  
wearables, 109, 113
- Smartphone, 63, 109, 111, 121, 187, 216, 238  
industry, 18
- Social  
media, 28, 184, 186  
networking, 18, 185, 186
- SoftMax function, 165
- Software defined network (SDN), 77, 80, 93, 101, 102
- Somewhat homomorphic encryption (SWHE), 196
- Sparse  
binary compression, 161  
representation methods, 162
- Sparification, 99, 102, 147, 153, 156, 160–163, 168, 176, 177, 228, 253  
approach, 99, 160–162  
target, 160  
techniques, 160  
technology, 99, 102
- Speech  
impairment, 120  
recognition, 2, 63, 110
- Stack overflow dataset, 66
- Statistical models, 110, 237, 238
- Stochastic  
approach, 157  
gradient  
approach (SGD), 51, 57, 58, 98, 156, 168, 170, 171, 203, 224, 243, 305, 310  
descent (SGD), 51  
quantization (SQ), 156, 157

- Straggler, 54, 70  
identification, 70  
nodes, 70
- Surveillance, 69, 121, 132, 138
- System  
architecture, 8, 15, 66, 80, 89, 134, 139, 143, 217, 218, 222, 228, 233  
performance, 6, 24, 100, 101, 133, 147–149, 155, 156, 161, 162, 166, 168, 169, 171, 175, 224, 282  
work, 91, 120
- T**
- T communication cycles, 303, 318
- Tackle, 33, 167, 214, 239
- Tamper-proof, 43, 135, 240
- Tandem, 60
- Technical  
assistance, 6  
requirements of AI applications, 67
- Technological soundness, 45
- Technology-specific applications, 16
- Telecommunications, 49, 185, 238
- Telehealth, 114
- Telemedicine, 116–119, 124, 125
- Tensor processing units, 7
- TensorFlow, 65, 313  
federated (TFF), 65
- Terminologies, 1, 20, 81, 215, 233
- Thelocal server, 26
- Theoretical convergence analysis, 171
- Three-dimensional (3D) space, 67
- Threshold, 51, 98, 161, 203
- Time  
consumption, 173, 177  
correlation sparsification (TCS), 162  
critical applications, 52
- Topology, 8, 54, 55, 71, 80, 82, 168, 198, 215, 233
- Traditional  
central networks, 71  
computing systems, 62  
learning, 25, 149, 150, 160, 161, 215, 222  
methods of AI, 6
- Training  
local models, 50  
procedure, 9, 12, 82, 96, 214, 217, 291  
process, 19, 49–51, 54, 55, 59, 68, 70, 155, 175, 193, 198, 230, 239, 272, 276, 282, 287
- Trusted execution environment (TEE), 24, 43–45, 189, 230
- Trutag technologies, 121
- Two-dimensional (2D) data, 33
- U**
- Uber, 186
- Ultrasound, 67, 66
- Unified model, 35
- Unmanned aerial vehicles (UAV), 67, 68, 107, 251
- V**
- Vehicular  
technology, 67  
edge computing (VEC), 67  
insurance, 17  
networks, 67  
users, 67
- Vertical  
federated learning, 13, 14, 23, 28, 30, 45, 56, 83, 84, 95, 96, 133, 139, 183, 198, 199, 242, 286  
systems, 14, 95, 139
- Virtual  
appointments, 115  
platforms, 18
- Visual  
data, 67  
perception, 125
- Visualization, 109
- Vocal devices, 114
- W**
- Watermark attack, 205, 206
- Weak organizational ties, 125
- Wearable, 114, 117, 121, 125, 205  
based sensors, 69  
device, 16, 66, 112, 115, 131, 217  
systems, 6  
technology, 110, 118
- Web  
anomaly detection, 137  
applications, 137  
searches, 2
- Weight, 32, 33, 38, 67, 98, 160, 165, 167, 169
- Well-being, 124, 127

Wide range, 64, 107, 126, 134–136, 139, 193, 222, 238  
Widespread, 9, 61, 64, 115, 126, 127, 219, 291, 316  
Wi-Fi connectivity, 120  
Windy conditions, 67  
Wireless systems, 173  
Wrapping, 5  
Wristbands, 125  
WuXi pharmatech, 121

**X**

X-rays, 66, 67, 121

**Y**  
Yahoo, 186  
Yield, 90, 100, 226, 229, 291

**Z**

Zero  
    cons, 18  
    day attacks, 134  
    knowledge, 226  
    trust data sharing, 188  
zones, 331