UNIVERSITY OF LEEDS

Decentralised Federated Learning over Wireless Communication Networks

Abdelaziz M. Salama

Submitted in accordance with the requirements for the degree of PhD. in Electrical and Electronic Engineering

The University of Leeds

Faculty of Engineering and Physical Sciences

School of Electrical and Electronic Engineering

May 2024

Intellectual Property

Signature: *Abdelaziz Salama*

Declaration

The candidate confirms that the submitted work is his own, except for portions that are part of jointly authored publications. The contributions of both the candidate and the co-authors to these works have been clearly specified below. Proper acknowledgement has been given in the thesis wherever references have been made to the work of others.

The work in Chapter 2 has appeared in publication as follows:

A. Salama, A. Stergioulis, A. M. Hayajneh, S. A. R. Zaidi, D. McLernon, and I. Robertson, "Decentralised Federated Learning over Slotted ALOHA Wireless Mesh Networking," IEEE Access, vol. 11, pp. 18,326–18,342, 2023.

I declare that the work presented in this chapter is my own. I take full responsibility for the key contributions outlined. Dr Des McLernon and Dr Syed Ali Zaidi, my supervisors, are co-authors of the included paper, as they offer crucial insights, constructive feedback, and constant encouragement throughout my PhD journey. Additionally, Dr. A. Stergioulis contributed by reviewing the manuscript, A. M. Hayajneh assisted with setting up the simulation software, and I. Robertson provided advice on the paper structure.

The work in Chapter 3 has appeared in publication as follows:

A. Salama, A. Stergioulis, S. Zaidi, and D. McLernon, "Decentralised Federated Learning on the Edge over Wireless Mesh Networks," IEEE Access, vol. 11, pp. 124,709–124,724, 2023.

I declare that the work presented in this chapter is my own. I take full responsibility for the key contributions outlined. My supervisors, Dr Des McLernon and Dr Syed Ali Zaidi are co-authors of the included paper, as they provided indispensable guidance, strategic direction, and thoughtful critiques throughout my PhD journey. Additionally, Dr. A. Stergioulis contributed by reviewing the manuscript and helping with the geometric analysis used in the paper.

The work in Chapter 4 is submitted and under review:

A. Salama, S. A. Zaidi, D. McLernon, and M. M. Qazzaz, "Enhancing Byzantine-Resilience in Decentralised Federated Learning for Dynamic Networks," IEEE Transactions on Machine Learning in Communications and Networking, (Submitted in March, and still under review).

I declare that the work presented in this chapter is my own. I take full responsibility for the key contributions outlined. Dr Des McLernon and Dr Syed Ali Zaidi, my supervisors, are co-authors of the included paper, as they provided exceptional mentorship, comprehensive feedback, and unwavering support throughout my PhD journey. Additionally, M. M. Qazzaz contributed by reviewing the manuscript and assisting with setting up the simulation software.

The work in Chapter 5 is submitted and under review:

A. Salama, S. A. Zaidi, D. McLernon, M. Hafeez, and M. M. Qazzaz, "Enhancing Reliability and Efficiency in Collaborative Learning through Decentralised Federated Learning using MPTCP," IEEE Open Journal of the Communications Society (Submitted in April, and still under review).

I declare that the work presented in this chapter is my own. I take full responsibility for the key contributions outlined. Dr Des McLernon and Dr Syed Ali Zaidi, my supervisors, are co-authors of the included paper, as they provided invaluable oversight, detailed feedback, and critical advice throughout my PhD journey. Additionally, M. Hafeez contributed by reviewing the manuscript, providing a theoretical explanation of the MPTCP communication protocols, and advising on the paper structure. M. M. Qazzaz also assisted with setting up the simulation software.

Signature: *Abdelaziz Salama*

Abstract

As the proliferation of IoT devices and smart technologies continues, their potential remains underutilised due to significant privacy and data sensitivity concerns. In this thesis, I present a secure, decentralised, and intelligent framework designed to optimise diverse problems through Federated Learning (FL) schemes and machine learning models on endpoint devices. Unlike traditional centralised approaches, this framework enables collaborative problem-solving without data sharing, relying instead on the exchange of model updates to refine a global solution. The key to this decentralised model is the establishment of robust and efficient communication networks that manage interactions and data exchanges between devices.

This work explores Decentralised Federated Learning (DFL), which enhances participant collaboration while ensuring privacy and mitigating the communication bottlenecks inherent in Centralised Federated Learning (CFL). By improving inter-device communication within the DFL network and optimising the associated learning models, I aim to boost overall system performance and reliability. Furthermore, the presence of adversarial devices poses significant threats; thus, strategies to exclude untrustworthy devices are critical to maintaining the integrity and efficiency of the network.

This thesis contributes towards a comprehensive analysis of network communication, geometric configurations, and system robustness. I introduce innovative DFL models and simulation techniques, demonstrating a robust and server-free FL process. Enhancements in model accuracy have been achieved, leading to an intelligent, low-latency, and adaptable framework suitable for various important applications, including Autonomous

Vehicles (AVs) and IoT systems.

While further advancements are necessary, this thesis marks substantial progress towards a flexible and distributed DFL scheme. It is anticipated that this foundational work will encourage continued enhancements in communication efficiencies, fostering more effective collaboration and sustained privacy in FL environments.

Acknowledgements

I extend my deepest gratitude to my supervisors, Dr Des McLernon and Dr Syed Ali Zaidi, whose extensive support and guidance have been invaluable throughout my research journey. I do appreciate their enduring support, assistance, and kindness over the last four years. Their mentorship has not only enriched my academic experience but also provided me with a prestigious opportunity to collaborate and learn.

I would like to extend my sincere gratitude to the Ministry of Higher Education in Libya for generously funding my PhD studies.

I also owe a tremendous debt and am grateful to my dear wife, Takwa Treki, whose love and support have been my anchor throughout this journey. My life is so much brighter, and my challenges are more manageable because of our shared path. I also give special thanks to our beautiful twins, Zain and Aleen, who were born just over two years into my PhD studies. They have both enriched our lives so much as well as making night sleep a thing of the past!

Lastly, my heart reaches out across continents to Libya, where my parents, Mohamed Salama and Fawzia Aldeeb, along with my extended family, have given me steadfast emotional support. Their love, stretched over miles, has been a constant source of comfort and motivation. Their sacrifices and belief in my abilities have been fundamental to my success.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In recent years, the proliferation of the Internet of Things (IoT) and smart wearable technologies has significantly expanded across various sectors, including wearable devices, home automation, autonomous vehicles, health monitoring systems, fitness trackers, etc. The wireless devices within these sectors' landscape are generating vast quantities of data, necessitating extensive processing. Typically, in a star network configuration where devices are linked to a central server, as depicted in Fig. 1.1, this data is routed to centralised servers for analysis in the traditional cloud computing framework. Subsequently, processed results are dispatched back to the originating devices, placing a considerable strain on network resources. Moreover, the transmission of data can demand significant bandwidth and incur costs related to resource utilisation, such as energy expenditure, potentially leading to diminished network efficiency as data volumes expand [1]. In response to these issues inherent in standard cloud computing, edge computing has surfaced as a viable solution, positioning processing and data storage capabilities closer to the source devices [2]. Within this context, machine learning (ML) emerges as a key technique for digesting data generated at the edge [3]. Applications span a broad spectrum, including vehicle networks, smart urban development, self-driving cars, security surveillance, digital health solutions, drones, protective robots, and a range of industrial IoT scenarios [4].

The voluminous data collected by sensors and edge devices holds paramount importance. Leveraging advanced ML techniques, this data can be harnessed to develop classification models capable of learning and making predictions to fulfil end-user demands efficiently. ML algorithms are pivotal in enabling a wide array of applications, such as home automation control, autonomous vehicle operation, and health monitoring for seniors, including heart rate and fall detection. However, edge device generated data often contains highly personal details, raising substantial concerns regarding data privacy and security. The reluctance to centralise sensitive information underscores the need for privacy-preserving solutions.

Federated Learning (FL) [5], a collaborative learning approach, has been introduced to mitigate privacy and security concerns by allowing for model development across dis-

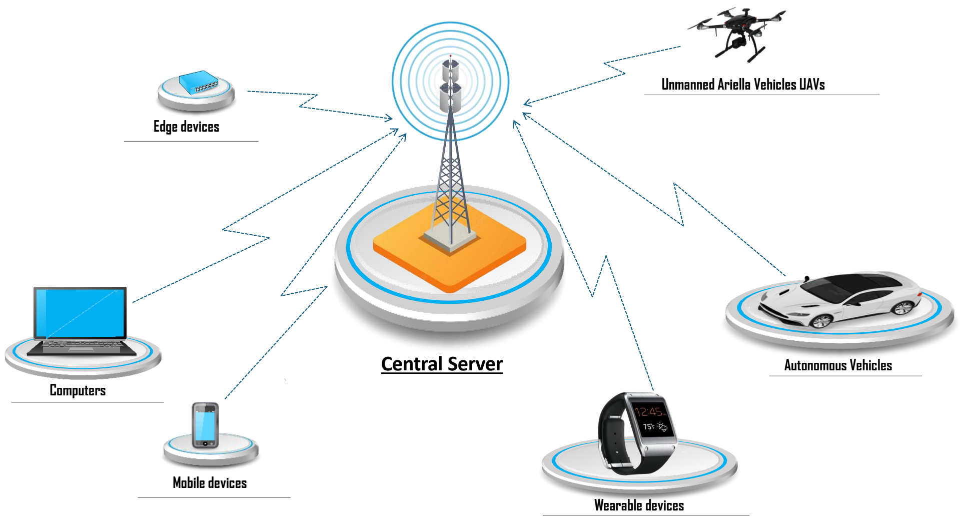


Figure 1.1: Example of the traditional centralised system layout

tributed ML frameworks. This method enables the construction of models from disparate, sensitive local datasets, facilitating distributed training while ensuring user data remains confidential [5], [6].

FL's capacity to safeguard client data privacy by sharing only model parameters derived from local training rather than the raw data itself has garnered considerable interest. Figure 1.2 illustrates a typical deployment scenario within a healthcare context, highlighting how entities such as Community Hospitals, Research Medical Centres, and Cancer Treatment Centres can independently train models on their respective confidential datasets and collectively enhance system performance by aggregating the model parameters to forge a comprehensive global model.

While FL systems hold considerable promise, they encounter numerous obstacles, chiefly due to their dependence on a central FL server. A paramount issue within FL is establishing efficient communication to ensure connectivity among all devices and maintain optimal performance [8]. This challenge is particularly pronounced in environments where securing a dependable central server proves challenging, such as in entirely autonomous networks. Additionally, the reliance on a singular server introduces the risk of a single point of failure, potentially leading to communication bottlenecks. Consequently, this study investigates the potential of Decentralised Federated Learning (DFL) as a strategy

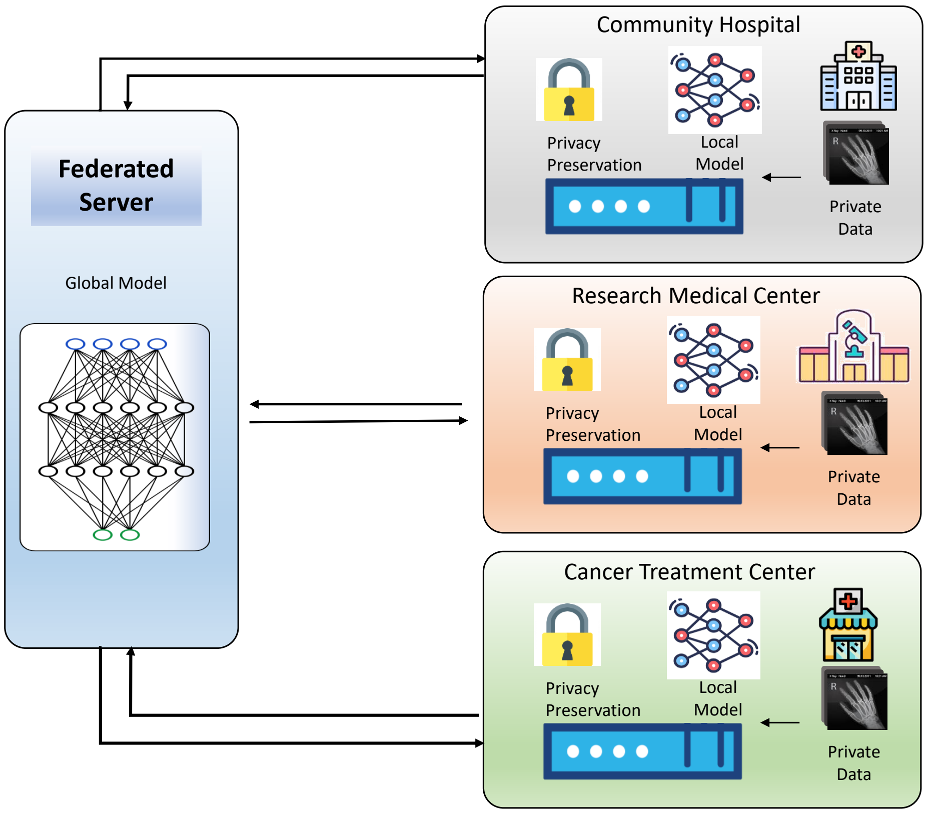


Figure 1.2: Federated learning example diagram [7]

to surmount these hurdles. In the DFL framework, devices exchange model parameters directly via peer-to-peer communication, with each device functioning as a mini-server, aggregating and averaging parameters from its neighbours to refresh a local sub-global model accessible within a one-hop communication radius.

So one of the objectives of this thesis is to assess the practicality of implementing the DFL methodology amidst varying spatio-temporal conditions. In Chapter 2, I deploy DFL algorithms within mesh networks situated in diverse settings, leading to two distinct practical deployment strategies: 1) Autonomous Decentralised IoT Networks, typical of scenarios requiring intrinsic decentralisation due to limited device capabilities, such as wearable technologies or remote medical applications, and 2) Edge/Fog-Assisted IoT Networks, hierarchical structures where IoT data is initially processed by edge/fog gateways before forming a mesh network for DFL application, with agricultural IoT systems and smart city initiatives serving as prime examples [9], [10].

Furthermore, in this thesis, I pioneer the formulation of communication and computational efficiency in DFL across wireless networks through the implementation of efficient methodologies. These methodologies eliminate the need for information about future training data. I explore the integration of DFL with some protocols as a mechanism to probe network performance and ideal configurations within these models. Therefore, I delve into the creation of methods that efficiently handle both communication and computation, encapsulating the overarching expenses associated with conducting DFL in wireless networks, independent of the particular protocol used for exchanging parameters between the participant devices. Consequently, the theoretical findings in Chapter 2 and Chapter 3 hold broad relevance across a spectrum of wireless communication protocols, including Slotted-ALOHA and Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA), respectively. Given the versatility of my proposed approaches across various wireless communication frameworks, this research holds substantial promise for further exploration and application by both the academic and industrial sectors. To the best of my knowledge, this thesis presents research at the forefront of leveraging geometric analysis and a variety of aggregation techniques. The aim is to markedly enhance both communication efficiency and learning effectiveness within DFL systems.

Proposing the integration of DFL with the slotted ALOHA protocol is designed to bolster network flexibility as devices collaborate in a peer-to-peer fashion. This protocol aligns well with a broad spectrum of existing applications and devices. Additionally, the DFL framework aims to strengthen data privacy, enhance the protection of sensitive information, and elevate the predictive precision of algorithms in use. Consequently, this approach seeks to establish a robust system architecture that is resilient in the face of challenges. Such a framework promises to leverage a broader spectrum of user data for training, refining a global model to augment local models across participant devices, thereby obviating the need to exchange any data other than local model parameters.

1.1 Background

This section presents a brief summary of the basic principles and previous steps taken in DFL. It briefly outlines the evolution of machine learning techniques that facilitate the execution of learning tasks directly on distributed devices, focusing on the shift from centralised to decentralised approaches. The discussion highlights key technological advancements and theoretical underpinnings that have shaped current practices and research directions. While this section provides the necessary context to understand the scope of the thesis, a more comprehensive exploration of related works and detailed discussions of specific methodologies and their applications are presented in the subsequent Chapters. This structure ensures a foundational understanding before delving into the in-depth analyses and novel contributions of the included papers.

1.1.1 Distributed Machine Learning

There are a variety of machine learning (ML) and deep learning algorithms employed across numerous studies. For example, Convolutional Neural Networks (CNNs) are extensively used for image classification and other similar tasks due to their demonstrated capability to achieve high accuracy and effectively learn from extensive datasets of thousands of images [11].

In the era of the IoT, where many devices collect a large amount of data through different services and applications, integrating efficient wireless networks for the modern distributed ML is highly important [12]. Furthermore, the ever-increasing use of data for wireless communications has made it almost impossible to transfer local rows of data to central servers due to security and privacy issues, as well as the potential to massively transfer a large data size. Hence, FL offers a solution to lower data transfers and decrease bandwidth consumption in wireless communication networks.

FL can be defined as a distributed ML approach that enables devices to collaborate without sharing personal or sensitive information. In this model, data remains on the device, and only the model parameters are shared with others. To deploy traditional FL, also referred to as centralised FL (CFL), along with its alternative, Decentralised DFL, it is crucial to embed local machine learning algorithms within the terminal devices (participants). This setup allows each device to train the algorithm locally on its data before parameters are shared either with a central server in CFL or directly with neighbouring devices in DFL. In this thesis, I aim to apply both CFL and DFL strategies to tackle a classification problem. Therefore, the CNN algorithm will serve as the primary method for training local models within the proposed systems. This choice is predicated on CNN's ability to efficiently handle large volumes of data and its effectiveness in extracting meaningful patterns from complex datasets, thereby enhancing the overall performance and reliability of the FL framework.

1.1.2 Centralised Federated Learning (CFL)

The primary goal of CFL systems is to train a global model for optimising a certain ML algorithm performance via collaboration between participating devices through a central server that performs model aggregation from these participants, where each device is represented as a local learner [5]. Figure 1.2 illustrates the essential architecture of CFL and outlines the four key phases involved in training a CFL network, which is repeated until the system achieves convergence [13]:

1. Local Learning: Each edge device processes its own data to train the model locally, updating crucial model parameters like neural biases and weights.
2. Uploading Parameters: Devices send their updated model parameters to the central server through communication channels.
3. Global Aggregation: Upon receiving the parameters, the central server combines them to refresh the global model.
4. Model Synchronisation: The updated global model is then broadcast back to the devices. The cycle returns to the first step and continues until the system converges [14].

In CFL, local algorithms on each device typically employ optimisation techniques like Stochastic Gradient Descent (SGD) to iteratively refine the model. The global model is then formed by aggregating these locally updated parameters, potentially weighting them based on the quality of the updates [11]. A key feature of CFL is that it keeps user data on the device, minimising data transmission and addressing privacy concerns. Nevertheless, CFL can encounter scalability problems due to its reliance on a central server. Despite optimisation of the server's hardware and software, performance issues may arise as the number of client nodes grows [15]. This can lead to communication bottlenecks as traffic volumes increase, overloading the system. Moreover, accessing a central node may not always be feasible, for example, in scenarios involving autonomous vehicles or high-mobility sensor systems. To circumvent these issues, decentralised architectures have been proposed, eliminating the need for a central server. In these systems, devices communicate only with their immediate neighbours, sharing and updating models locally [16]–[18].

The training process is initiated by the initial model parameter ($\mathbf{W}_0$). At the beginning of each CFL iteration $k = 1, \ldots, K$, the server shares $\mathbf{W}_{k-1}$ with all participating devices. Each device $j \in \{1, 2, \ldots, M\}$ holds a subset $\mathcal{D}_j = \{(x_n^j, y_n^j)\}_{n=1}^{N_j}$ of the entire dataset $\mathcal{D}$, with $\mathcal{D} = \cup_{j=1}^{M} \mathcal{D}_j$, where N_j denotes the dataset size of the j-th device. Each device engages in Stochastic Gradient Descent (SGD) on the local loss function. The device subsequently updates its local model weights using the following equation:

$$\mathbf{W}_k^j = \mathbf{W}_{k-1} - \frac{\alpha_k}{N_j} \sum_{n=1}^{N_j} \nabla_W f(h(x_n^j, \mathbf{W}_{k-1}), y_n^j), \quad k = 1, \ldots, \mathbf{K}, \tag{1.1}$$

here α_k represents the learning rate. Subsequently, each device transmits its local model parameters $\mathbf{W}_j^k$ to the server for global aggregation. This update step, known as the global update, is defined by the equation:

$$\mathbf{W}_k = \sum_{j=1}^{M} \rho_j \mathbf{W}_k^j, \quad k = 1, \ldots, K, \tag{1.2}$$

where $\rho_j := \frac{N_j}{N}$ denotes the weight of the j-th device's dataset size relative to the total dataset size N. Furthermore, each device j updates its local parameters by performing a single iteration of the Stochastic Gradient Descent (SGD) algorithm [19] on a subset of its local data. This approach, where local SGD updates are used, is referred to as Federated SGD (FedSGD) [5] [20].

Federated Averaging (FedAvg) is an advanced adaptation of FedSGD, featuring distinct computations for local parameters. In FedAvg, the server dispatches W_{k-1} to the devices at the start of each iteration $k = 1, \ldots, K$. Each device $j \in \{1, 2, \ldots, M\}$ maintains a subset $D_j = \{(x_{jn}, y_{jn})\}_{n=1}^{N_j}$ of the collective dataset D, where $D = \bigcup_{j=1}^{M} D_j$, and executes E local iterations of SGD, $i = 1, \ldots, E$, on a subset of $N_k^j \leq N_j$ data points. The local parameter $W_{i,k}^j$ is updated as follows:

$$\mathbf{W}_{i,k}^j = \mathbf{W}_{i-1,k}^j - \frac{\alpha_k}{N_k^j} \sum_{n=1}^{N_k^j} \nabla_{\mathbf{W}} f(h(x_n^j, \mathbf{W}_{i-1,k}^j), y_n^j), \quad k = 1, \ldots, K, \tag{1.3}$$

where $\mathbf{W}_k^j = \mathbf{W}_{E,k}^j$ [21]. This updated formula corresponds to the global update previously described in Eq. 1.2.

Practically, not all devices (clients) participate in each FedAvg iteration due to bandwidth or network congestion. Consequently, only a subset of potentially influential devices is chosen for training in each iteration, a process known as client selection. The goal of client selection is to identify informative clients that contribute to faster convergence and reduced communication costs in the early training stages [22].

While FedSGD requires less computational effort due to its single local iteration, FedAvg effectively reduces the communication iterations required for training by a factor of E. This reduction significantly lowers the communication demands, particularly in decentralised data scenarios involving deep neural networks (DNNs) [6].

Several features distinguish FL from other parallel optimisation methods [23]. Notably, in extensive networks generating large data volumes, the connection speed between the central server and devices may slow, leading to delayed communications crucial in latency-

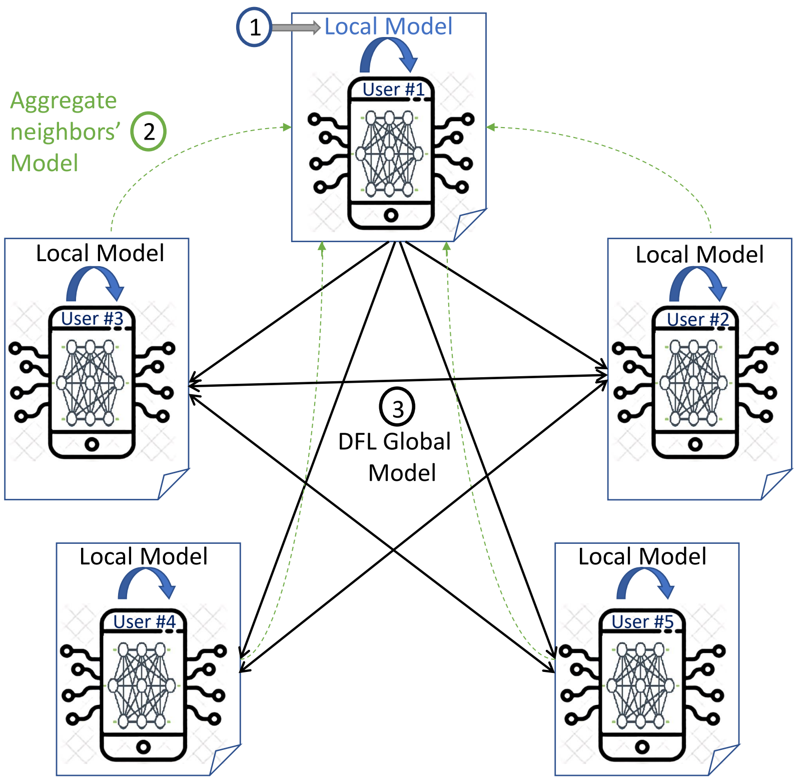


Figure 1.3: Decentralised federated learning layout [7]

sensitive applications such as intelligent healthcare systems [24]. Hence, there is a pressing need to develop communication-efficient FL approaches and overcome the bottleneck issue in the FL approach. These issues will be elaborated upon in the following section of this thesis.

1.1.3 Decentralised Federated Learning (DFL)

The DFL framework operates without a central server and involves only terminal participants (nodes) [15], as illustrated in Fig. 1.3. This setup facilitates parameter exchanges directly between peers. Additionally, the FedAvg algorithm or any other global aggregator algorithms can be used at each node to develop global models independently in the DFL system. Every participant refines its model locally based on its data and then integrates this with models from neighbouring nodes using the global aggregator algorithm. This collective model is then shared back to the peers in subsequent iterations until convergence is achieved.

In [25], the Combo algorithm introduces a technique where only a segment of the model parameters is shared and averaged, effectively reducing communication bandwidth demands without compromising system performance or convergence speed.

Although DFL addresses several limitations of conventional FL systems (e.g., CFL) requiring a central server, it primarily uses local model averaging for model integration on the client side, which may not be efficient in scenarios with heterogeneous data. Here, each local update pushes the model towards an individual optimum and averaging across diverse client models might lead to suboptimal global solutions, especially when training data varies significantly across clients [26]–[28].

Moreover, prior research often overlooks the nuances of wireless communication environments, neglecting factors like channel fading, link blockages, and interference. Commonly, participant selection does not account for the limitations imposed by the communication medium, which can affect the reliability and adaptability of real-time scenarios such as autonomous vehicles and UAV networks.

1.2 Challenges

FL is distinguished by unique challenges not typically encountered in broader distributed learning contexts. This section outlines the main principal challenges that uniquely characterise the FL environment, extending beyond general distributed optimisation and privacy-focused data analytics:

1.2.1 Communication Bottleneck

The primary challenge in FL is the high cost of communication, often emerging as a significant bottleneck. Unlike intra-data or inter-data centre communications that benefit from high speed and bandwidth, end-user connections, such as wireless links used in FL, are generally slower, more costly, and less reliable, particularly in remote locations. Consequently, the communication between clients and the central server may significantly lag behind the speed of local computations at times [29]. Moreover, while FL networks, especially in cross-device settings, may comprise millions of clients, the system's architecture and the server's capacity usually limit the number of clients that can actively participate in each communication round. To efficiently accommodate models on decentralised devices within such networks, the development of methods to enhance communication efficiency is crucial. Literature suggests several strategies to mitigate this challenge, including:

i. Communication compression: This involves reducing the size of the data transmitted in each communication round.

ii. Enhanced efficiency methods: These strategies aim to decrease the total number of necessary communication rounds.

iii. Client weights: This technique scales the clients' contributions to the learning process in each communication round.

1.2.2 Device Distribution and Systems Heterogeneity

Unlike traditional cloud-based distributed training, which leverages high-end computational architectures [30], FL operates under the constraints of data locality, with clients often being resource-limited, heterogeneous, and unreliable devices. This unique deployment scenario of FL is significantly challenged by the extensive diversity in client hardware, which can drastically vary in processing capabilities and reliability [31]–[33]. For instance, hardware heterogeneity may lead to notable disparities in processing speeds [34], potentially slowing down the aggregation process as the server waits for slower, less capable devices, termed stragglers. Furthermore, devices with modest memory and processing power might struggle to handle larger models, leading to their exclusion from the FL process due to capacity limitations or timeouts, thus discarding their unique data, which could be valuable. Intriguingly, the technical capabilities of devices might also reflect the demographic and socio-economic status of their owners, rendering the exclusion of such devices not only a technical decision but also a socio-economic concern [35].

The training process in FL is further complicated by challenges in both downstream (model) and upstream (update) network communications, which can severely bottleneck the training procedure [36]. Consequently, FL methods must be designed to be resilient to:

- The heterogeneity of devices, including variations in network connection speed, bandwidth, storage capacity, compute power, and battery life.
- The partial participation of clients in the learning process.
- The presence of stragglers and the potential for device dropout.

1.2.3 Network Security and Reliability

FL for dynamic networks grapples with significant robustness challenges, notably vulnerability to Byzantine attacks during the model training phase. Byzantine nodes, characterised by their potentially malicious behaviour, can lead to severe consequences such as model corruption or targeted disruptions of the learning process. Although research on Byzantine-resilient FL is advancing, decentralised implementations often receive less focus, heightening the risk in scenarios where network integrity is critical.

This expanded version incorporates a broader explanation of the challenges and strategic responses to Byzantine attacks, particularly in DFL, emphasising the importance of network integrity in safety-critical applications, such as autonomous vehicles.

1.3 Contributions

This thesis aims to introduce several original models using various communication techniques, such as resource optimisation, clustering and multi-paths protocol, which are highlighted in the Chapters [2-5], that accommodate decentralised setups and heterogeneous data. It will also thoroughly evaluate the communication dynamics among participants, considering the potential interference and noise that may affect transmissions, to realistically simulate communication links during the learning process. Furthermore, considering the significant challenges posed by adversarial attacks on the learning process, particularly within DFL frameworks, there is a pressing need for the development of robust and dependable defences. Accordingly, these challenges and the proposed solutions will be discussed in Chapter 4.

The specific contributions of this PhD thesis are summarized in Table 1.1 below and detailed in the following descriptions:

Chapter no.	Key Contribution	Comments	Published Work
Chapter 2	Geometric analysis for DFL network and optimisation of communication during the learning process using Slotted-ALOHA.	Simulate the dynamic of clients and consider real-world communication constraints to define the actual number of clients for each iteration.	[7]
Chapter 3	Further optimisation of DFL learning by implementing various aggregation methods and minimising the communication overhead using an efficient compression approach for the local algorithm.	Improves the global model and increases robustness, particularly in non-iid client scenarios. The implemented compression mechanism significantly reduces the local model size without affecting the learning performance.	[37]
Chapter 4	Enhancing Byzantine Resilience in DFL network, particularly in environments with dynamic clients.	Implements a novel aggregation policy designed to mitigate and neutralize the adverse effects of potential poisoning attacks.	Under review
Chapter 5	Integration of a model-based Multipath TCP (MPTCP) in the DFL framework to improve communication reliability during the learning process.	Effectively uses the available communication resources by maximising the utilisation of available MPTCP subflows.	Under review

Table 1.1: Summary of the specific contributions of this PhD thesis.

Chapter 2, Decentralised Federated Learning over Slotted ALOHA Wireless Mesh Networking [7], introduces a proposal design of the DFL framework, which operates flexibly without a central server and instead uses direct collaboration among one-hop neighbours. This collaboration is sensitive to the dynamics of communication networks, including network topology, MAC protocols, and both large-scale and small-scale fading on links. I apply stochastic geometry to explicitly model these dynamics, enabling precise quantification of the DFL's performance. The primary goal is to evaluate the impact of the communication between devices during the learning process and then enhance it to improve the classification accuracy of the model without compromising privacy and effectively managing network dynamics.

Chapter 3, Decentralised Federated Learning on the Edge over Wireless Mesh Networks [37], where I introduced a further optimisation of DFL performance by implementing various aggregation methods and minimising the communication overhead using an efficient compression approach for the local algorithm. System simulations are conducted to evaluate the proposed decentralised architecture under a variety of network parameters and different aggregation methods such as FedAvg, Krum, and Median methods. The model is trained on the widely recognised EMNIST dataset for benchmarking handwritten digit classification and incorporates a state-of-the-art compression technique based on genetic algorithms. This approach significantly reduces the size of the models shared over the wireless channel, compressing participants' local model sizes to nearly half of their original size relative to baselines. This effectively diminishes complexity and communication overhead.

Chapter 4, Enhancing Byzantine-Resilience in Decentralised Federated Learning for Dynamic Networks [38], under review as of May 2024, in which this chapter tackles the challenge of enhancing DFL performance, particularly in environments with Byzantine devices. The goal is to develop a robust model for ad hoc networks that leverage the CSMA/CA protocol for secure data transmissions. The proposed DFL model, which is Byzantine-resilient and designed for wireless networks, identifies and eliminates untrustworthy devices using a novel DFL reputation scoring system. It also incorporates advanced aggregation methods and spatial analysis to sustain throughput. I assess the DFL network's performance through two critical metrics: 1) robustness against adversarial attacks and 2) high accuracy and low loss, considering both complexity and network dynamics. Therefore, Chapter 4 is assigned to address the susceptibility of DFL networks to adversary attacks by devising a comprehensive adversary optimisation strategy aimed at enhancing network security and reliability. This approach includes:

(i) Implementing advanced anomaly detection techniques to identify and isolate adversary (e.g., Byzantine) nodes or clients effectively.

(ii) Deploying mitigation strategies that leverage device-weighted assessments and reputation scoring mechanisms alongside robust aggregation methods to reduce the influence of compromised nodes on the overall model.

(iii) Ensuring the maintenance of high model accuracy and overall network performance despite the presence of Byzantine faults, thus safeguarding the integrity of the learning process.

Further, this work explores the integration of these strategies within dynamic networks, where rapid and reliable decision-making is paramount. I aim to demonstrate that a well-protected DFL framework can significantly enhance the trustworthiness and efficacy of machine learning deployments in safety-critical environments such as autonomous driving.

Chapter 5, Enhancing Reliability and Efficiency in Collaborative Learning through Decentralised Federated Learning using MPTCP [39], under review as of May 2024, which advances DFL effectiveness by improving communication reliability during the learning process. With the increasing integration of Multipath TCP (MPTCP) in devices, including those from Apple, I propose its combination with DFL to enhance throughput, reduce latency, and achieve superior collaborative learning outcomes suited for real-world applications. I introduce three unique DFL network topologies: Simultaneous Subflows Utilisation, Strategic Subflow Allocation, and Master Subflow Priority. This study presents a novel approach for dynamically selecting the optimal DFL network topology, adopting Reinforcement Learning algorithms to identify the most effective network configuration for managing data flow across available subflows with an MPTCP-based model. Through comprehensive system simulations, I evaluate the proposed decentralised framework against contemporary FL models, demonstrating its potential in real-world scenarios. Thus, further advances in the DFL network are introduced by enhancing communication reliability throughout the learning process, focusing on the potential to enhance data transmission efficiency and reduce latency in DFL environments. The discussion delves into how MPTCP's capability to manage multiple data paths can significantly improve the robustness and speed of model training in DFL setups.

1.4 Summary

In summary, the objective was (and continues to be) to develop a robust DFL framework that enables devices to collaborate securely and privately with an efficient and adaptable wireless communication design resilient to adversarial and Byzantine attacks. This innovative topic has afforded me the opportunity to publish several papers during my PhD, resulting in a thesis by publication. The main body of the thesis (Chapters 2 - 5) comprises reformatted but otherwise verbatim reproductions of my published works—two journal papers [7][37] and one conference paper [40], along with two additional journal papers currently under review (i.e., Chapter 4 and Chapter 5).

List of Publication

- **A. Salama**, A. Stergioulis, A. M. Hayajneh, S. A. R. Zaidi, D. McLernon, and I. Robertson, “Decentralised Federated Learning over Slotted ALOHA Wireless Mesh Networking,” IEEE Access, vol. 11, pp. 18 326–18 342, 2023.

- **A. Salama**, A. Stergioulis, S. Zaidi, and D. McLernon, “Decentralised Federated Learning on the Edge over Wireless Mesh Networks,” IEEE Access, vol. 11, pp. 124 709–124 724, 2023.

- **A. Salama**, S. A. Zaidi, D. McLernon, and M. M. Qazzaz, “FLCC: Efficient Distributed Federated Learning on IoMT over CSMA/CA,” in 2023 IEEE 97th Vehicular Technology Conference (VTC2023-Spring). IEEE, 2023, pp. 1–6.

- **A. Salama**, S. A. Zaidi, D. McLernon, and M. M. Qazzaz, “Enhancing Byzantine-Resilience in Decentralised Federated Learning for Dynamic Networks,” IEEE Transactions on Machine Learning in Communications and Networking, (Submitted in March, and still under review).

- **A. Salama**, S. A. Zaidi, D. McLernon, M. Hafeez, and M. M. Qazzaz, “Enhancing Reliability and Efficiency in Collaborative Learning through Decentralised Federated Learning using MPTCP,” IEEE Open Journal of the Communications Society (Submitted in April, and still under review).

References

[1] W. Yu, F. Liang, X. He, *et al.*, "A survey on the edge computing for the internet of things," *IEEE access*, vol. 6, pp. 6900–6919, 2017.

[2] G. Premsankar, M. Di Francesco, and T. Taleb, "Edge computing for the internet of things: A case study," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 1275–1284, 2018.

[3] W. Y. B. Lim, N. C. Luong, D. T. Hoang, *et al.*, "Federated learning in mobile edge networks: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 3, pp. 2031–2063, 2020.

[4] J. Bian, A. Al Arafat, H. Xiong, *et al.*, "Machine learning in real-time internet of things (iot) systems: A survey," *IEEE Internet of Things Journal*, vol. 9, no. 11, pp. 8364–8386, 2022.

[5] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*, PMLR, 2017, pp. 1273–1282.

[6] J. Konečnỳ, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," *arXiv preprint arXiv:1610.05492*, 2016.

[7] A. Salama, A. Stergioulis, A. M. Hayajneh, S. A. R. Zaidi, D. McLernon, and I. Robertson, "Decentralized federated learning over slotted aloha wireless mesh networking," *IEEE Access*, vol. 11, pp. 18 326–18 342, 2023.

[8] M. Aledhari, R. Razzak, R. M. Parizi, and F. Saeed, "Federated learning: A survey on enabling technologies, protocols, and applications," *IEEE Access*, vol. 8, pp. 140 699–140 725, 2020.

[9] Z. Eghbali and M. Z. Lighvan, "A hierarchical approach for accelerating iot data management process based on sdn principles," *Journal of Network and Computer Applications*, vol. 181, p. 103 027, 2021.

[10] U. Shafi, R. Mumtaz, J. García-Nieto, S. A. Hassan, S. A. R. Zaidi, and N. Iqbal, "Precision agriculture techniques and practices: From considerations to applications," *Sensors*, vol. 19, no. 17, p. 3796, 2019.

[11] M. Xin and Y. Wang, "Research on image classification model based on deep convolution neural network," *EURASIP Journal on Image and Video Processing*, vol. 2019, no. 1, pp. 1–11, 2019.

[12] S. Hu, X. Chen, W. Ni, E. Hossain, and X. Wang, "Distributed machine learning for wireless communication networks: Techniques, architectures, and applications," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 3, pp. 1458–1493, 2021.

[13] L. Lyu, H. Yu, and Q. Yang, "Threats to federated learning: A survey," *arXiv preprint arXiv:2003.02133*, 2020.

[14] L. U. Khan, W. Saad, Z. Han, and C. S. Hong, "Dispersed federated learning: Vision, taxonomy, and future directions," *IEEE Wireless Communications*, vol. 28, no. 5, pp. 192–198, 2021.

[15] H. Zhang, J. Bosch, and H. H. Olsson, "Federated learning systems: Architecture alternatives," in *2020 27th Asia-Pacific Software Engineering Conference (APSEC)*, IEEE, 2020, pp. 385–394.

[16] P. Vanhaesebrouck, A. Bellet, and M. Tommasi, "Decentralized collaborative learning of personalized models over networks," in *Artificial Intelligence and Statistics*, PMLR, 2017, pp. 509–517.

[17] A. Bellet, R. Guerraoui, M. Taziki, and M. Tommasi, "Personalized and private peer-to-peer machine learning," in *International conference on artificial intelligence and statistics*, PMLR, 2018, pp. 473–481.

[18] C. He, C. Tan, H. Tang, S. Qiu, and J. Liu, "Central server free federated learning over single-sided trust social networks," *arXiv preprint arXiv:1910.04956*, 2019.

[19] H. Wen, Y. Wu, J. Hu, Z. Wang, H. Duan, and G. Min, "Communication-efficient federated learning on non-iid data using two-step knowledge distillation," *IEEE Internet of Things Journal*, 2023.

[20] A. M. BENHANGI, "Communication-computation efficient federated learning over wireless networks," 2023.

[21] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of fedavg on non-iid data," *arXiv preprint arXiv:1907.02189*, 2019.

[22] J. Xu, H. Wang, and L. Chen, "Bandwidth allocation for multiple federated learning services in wireless edge networks," *IEEE transactions on wireless communications*, vol. 21, no. 4, pp. 2534–2546, 2021.

[23] J. Wen, Z. Zhang, Y. Lan, Z. Cui, J. Cai, and W. Zhang, "A survey on federated learning: Challenges and applications," *International Journal of Machine Learning and Cybernetics*, vol. 14, no. 2, pp. 513–535, 2023.

[24] J. Li, Y. Meng, L. Ma, *et al.*, "A federated learning based privacy-preserving smart healthcare system," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 3, 2021.

[25] C. Hu, J. Jiang, and Z. Wang, "Decentralized federated learning: A segmented gossip approach," *arXiv preprint arXiv:1908.07782*, 2019.

[26] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh, "Scaffold: Stochastic controlled averaging for federated learning," in *International Conference on Machine Learning*, PMLR, 2020, pp. 5132–5143.

[27] F. Sattler, S. Wiedemann, K.-R. Müller, and W. Samek, "Robust and communication-efficient federated learning from non-iid data," *IEEE transactions on neural networks and learning systems*, vol. 31, no. 9, pp. 3400–3413, 2019.

[28] N. Shoham, T. Avidor, A. Keren, *et al.*, "Overcoming forgetting in federated learning on non-iid data," *arXiv preprint arXiv:1910.07796*, 2019.

[29] O. R. A. Almanifi, C.-O. Chow, M.-L. Tham, J. H. Chuah, and J. Kanesan, "Communication and computation efficiency in federated learning: A survey," *Internet of Things*, vol. 22, p. 100 742, 2023.

[30] K. Hazelwood, S. Bird, D. Brooks, *et al.*, "Applied machine learning at facebook: A datacenter infrastructure perspective," in *2018 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, IEEE, 2018, pp. 620–629.

[31] C.-J. Wu, D. Brooks, K. Chen, *et al.*, "Machine learning at facebook: Understanding inference at the edge," in *2019 IEEE international symposium on high performance computer architecture (HPCA)*, IEEE, 2019, pp. 331–344.

[32] A. Ignatov, R. Timofte, A. Kulik, *et al.*, "Ai benchmark: All about deep learning on smartphones in 2019," in *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, IEEE, 2019, pp. 3617–3635.

[33] K. Bonawitz, H. Eichner, W. Grieskamp, *et al.*, "Towards federated learning at scale: System design," *Proceedings of machine learning and systems*, vol. 1, pp. 374–388, 2019.

[34] M. Almeida, S. Laskaridis, I. Leontiadis, S. I. Venieris, and N. D. Lane, "Embench: Quantifying performance variations of deep neural networks across modern commodity devices," in *The 3rd international workshop on deep learning for mobile systems and applications*, 2019, pp. 1–6.

[35] P. Kairouz, H. B. McMahan, B. Avent, *et al.*, "Advances and open problems in federated learning," *Foundations and trends® in machine learning*, vol. 14, no. 1–2, pp. 1–210, 2021.

[36] F. Sattler, K.-R. Müller, and W. Samek, "Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints," *IEEE transactions on neural networks and learning systems*, vol. 32, no. 8, pp. 3710–3722, 2020.

[37] A. Salama, A. Stergioulis, S. A. R. Zaidi, and D. McLernon, "Decentralized federated learning on the edge over wireless mesh networks," *IEEE Access*, vol. 11, pp. 124 709–124 724, 2023.

[38] A. Salama, S. A. R. Zaidi, and D. McLernon, "Enhancing byzantine-resilience in decentralised federated learning for dynamic networks," *IEEE Transactions on Machine Learning in Communications and Networking*, vol. under review, 2024.

[39] A. Salama, S. A. R. Zaidi, and D. McLernon, "Enhancing reliability and efficiency in collaborative learning through decentralised federated learning using mptcp," *IEEE Open Journal of the Communications Society*, vol. under review, 2024.

[40] A. Salama, S. A. Zaidi, D. McLernon, and M. M. Qazzaz, "Flcc: Efficient distributed federated learning on iomt over csma/ca," *arXiv preprint arXiv:2304.13549*, 2023.

Chapter 2

Decentralised Federated Learning over Slotted ALOHA Wireless Mesh Networking

Chapter source: A. Salama, A. Stergioulis, A. M. Hayajneh, S. A. R. Zaidi, D. McLernon, and I. Robertson, "Decentralised federated learning over slotted aloha wireless mesh networking," IEEE Access, vol. 11, pp. 18 326–18 342, 2023.

Abstract

Federated Learning (FL) presents a mechanism to allow decentralised training for machine learning (ML) models inherently enabling privacy preservation. The classical FL is implemented as a client-server system, which is known as Centralised Federated Learning (CFL). There are challenges inherent in CFL since all participants need to interact with a central server resulting in a potential communication bottleneck and a single point of failure. In addition, it is difficult to have a central server in some scenarios due to the implementation cost and complexity. This study aims to use Decentralised Federated learning (DFL) without a central server through one-hop neighbours. Such collaboration depends on the dynamics of communication networks, e.g., the topology of the network, the MAC protocol, and both large-scale and small-scale fading on links. In this paper, we employ stochastic geometry to model these dynamics explicitly, allowing us to quantify the performance of the DFL. The core objective is to achieve better classification without sacrificing privacy while accommodating for networking dynamics. In this paper, we are interested in how such topologies impact the performance of ML when deployed in practice. The proposed system is trained on a well-known MINST dataset for benchmarking, which contains labelled data samples of 60K images each with a size 28×28 pixels, and 1000 random samples of this MNIST dataset are assigned for each participant. The participants implement a CNN model as a classifier model. To evaluate the performance of the model, a number of participants are randomly selected from the network. Due to randomness in the communication process, these participants interact with the random number of nodes in the neighbourhood to exchange model parameters which are subsequently used to update the participants' individual models. These participants connected successfully with a varying number of neighbours to exchange parameters and

update their global models. The results show that the classification prediction system was able to achieve higher than 95% accuracy using the three different model optimisers in the training settings (i.e., SGD, ADAM, and RMSprop optimisers). Consequently, the DFL over mesh networking shows more flexibility in IoT systems, which reduces the communication cost and increases the convergence speed which can outperform CFL.

2.1 Introduction

In recent years, the number of Internet of Things (IoT) and smart wearable devices have witnessed an increased proliferation in several vertical domains (e.g., wearables, home automation systems, smart glasses, health monitors, health-fitness trackers, smart grids, etc). These devices use a variety of wireless technologies and thus manifest different topological properties. For instance, LoRa deployment support one hop connection to the gateway with a distance-dependent spreading factor manifesting star topology. In contrast, Thread, Zigbee, and BLE Mesh manifest the mesh topology. The Mesh deployment for local connectivity is gaining extra momentum with the rapid evolution of standards.

The huge amount of data that sensors and edge devices collect is critical. It can be gathered and analysed using modern ML techniques to build a classification model that can learn, predict, and meet the end user's requirements optimally. ML algorithms can enable various applications. These include controlling and monitoring the home appliance, controlling autonomous vehicles, and monitoring various health parameters of the elderly such as heart rate, fall detection, etc. However, the data from an edge device may carry very personal information. Thus, data privacy and security are significant challenges as users usually do not allow sharing of sensitive information and data by putting it all in one central location.

Now, Federated Learning (FL) [1] has recently emerged to address this issue and solve participants' essential requirements and concerns to preserve privacy and data security. FL has developed as a new paradigm in building models from distributed ML setups that

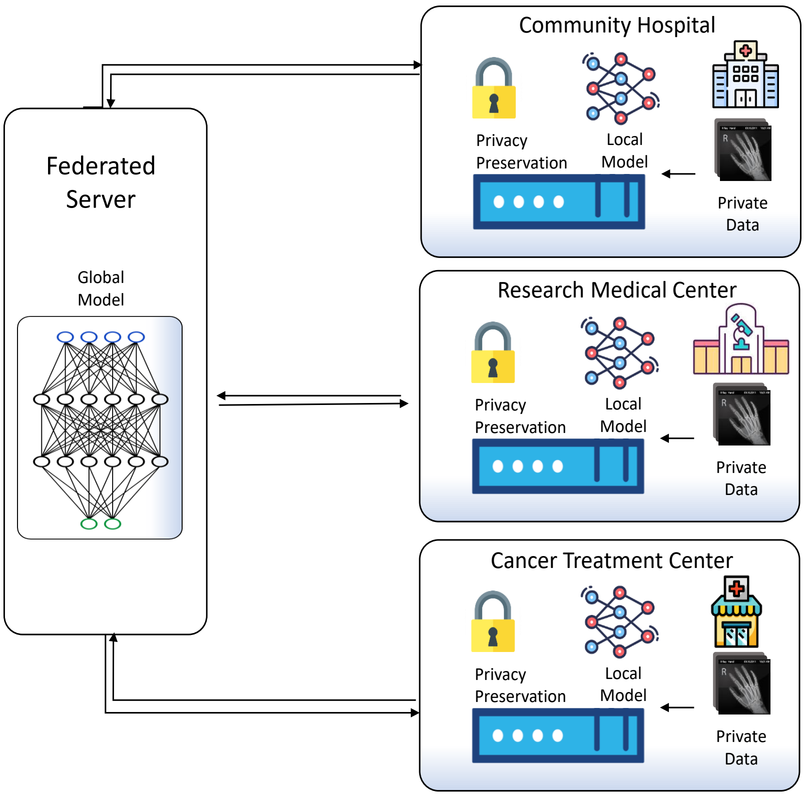


Figure 2.1: A federated learning approach

can offer the opportunity to learn a model from multiple disjoint sensitive local datasets while keeping the user data private through distributed training [1] and [2].

FL has attracted much attention because of its ability to preserve the privacy of the client's data by sharing only the locally trained model parameters instead of the local data itself. Figure 2.1 shows a reference deployment scenario for a healthcare application. This example shows how the Community Hospital, the Research Medical Centre, and the Cancer Treatment Centre all train their local models using their local private data and share only the model parameters to create a global model in order to improve the system performance.

Thus, FL performs the aggregation and analysis of the local models and updates the models on the participants' devices or servers without sharing any devices' data with others, thus keeping the private data protected. So, each participating device will train a model exploiting its local data usually using classification or regression algorithms (i.e., employing a Deep Neural Network (DNN)), and only share the model parameters with a central server. Afterwards, the FL central server will aggregate the parameters of these local models to process and create a global model. The FL server broadcasts the global model to update the local models. The system will iterate the same procedure until the system achieves a convergence state [3]. This paradigm of FL is also known as Centralised Federated Learning (CFL).

2.1.1 Motivation

The data generated by IoT devices and smartphone terminals has become essential for driving intelligent services through applications of Machine Learning (ML). Various applications ranging from healthcare to autonomous vehicles are rapidly deploying IoT solutions. It is thought that FL can offer more secure and shared security services for a wider range of applications, helping to support the steady growth of distributed ML applications [4]. Although the CFL systems are indeed promising, they face several limitations and challenges as they require a central FL server. Efficient communication is a critical challenge that needs to be addressed in FL to ensure that all participating devices are connected and that performance is not compromised [5]. Furthermore, in particular, it is hard to implement in some scenarios where a reliable and robust central server is difficult to find (e.g., a fully self-driving and autonomous network). Moreover, the CFL network faces the limitation of a single point of failure (communication bottleneck) at the central server. Therefore, this research will study a Decentralised Federated Learning (DFL) approach to overcome these challenges. In the DFL model, all neighbouring devices share their model parameters directly in a peer-to-peer manner. In this case, each device acts as a server by aggregating the parameters from neighbours and then averaging them to update a sub-global model that can be shared with others within one hop of a communication link.

The aim of this paper is to evaluate the feasibility of the implementation of the DFL approach under spatio-temporal dynamics. DFL algorithms are implemented using clusters of mesh networking groups located in different environments. Our system model can be translated into two practical deployment modalities: 1) Autonomous and Decentralised IoT networks: these are the networks formed by limited capability IoT and edge devices where decentralised FL is required intrinsically. For instance, networks formed by wearable IoT devices in battlefields or remote hospitals. 2) Edge/Fog-Assisted IoT networks: these are the hierarchical networks where the data from IoT devices is collected and processed by the edge/fog gateways [6]. These gateways then form a mesh network where DFL must be implemented. A typical example of such deployment is in agricultural IoT setup [7] or other smart city IoT applications. Both cases can be translated into the system model considered in this paper. In addition, this study will implement an FL network based on the slotted ALOHA protocol as a sub-optimal MAC protocol to evaluate the performance of the network and optimal configuration under the proposed setup.

The combination of the DFL with the slotted ALOHA mesh networking protocol is proposed to satisfy the users' privacy preservation, increase the protection for confidential data, increase the prediction accuracy of the implemented algorithms and build a robust system. This system can help us exploit a greater percentage of the users' data that will be trained to create a global model that can improve the local models in each participating device without sharing any data with other participants except the local model parameters.

2.1.2 Key Contributions

Limited studies were conducted using FL with a central server to achieve better results. The objectives of this study are to introduce and stimulate the Centralised and Decentralised FL's wireless communication stage between the devices in the learning process based on qualitative examinations of the CFL approach with a central server and the DFL approach over a Wireless Mesh Networking (WMN) without a central server. Furthermore, this study aims to simulate the communication network for the CFL and DFL models to design a robust wireless communication network during the training process, which helps to evaluate how the model can perform in different network conditions, such as congestion and interference.

2.1.3 Paper Organisation

The rest of the paper is organised as follows. In Section 2.2, the related work on the FL approach is provided. Background and challenges for DFL over WMN are presented in Sec. 2.3. Section 2.4, introduces the system model in terms of theoretical analysis of device communication in the network and learning metrics for DFL over WMN. The learning criteria for DFL are demonstrated in Sec. 2.5. The proposed framework simulation and results are summarised in Sec. 2.6. Finally, a summary of this research and future work is presented in Sec. 2.7.

2.2 Related Work

Many recent papers have investigated the fundamentals of CFL algorithms [8]–[11] and [12]. CFL and its central server algorithm are called the Federated averaging (FedAvg) algorithm and were first proposed and implemented in [1]. The FedAvg algorithm is implemented to create a global model by averaging the aggregated parameters from the participants [1].

In [8], comprehensive research of CFL for mobile-edge networks was presented. The authors examined the critical implementation issues with existing solutions and potential applications of CFL in IoT and mobile edge networks. In addition, some existing limitations and challenges in CFL are highlighted, such as the difficulty of aggregating sufficient data, real applications' heterogeneous data distribution, and theoretical analysis of device communication and convergence. The work [9] reviewed the challenges in implementing CFL, future research directions and the existing CFL approaches. In [10] and [12] the authors surveyed the CFL implementations, devised a taxonomy, and overviewed the currently proposed solutions and their challenges in the CFL framework. They presented the essentials of preserving privacy and checking fairness in CFL.

The study in [13] conducted a thorough and comprehensive examination of the architecture, design, and deployment of FL, comparing it to centralised and distributed on-site

ML-based systems. Furthermore, the challenges and potential future directions for research in FL were discussed, where some classification problems of FL topics and research fields were also presented, based on a thorough literature review, including taxonomies for its important technical and emerging aspects, such as the core system model and design, application areas, privacy and security, and resource management.

The authors in [11] examined the “In-Edge-AI” model for edge networks to allow for efficient collaboration between terminal devices and terminal servers to exchange learning model parameter updates. They explored two use scenarios: edge caching and compute offloading. To efficiently support these scenarios, they trained a double deep Q-learning (DDQN) model via CFL. Lastly, the authors in [8]–[11] and [12] addressed several existing issues in CFL for actual applications, such as the ability of mobile devices to handle a high computation process and the power consumption and battery life to keep connected to a central server. Furthermore, CFL raises concerns about flexibility because it may cease to function due to the aggregation server’s failure (i.e., due to a malicious assault or physical flaw). Moreover, training CFL models via IoT networks necessitates many communication resources to allow participants to communicate with a central server [14].

Most existing DFL systems are based on gossiping schemes, and the number of neighbours in the learning process are chosen regardless of communication challenges, end-user capability and network capacity. For instance, the works in [15] and [16] implement a classic DFL algorithm that allows a user to aggregate the model parameters from an estimated number of multiple neighbouring devices, and Ramanan et al. [17] propose an alternative approach that uses a blockchain-based FL scheme to aggregate updates for the participants’ devices. However, these approaches suffer from several limitations related to the communication constraints in the real environment applications, the data size and the terminal capability (i.e., energy consumption and computation cost) of blockchain-based transactions. In summary, we list some existing works on FL-related topics with our paper’s contribution in Table 2.1.

Table 2.1: Existing works on FL-related topics with our paper's contribution.

Related works	Key topic	Evaluate recent advances design of the network in FL approaches					
		Resource management and system cost	Security and privacy	Users' distribution analysis	Comms. in the network	FL Network intensity and performance	Central Server-free
[22]	DFL concept	×	✓	×	✓	×	✓
[23]	FL concept	×	✓	×	✓	×	✓
[24]	Distributed ML	✓	×	×	✓	×	×
[25]	Security and privacy in FL	×	✓	×	×	×	✓
[8]	FL in edge networks	✓	✓	×	×	×	×
[26]	FL for IoT	✓	✓	×	✓	×	×
[27]	FL for IIoT	✓	✓	×	×	×	×
[28]	FL for health informatics	✓	✓	×	✓	×	×
[29]	Decentralised Wireless FL	×	✓	×	✓	×	✓
[30]	DFL framework	✓	✓	×	×	×	✓
[31]	Blockchain-based FL	✓	✓	×	×	×	✓
This work	DFL over WMN	✓	✓	✓	✓	✓	✓

2.3 Background and Challenges

The implementation of the DFL approach over a WMN is supposed to reduce communication costs, cope with the single point of failure issue in CFL, and provide innovative capabilities in a range of aspects, including healthcare systems (e.g., monitoring physiological data like heart-rate variability [18] to classify various cardiac pathologies), industry, and smart homes [19]–[21].

In addition, the combination of DFL and WMN using mesh protocols will likely be helpful in preserving privacy, guaranteeing a robust network, and improving the battery life of the devices by reducing communication costs. Instead of transferring data from the device to the central server provider, which will need large bandwidth and consume high power on the edge device, the decentralised approach can be applied to minimise these back-and-forth journeys of data. This decentralised fashion is implemented by processing the data into the edge device and communicating with the other neighbours using the mesh networking links to exchange and update the model parameters.

This paper will simulate and implement the DFL model over WMN system protocols. The model and the system performance will be evaluated by training the model using a dataset divided into training data, validation data, and test data. The performance metrics of the algorithms will be prediction accuracy, communication cost, and latency. Both DFL and WMN protocols are implemented in some applications separately and individually.

This research proposes integrating DFL and WMN into one intelligent system to optimise for robust communication networks that could be applied in many IoT applications to ensure participant privacy preservation and data security. Although DFL and WMN have impressive characteristics and features, they reveal several challenges and problems faced by engineers that can influence the model's accuracy. The following is a short summary of those limitations:

I. The network will be designed for low-power IoT devices under IEEE 802.15.4-2006 and WMN would need to be designed to coexist in IoT systems with other technologies, and not to replace them [32].

II. Convergence speed: DFL algorithms usually adopt a peer-to-peer one-layer architecture. Each participant collects and aggregates all the local model updates of one-hop neighbours in a CFL architecture. With such multi-hop architecture, the wireless routing paths between participants can be easily saturated, resulting in a slower convergence speed [33].

III. Unbalance: the amount of data varies at each participant resulting in different local training data quality.

IV. Lack of stability and flexibility in communication networks of a massive number of devices in real-time applications.

The communication stage is one of four main steps in the learning process that cannot be neglected, and most researchers do not consider it analytically in their research. In this paper, this stage will be addressed in detail. We propose using mesh networking to maximise the communication stage's flexibility and the channel's capacity during the learning process. The motivation for this is the fact that the DFL algorithms can efficiently update the terminal edge with the parameters through the Thread protocol or any other mesh networking protocols (e.g., ZigBee or Bluetooth). This will allow us to design and develop a global model that can precisely analyse the end-users data without sharing the data with a central server or any other devices within the network. In other words, the data stays protected locally and never leaves the device itself and this will achieve personalisation and guarantee high Quality of Service (QoS) as well as enhance the performance of devices in IoT applications.

To the best of our knowledge, this research is the first work that combines DFL and WMN using the slotted ALOHA protocol. The model results verify the intuition, showing that implementing DFL over mesh networks can offer more flexibility as no central server is required and promises more communication channels available to communicate. More participants can be involved in the learning process in the form of neighbour groups. The rest of this paper will introduce the wireless communication characteristics for the mesh networking and DFL criteria. The wireless communication constraints will be considered and CFL and DFL models will be implemented by simulations.

2.3.1 Traditional Machine Learning (ML) on Edge Devices

There are different kinds of ML and deep learning algorithms used in various proposals. For instance, Convolution Neural Network (CNN) algorithms are powerful tools widely used in image classification processing and other classification problems [34] since CNN has a proven ability to achieve higher accuracy, and can efficiently learn from thousands of image datasets. To implement CFL and DFL algorithms, the local ML algorithms are required to be embedded in the terminal devices (participants) to train the algorithm on the local data before sharing the parameters with the server in a CFL approach or with the neighbours in a DFL approach. Details on ML-enabled edge devices challenges and opportunities have been addressed in many research papers in the recent past [35], [36] and they are out of the scope of this paper. In this research, the CFL and DFL models will be implemented for a classification problem, and the CNN algorithm will be the main algorithm that is used to train the local models on the proposed system.

2.3.2 Centralised Federated Learning (CFL)

The main objective of CFL systems is to train, in coordination with a central server for model aggregation, a shared global model from participating devices that act as local learners. Figure 2.2 shows the fundamental CFL architecture and the main four steps to train a CFL network, where these steps are iterated until reaching the convergence status [12]:

I. Local learning where each edge device uses its local dataset to train the model locally and update the parameters of the ML model (e.g., neural biases and weights).

II. Upload model parameters to the central server: participants upload (transmit) their parameters to the central server via communication channels.

III. Global aggregation: the central server aggregates the local models' parameters from those successfully received to update a new version of the global model on the server.

IV. Download (broadcast) and synchronise the devices with the latest global model updates [14], and then back to step 1 and repeat until system convergences.

In the CFL process, each device's local algorithm has an optimisation technique to update the model iteratively, such as Stochastic Gradient Descent (SGD). Afterwards, the global model emerges from aggregating the local parameters from participants' devices, which can then be weighted according to the perceived quality of the updates of the devices [34]. One crucial property of CFL is that the participant user data never transfers between devices and the server, which reduces communication costs and data sharing privacy concerns. However, due to the central server node, the CFL system will run into scalability issues. Even if the server node's hardware and software capabilities have been optimised, the server node's performance will not improve when thousands of client nodes join [37]. Communication bottlenecks may appear due to the amount of traffic that is increasing exponentially, and the system becomes overburdened. Furthermore, accessing a central node cannot be possible in some scenarios, for instance, self-driving vehicles and high mobility sensor systems. Decentralised architectures have recently been proposed to avoid

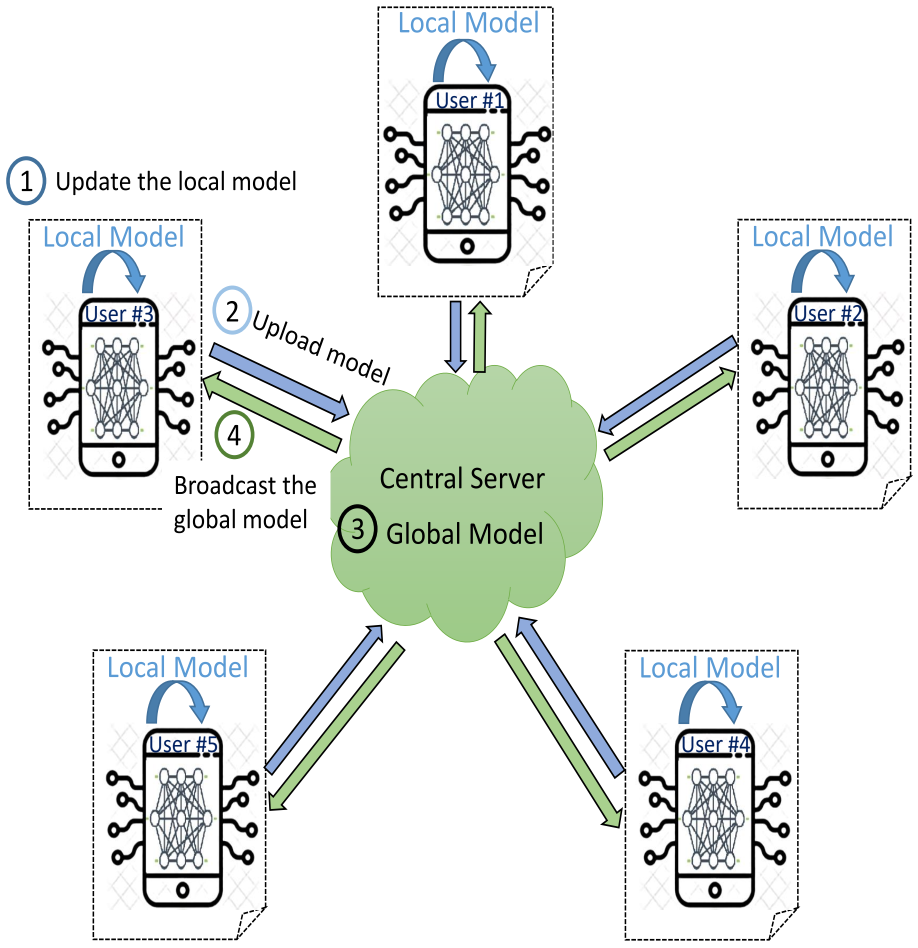


Figure 2.2: CFL framework over an IoT network.

communication bottlenecks and protect data privacy [38], [39]. Remove the centralised server, and each participant only communicates with its one-hop neighbours in its local area and exchanges their local models and updates parameters [40].

2.3.3 Decentralised Federated Learning (DFL)

As shown in Figure 2.3, the DFL framework does not need a central server to coordinate the training tasks and contains only terminal participants (nodes) [37]. The idea is that each participant exchanges parameter updates with neighbours in a peer-to-peer manner. Besides the local model algorithm, the FedAvg algorithm is employed on each terminal participant to create its global models in the DFL approach with no central server. Each participant trains on its local data and averages it within the aggregated models' parameters from the selected neighbours using the FedAvg algorithm to broadcast

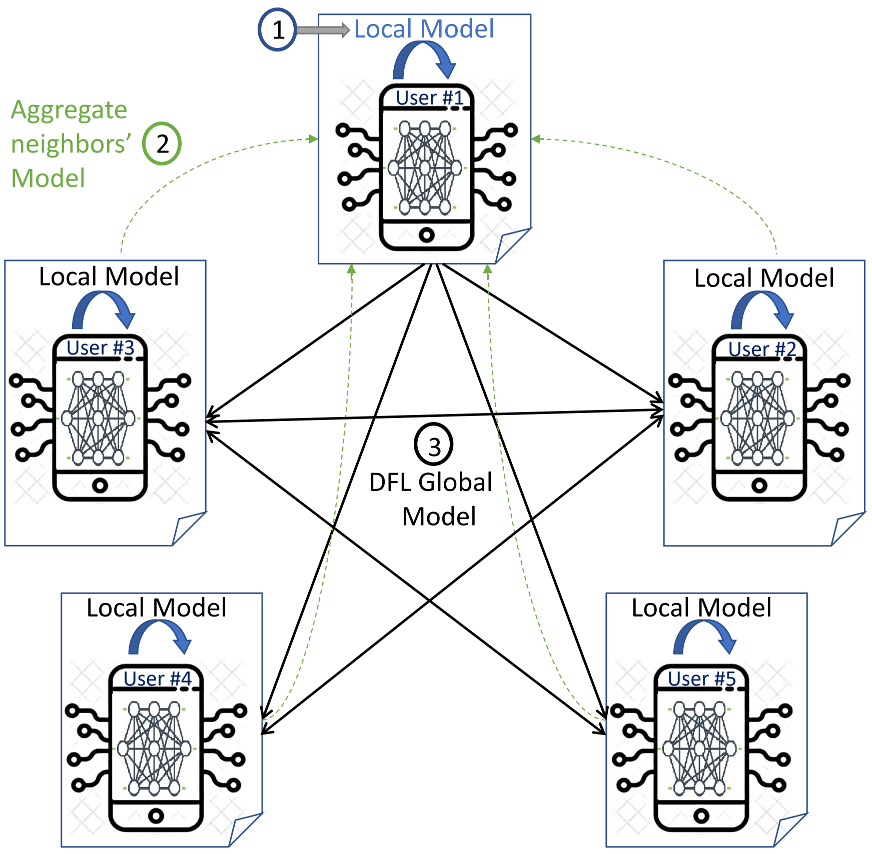


Figure 2.3: DFL framework over an IoT WMN.

an updated global model to the neighbours again at each iteration. Afterwards, the same procedure is applied to all other participants until the system converges.

In [41], the Combo algorithm proposed an approach in which the participants send and average a segment of the models' parameters to reduce the required communication bandwidth without affecting the system performance and convergence rate. Even though the proposed DFL algorithms overcome some of the problems associated with general FL systems that require a central server, they use model averages to fuse models at the local clients, which is not always very efficient in heterogeneous data scenarios. For each, local model parameters are updated toward the local optimum, and averaging the model parameters from different clients leads to the averaged outcome of each client's local optimum being used. The optimum of each participant's loss function may be quite far from the others, which is also far from the global optimum due to different participants owning different sets of training data. Thus, those datasets typically have different distributions or even no overlap, which is defined as data heterogeneity [42]–[44].

Furthermore, most prior works do not consider the wireless environment in the communication network, so they do not account for wireless impairments caused by channel fading, link blockages, and wireless interference. Most researchers choose the number of participants by estimation without concerning the communication medium constraints, which is not always very efficient in terms of reliability and flexibility for real-time applications (e.g., AV and UAV networks).

Therefore, this study will focus on designing a model based on the gossipy method [45] that can deal with decentralised approaches and heterogeneous datasets, and it will precisely analyse the communication process between the participants in the network, considering the interference and the noise in the transmission medium to simulate a real scenario of the communication connection between the terminal devices through the learning process.

2.3.4 Wireless Mesh Networking (WMN)

Nowadays, many access points have overlapping areas, and almost every traditional wireless network has to be connected to the wired network. In this scenario, the cost of installing IoT devices is costly and extremely difficult. Thus, a WMN will benefit from its flexibility to connect the devices within the network and offer a different perspective than non-mesh networks. The connectivity needs in wireless mesh networks are reduced mainly because the devices within the network have a multi-route capability to send and receive packets. In addition, the WMN also has a range of advantages, such as self-healing and self-organising, attracting a vast number of investigations and research developments [46]. Furthermore, WMN can reduce the networking cost for innovative home applications using low-profile hardware.

The routing protocols significantly contribute to WMNs as they help find the best path between multi-hop networks in unreliable wireless media. The WMN protocols have been widely investigated to achieve higher throughput, low latency and low power consumption [46], [47]. According to some related research, ZigBee, Bluetooth Low Energy BLE, Z-wave, and Thread protocols are the most common protocols used in many wireless mesh networking applications. Each protocol has its own unique specification for particular implementation depending on the user requirement. For instance, the Z-wave protocol is advantageous for long-range coverage because of its low-frequency band (900 MHz) compared to others with a 2.4 GHz band (i.e., Bluetooth and Thread). According to [48], all protocols achieve similar performance (i.e., latency and throughput) for small networks and small payloads. By contrast, for large mesh networks with multi-hop nodes between the transmitter and the receiver, the Thread protocol achieves better performance metrics in terms of latency and efficiency.

2.4 System Model

2.4.1 Communication in WMN

In this paper, a typical receiver is considered that is connected to a corresponding desired transmitter. A Rayleigh fading channel is adopted for the small-scale path-loss model and complemented with a single slope large-scale path-loss. Hence, the received power at the typical receiver from the desired transmitter is $(P_k h_{ko} d_{ko}^{-\alpha})$ [49], where P_k is the signal transmit power from the desired transmitter device k, h_{ko} is the fading coefficient for the channel between device k to the target destination, d_{k0} is the distance between the desired transmitter k and the corresponding receiver and $(\alpha \geq 2)$ is the path-loss exponent.

The signal can be correctly decoded at the typical receiver if the corresponding SINR (Signal to Interference plus Noise Ratio) is higher than a certain threshold T_k. Therefore, the probability that SINR $\geq T_k$ is defined as:

$$P(\text{SINR} \geq T_k) = P\left(\frac{P_k h_{k0} d_{k0}^{-\alpha}}{\sum_{i\in\varphi} I_i + N_0} \geq T_k\right) \tag{2.1}$$

where I_i is the interference from the device i in the network $(I_i = P_i h_{i0} d_{i0}^{-\alpha} a_i)$, with $i = 1, 2, ..., M$, M is the total number of active devices within the desired receiver coverage area and $i \in \varphi$ where φ represents the number of all participant devices within the whole target area. Now a_i is a binary random variable that defines the state of the device, whether in a transmitting state or ready to receive such that $P_A =$Probability$\{a_i = 1\}$ represents the device is in transmitting state and $1 - P_A =$ Probability$\{a_i = 0\}$ represents the device is ready to receive and is not transmitting.

The proposed IoT network is assumed to have a small thermal noise power variance N_0 in comparison with the cumulative interference power (i.e., interference-limited network). Therefore, the noise effect in the network is negligible, and so Eq. (2.1) can be re-written as:

$$P(SIR \geq T_k) \cong P\left(\frac{P_k h_{k0} d_{k0}^{-\alpha}}{\sum_{i\in\varphi} I_i} \geq T_k\right) \tag{2.2}$$

$$\cong P\left(h_{k0} \geq \frac{T_k d_{k0}^{\alpha} \sum_{i\in\varphi} I_i}{P_k}\right). \tag{2.3}$$

Since the proposed IoT device network has Rayleigh fading channels, and for the sake of simplicity, the h_{k0} term is assumed to be an exponentially distributed random variable with a unit mean. The probability distribution function (PDF) of an exponential random variable X with unit mean can be expressed in the form of $f_X(x) = \exp(-x)$, and thus, Eq. (2.3) can be reformulated as:

$$P(SIR \geq T_k) \cong \mathbb{E}_{a_i}\left(\mathbb{E}_{h_{i0}}\Big(\int_{\frac{(T_k d_{k0}^{\alpha} \sum_{(i\in\varphi)} I_i)}{P_k}}^{+\infty} (e^{-x} dx)\Big)\right) \tag{2.4}$$

$$\cong \mathbb{E}_{a_i}\left(\mathbb{E}_{h_{i0}}\Big(\exp(-\frac{T_k d_{k0}^{\alpha} \sum_{(i\in\varphi)} I_i}{P_k})\Big)\right) \tag{2.5}$$

$$\cong \mathbb{E}_{a_i}\left(\mathbb{E}_{h_{i0}}\Big(\Pi_{(i\in\varphi)} \exp(\frac{T_k d_{k0}^{\alpha} I_i}{P_k})\Big)\right) \tag{2.6}$$

$$\cong \mathbb{E}_{a_i}\left(\mathbb{E}_{h_{i0}}\Big(\Pi_{(i\in\varphi)} \exp(\frac{-T_k d_{k0}^{\alpha} P_i h_{io} d_{k0}^{-\alpha} a_i}{P_k})\Big)\right) \tag{2.7}$$

where $\mathbb{E}_{a_i}$ is the expectation with respect to the random variable a_i and $\mathbb{E}_{h_{i0}}$ denotes the expectation of the fading coefficients from the devices i to the desired receiver.

The Bernoulli distribution property can be used to simplify Eq. (2.7), $\mathbb{E}_{a_i}\exp(a_i x) =$

$(1 - P_A + P_A \times \exp(x))$, and according to [49] $\mathbb{E}_{h_{i0}}(\exp(-h_{i0} \times y)) \cong 1/(1+y)$.

Consequently, the probability of successful transmission for the participant devices within the network area can be explicitly obtained as:

$$P(SIR \geq T_k) \cong \Pi_{(i \in \varphi)} \left(1 - P_A + \frac{P_A}{(1 + T_k(\frac{d_{k0}^{\alpha}}{d_{i0}})(\gamma_{ki})} \right) \tag{2.8}$$

where $\gamma_{ki} = P_i/P_k$ is the power ratio.

2.4.2 Achievable Transmission Capacity over Slotted-ALOHA

The ALOHA protocol is a class of fully decentralised MAC protocols [50] that does not perform carrier sensing and attempts to avoid packet collision. The slotted-ALOHA protocol was introduced to enhance the utilisation of the shared communication medium and reduce the chances of collisions for multiple transmitting devices by synchronising the transmission of devices at the beginning of discrete time slots.

In the WMN, the probability for each device sending a packet to neighbours is P_A and the probability of being ready to receive a packet is $(1 - P_A)$. Therefore, the probability of every mesh node getting a packet from one of its neighbours is $P_A(1 - P_A)P(SIR \geq T_k)$ where T_k is the predefined SINR threshold, and $P(SIR \geq T_k)$ represents the successful transmission probability. Based on Shannon's Theorem, the achievable transmission capacity increases when the transmit power increases, and the packet can carry up to $\log(1 + SIR)$ (bits/second/hertz) data [49]. Thus, the capacity C_{ALOHA} for the mesh network can be written as:

$$C_{ALOHA} = P_A(1 - P_A)\log(1 + SIR)P(SIR \geq T_k). \tag{2.9}$$

From Eq. (2.8), the outage probability (θ_k) constraints concerning P_A, T_K and γ_{ki} can be obtained as follows:

$$1 - \Pi_{i\in\varphi}\left(1 - P_A + \frac{P_A}{(1 + T_k(\frac{d_{k0}^{\alpha}}{d_{i0}^{\alpha}})(\frac{P_i}{P_k})}\right) \leq \theta_k. \tag{2.10}$$

To simplify Eq. (2.9), a natural logarithm is applied to compute the maximum achievable transmission capacity with respect to P_A. Hence, we use an auxiliary function $f(P_A) = \ln(C_{ALOHA}(P_A))$ to simplify the search of the optimal value P_A that maximises C_{ALOHA}. Then, the problem formulation can be written as:

$$\begin{aligned}\arg\max_{P_A} f(P_A) = \arg\max\Big(&\ln(P_A) + \ln(1 - P_A) \\ &+\ln(\log(1 + T_k)) \\ &+ \sum_{(i\in\varphi)} \ln\Big(1 - P_A + \frac{P_A}{(1 + T_k(\frac{d_{k0}^{\alpha}}{d_{i0}^{\alpha}})\gamma_{ki})}\Big)\Big)\end{aligned} \tag{2.11}$$

$$\text{s.t.} \quad \varepsilon_k \leq \sum_{(i\in\varphi)} \ln\left(1 - P_A + \frac{P_A}{(1 + T_k(\frac{d_{k0}^{\alpha}}{d_{i0}^{\alpha}})\gamma_{ki})}\right). \tag{2.12}$$

where $\varepsilon_k = \ln(1-\theta_k)$, derived from Eq. (2.10), is used to ensure that the system maintains a minimum level of reliability and performance, which is crucial for robust and efficient communication in our system.

With a Taylor series expansion, which is $\ln(1-x) = -x-x^2/2-x^3/3-\ldots \cong -x, (|x| < 1)$, so Eq. (2.12) can be simplified to:

$$\varepsilon_k \leq \sum_{(i\in\varphi)} \left(-P_A\Big(1 - \frac{1}{(1 + T_k(\frac{d_{k0}^{\alpha}}{d_{i0}^{\alpha}})\gamma_{ki}}\Big)\right). \tag{2.13}$$

Thus, Eq. (2.11) can be reformulated as:

$$\begin{aligned}\arg\max_{P_A} f(P_A) = \arg\max_{P_A}\Bigg(& \ln\big(P_A\big) + \ln\big(1 - P_A\big) + \ln\big(\log(1 + T_k)\big) \\ & + \sum_{(i\in\varphi)} \Bigg(- P_A + \frac{P_A}{(1 + T_k(\frac{d^{\alpha}_{k0}}{d^{\alpha}_{i0}})\gamma_{ki})}\Bigg)\Bigg). \end{aligned} \tag{2.14}$$

Hence, Eq. (2.14) can be written as:

$$\begin{aligned}\arg\max_{P_A} f(P_A) = & \arg\max_{P_A}\Bigg(\ln(P_A) + \ln(1 - P_A) \\ & + \ln\big(\log(1 + T_k)\big) - P_A f_2\Bigg)\end{aligned} \tag{2.15}$$

where $f_2 = \sum_{(i\in\varphi)}[1 - \frac{1}{(1+T_k(\frac{d^{\alpha}_{k0}}{d^{\alpha}_{i0}})\gamma_{ki})}]$, and therefore a partial derivative of $f(P_A)$ will be taken to find P_A that will maximise the function $f(P_A)$ in Eq. (2.15):

$$\frac{\partial f(P_A)}{\partial P_A} = \Bigg(\frac{1}{P_A} - \frac{1}{(1 - P_A)} - f_2\Bigg).$$

The probability of being in transmitting mode using the slotted ALOHA protocol at $\frac{\partial f(P_A)}{\partial P_A} = 0$ is defined as $P_A(0)$ and obtained as:

$$P_A(0) = \frac{f_2 + 2 - \sqrt{f_2^2 + 4}}{2f_2}$$

so, $\frac{\partial f(P_A)}{\partial P_A} > 0$, when $0 \leq P_A < P_A(0)$ and $\frac{\partial f(P_A)}{\partial P_A} < 0$ when $P_A(0) < P_A < 1$. Consequently, if the system is assumed to have a constant threshold T_k and power ratio γ_{ki}, then $f(P_A)$ will increase when P_A is increased within the range of 0 to $P_A(0)$, and by contrast, it will decrease when $P_A > P_A(0)$.

A special case has been adopted by setting the system's outage probabilities equal to its SINR threshold values to find the maximum transmission capacity based on the slotted ALOHA protocol for the wireless mesh networks. Then Eq. (2.9) can be reformulated to restate the maximum achievable transmission capacity as follows:

$$\begin{aligned}\max(C_{ALOHA}) &= P_A(1-P_A)\log(1+T_k)(1-\theta_k),\\ &= \Bigg(P_A(1-P_A)\log(1+T_k)\\ &\times \Pi_{(i\in\varphi)}\Big(1-P_A+\frac{P_A}{(1+T_k(d_{k0}^{\alpha}/d_{i0}^{\alpha})}(\gamma_{ki})\Big)\Bigg).\end{aligned} \tag{2.16}$$

From this analysis, the intuition is provided as to what extent the network parameters T_k, P_A and γ_{ki} effect the maximum achievable transmission capacity of the slotted-ALOHA mesh network.

2.4.3 Latency

The required time to achieve a convergence state in any FL model plays a key role in evaluating the system's performance. Therefore, the average learning latency for the participants will be one of the performance metrics. Latency in the proposed centralised and decentralised FL will be defined as the expected time duration (in seconds) required for the model to complete learning in a typical one-hop mesh communication network. Let R denotes the smallest number of iterations to meet the convergence criterion. The expected learning latency (T_{total}) is a function in the computation time ($T_{computation}$) which is the required time for the device to run and update the local model, the server to devices (participants) broadcast communication time ($T_{broadcast}$) and the device to server communication time (T_{cmm}) over a number of iterations R for the number of successful transmit participants A_i in the learning process:

$$T_{total} = R\sum_{(s_i=1)}^{A_i}\big(T_{cmm}^{(s_i)}\big) + R(T_{computation}+T_{broadcast}). \tag{2.17}$$

The summation of the communication process time T_{cmm} in real applications is slightly variable depending on the number of participants in each iteration. In order to simplify the calculation, it will be assumed that the network has a fixed average number of successful transmission participants A_i for the whole process in centralised and decentralised

FL.

2.4.4 Accuracy and Loss

Accuracy and loss functions are the two main model metrics that are mainly applied to adjust the model weights during the training process and to measure the system performance in order to optimise a model (e.g., a convolution neural network model) and solve, for example, a face recognition problem. Accuracy is calculated as follows:

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}}. \tag{2.18}$$

Loss is a measure of the difference between the actual output value and the predicted output value by the implemented model. In the classification models whose output values are an array of probability values between 0 and 1, the most common loss function applied is the cross-entropy loss function. The cross-entropy loss function is also known as logistic loss, log loss or logarithmic loss. The probability of each predicted value is weighed against the actual desired output 0 or 1, and a loss is measured based on how far it is from the actual expected value for each sample. A larger loss for significant differences close to 1 and more, and a slight loss for minor differences tending to 0, and therefore, the overall cross-entropy loss of 0 means the model is perfect. The cross-entropy loss function is defined as:

$$Loss = -\sum_{j=1}^{m}\sum_{i=1}^{n} y_{(i,j)} \log(p_{(i,j)}), \tag{2.19}$$

for n classes and m samples

where $y_{(i,j)}$ is the actual output and $p_{(i,j)}$ is the softmax probability of the model output for the i^{th} class in the classification problems and j^{th} instance sample.

Therefore, the objective is almost always to increase the accuracy and minimise the loss of the FL models or any other implemented models.

2.5 The Learning Criterion for DFL

The designed system considers a group of φ individual participants nodes (i.e., edge devices) in which set $\varphi = 1, \ldots, K$ is randomly located as a spatial point process following a stationary Poisson Point Process PPP [51] with intensity (λ) in disk cells with uniformly distributed over a large-scale network. In supervised learning, each node $i \in \varphi$ has access to a dataset $\mathfrak{O}_{(i)}$ consisting of k instance-label pairs of samples (X_i^n, Y_i^n), where X_i^n and Y_i^n represent the labelled sample input and output for the device i, respectively and $n = 1, ..., k$. The samples are a subset of heterogeneous or homogeneous datasets that follow an unknown probability distribution $p(x, y)$ and possibly have non-empty interaction for different sets (i.e., $\mathfrak{O}_{(i)}$ and $\mathfrak{O}_{(j)}$ where $i \neq j$). Each instance $X_i^n \in X_i \subseteq X$, where X_i denotes the local instance space of node i and X denotes a global instance space, which satisfies $X \subseteq \bigcup_{i=1}^{\varphi} X_i$.

Similarly, let y denote the set of all possible labels over all the nodes. Some examples include $y = 0, 1$ for binary classification learning and $y = \mathbb{R}$ for regression learning.

There are $X_i^{(1)}, X_i^{(2)}, ..., X_i^{(k)}$ i.i.d samples at each participant node, and the total number of examples k is a variable depending on the size of the local datasets for each one.

All participants' devices have to have the same machine-learning model (e.g., a CNN), which have a common weight parameters matrix ($\mathbf{W}$). The main aim of the designed model is to minimise the cross-entropy loss function between the expected (actual) output and the predicted output:

$$\arg\min_{\mathbf{W}} F(\mathbf{W}) \triangleq \frac{1}{A_i} \sum_{(i \in \varphi)} f_i(\mathbf{W}) \tag{2.20}$$

where $F(\mathbf{W})$ denote the global loss function, $f_i(\mathbf{W})$ represents the local loss for the device i, s_i represents the set of active devices that succeed in transmitting to the i^{th} participant at each iteration in the training process and A_i is the length of the s_i set as:

$$s_i = 1, 2, \ldots.., A_i, \forall \text{ SINR } \geq T_k.$$

The local loss function for the i^{th} device is calculated by the model cross-entropy loss for the local training dataset $\mathfrak{D}_{(i)}$ as follows:

$$f_i(\mathbf{W}) = \frac{1}{M} \sum_{(m=o)}^{M} l(h_{\mathbf{W}}(X_i^m), y_i^m), \qquad (2.21)$$

$$m = 0, 1, 2, ..., M$$

where $m \subseteq n$ is the subset of the total number of local training datasets and M is equivalent to the length of the participant datasets divided by the batch size as follows $(\frac{|\mathfrak{D}_{(i)}|}{B_i})$, and $l(h_{\mathbf{W}}(X_i^{(m)}), y_i^{(m)})$ denotes the cost function for the weights matrix $\mathbf{W}$ evaluated on a hypothesis $h_{\mathbf{W}}(X_i^m)$ With data samples X_i^m. For instance, the hypothesis for the simple linear regression is defined as $h_{\mathbf{W}}(X) = \mathbf{W}_0 + \mathbf{W}_1 X$.

At the t^{th} iteration of the DFL process, each device $i \in \varphi$ has a local parameters weights matrix $\mathbf{W}_i^{(t)}$ that is updated to maximise the model accuracy and to find an optimum solution to the target problem.

In the CFL models, there will be two optimiser levels. First is the local model level; the local optimiser in each device to update the local parameters based on the local dataset can use a common machine learning optimiser called Stochastic Gradient Descent algorithm SGD. Second is the global model level; a global model optimiser uses the aggregated parameters from neighbours (i.e., Federated Averaging algorithm (FedAvg)) to create and update the global model.

Each participant i makes M training passes over its local datasets $\mathfrak{D}_{(i)}$ to update its local model weights simultaneously via SGD with learning rate (η) and batch B_i which is shown as follows:

$$\nabla f_i(\mathbf{W}_i^t) = \frac{1}{M} \sum_{(m=0)}^{M} \nabla l(h_{(\mathbf{W}_i^t)}(X_i^m, y_i^m), \tag{2.22}$$

$$\forall m = 1, \ldots, M \text{ and } i \in \varphi$$

$$\mathbf{W}_i^t := \mathbf{W}_i^t - \eta_i \nabla f_i(\mathbf{W}_i^t). \tag{2.23}$$

Here $\nabla f_i(\mathbf{W}_i^t)$ denotes the gradient matrix obtained from the batch B_i of the device i's local samples after M local training on the local data at each iteration t. The gradient $\nabla f_i(\mathbf{W}_i^t)$ expresses the rate of change of f_i with respect to the model parameters $\mathbf{W}$ at the iteration t.

The total number of local training M is equivalent to the participant datasets length divided by the batch size ($M = |\mathfrak{D}_{(i)}|/B_i$) in order to train the model locally on its datasets before sharing the parameters with neighbours. After a certain number of iterations for each participant device trained a model on its local dataset with sufficient hyper-parameters, the models' parameters are shared with other neighbours via a one-hop communication process following the mesh topology.

For all participants, the updates are simultaneously done where each participant receives the other neighbours' weights and gradients and averages them with the local weights and gradients.

In the next step, each participant in the network successfully aggregates the local parameters $\mathbf{W}_i^t$ from the trusted neighbours who satisfied the wireless communication constraints to execute the Federated Averaging (FedAvg) algorithm in order to obtain a new global model at each t^{th} iteration. Then, each authorised participant sends a broadcast containing the latest global model weights matrix to all participants located within a one-hop mesh network and starts the next iteration. Figure 2.3 illustrates the proposal DFL network architecture and the learning process steps. The global model which can be denoted as $\hat{\mathbf{W}}_i^t$ is executed by averaging the local model with the aggregated model weights from the neighbours s_i (the set of active devices whose received their weights successfully) at the iteration t.

These weights are then applied on the local model using the device dataset $\mathfrak{D}_{(i)}$ and batch size B_i to create an updated weights matrix $\mathbf{W}_i^{(t+1)}$ and broadcast it to the neighbours in the next iteration as follow:

$$\hat{\mathbf{W}}_i^t = \frac{1}{A_i + 1}\Big(\mathbf{W}_i^t + \sum_{(s_i=1)}^{A_i} \hat{\mathbf{W}}_{s_i}^t\Big) \tag{2.24}$$

$$\hat{\nabla} f(\hat{\mathbf{W}}_i^t) = \frac{1}{M} \sum_{(m=1)}^{M} \hat{\nabla} l \left(h_{(\hat{\mathbf{W}}_i^t)}(X_i^m, y_i^m) \right); \tag{2.25}$$

$$\hat{\mathbf{W}}_i^t = \mathbf{W}_i^{(t+1)}. \tag{2.26}$$

Then these new parameters $\mathbf{W}_i^{(t+1)}$ at the device i will be used to train the local model in the next iteration and also share $\mathbf{W}_i^{(t+1)}$ with all participants within one-hop communication range to repeat the same procedure at each device until achieving the predefined convergence bound ε_k of the implemented model. The convergence is determined by measuring the average loss function of all active device $(s_i = 1, ..., A_i)$ whose successful transmission to the i^{th} device at each iteration as follows:

$$\begin{aligned} \arg\min_{\mathbf{W}} \nabla F_i(\mathbf{W}) \triangleq & \Big[\Big(\frac{1}{(A_i+1)}(\hat{\nabla} f_i(\hat{\mathbf{W}}_i^t) \\ & + \sum_{s_i=1}^{A_i} \hat{\nabla} f_{s_i}(\hat{\mathbf{W}}_{s_i}^t)\Big) \Big| A_i \geq 1\Big] \leq \varepsilon_k. \end{aligned} \tag{2.27}$$

The participants will communicate and exchange the parameters and the system loss function's values over a wireless mesh network via a peer-to-peer manner. The number of successful transmit devices (participants) is subject to wireless communication con-

Algorithm 2.1 Decentralised Federated Learning

```
1: All participants have initial weights with $\mathbf{W}(0)$
2: **for** each iteration $t = 1, 2, 3, \ldots..R$ **do**
3:   **for** each device $i = 1, 2, 3, \ldots..K$ **do {in parallel}**
4:       **for** $m = 1, 2, 3, \ldots M$, where $M = \frac{|\mathfrak{D}_{(i)}|}{B_i}$ **(Local training steps)**
5:         $\nabla f_i(\mathbf{W}_i^t) = \frac{1}{M} \sum_{(m=1)}^{M} \nabla l(h_{(\mathbf{W}_i^t)}(X_i^m, y_i^m))$
6:         $\mathbf{W}_{(i)}^t \longleftarrow \mathbf{W}_{(i)}^t - \eta_i^t \nabla f_i(\mathbf{W}_i^t; B_i)$
7:       **end**
8:       **from** the $i^{th}$ device's neighbours $s_i = 1, 2, \ldots..A_i$ **do {in parallel}**
9:         receive $\hat{\mathbf{W}}_{s_i}^t, \hat{\nabla} f_{s_i}(\hat{\mathbf{W}}_{s_i}^t)$,
10:        $\hat{\mathbf{W}}_i^t \longleftarrow \frac{1}{(A_i+1)}(\mathbf{W}_i^t + \sum_{s_i=1}^{A_i}(\hat{\mathbf{W}}_{s_i}^t))$
11:        $\hat{\nabla} f_i(\hat{\mathbf{W}}_i^t) = \frac{1}{M} \sum_{(m=1)}^{M} \hat{\nabla} l(h_{(\hat{\mathbf{W}}_i^t)}(X_i^m, y_i^m))$
12:          If $\frac{1}{(k+1)}(\hat{\nabla} f_i(\hat{\mathbf{W}}_i^t) + \sum_{s_i=1}^{A_i} \hat{\nabla} f_{s_i}(\hat{\mathbf{W}}_{s_i}^t)) \; |A_i \geq 1] \leq \varepsilon_k$
13:              **Yes:** end process (Gradient Convergence)
14:              **No:** continue
15:          $\mathbf{W}_i^{(t+1)} \longleftarrow \hat{\mathbf{W}}_i^t$
16:   **end**
17: **end**
```

straints. Consequently, it is possible to show that every device i in the network, after receiving the latest global model update, can train its model using the latest update and then evaluate how the model performance improved in terms of the loss functions and accuracy.

To sum up, the DFL process is divided into t iterations. At each iteration, each i^{th} participant performs local learning and exchanges their parameters $\hat{\mathbf{W}}_i^t$ in parallel with other cooperative neighbours s_i to create global models at each device in a central server-free manner. The R symbol in pseudocode represents the total number of iterations the system needs to achieve the consensus and meet the gradient divergence condition in Eq. (2.27). Finally, the DFL learning steps are presented in Algorithm 2.1.

2.6 Simulations and Results

2.6.1 The Simulation of the Network Communication

The proposal network for the CFL and DFL approaches will have many participants distributed on a two-dimensional bounded space. A large-scale circular area with radius

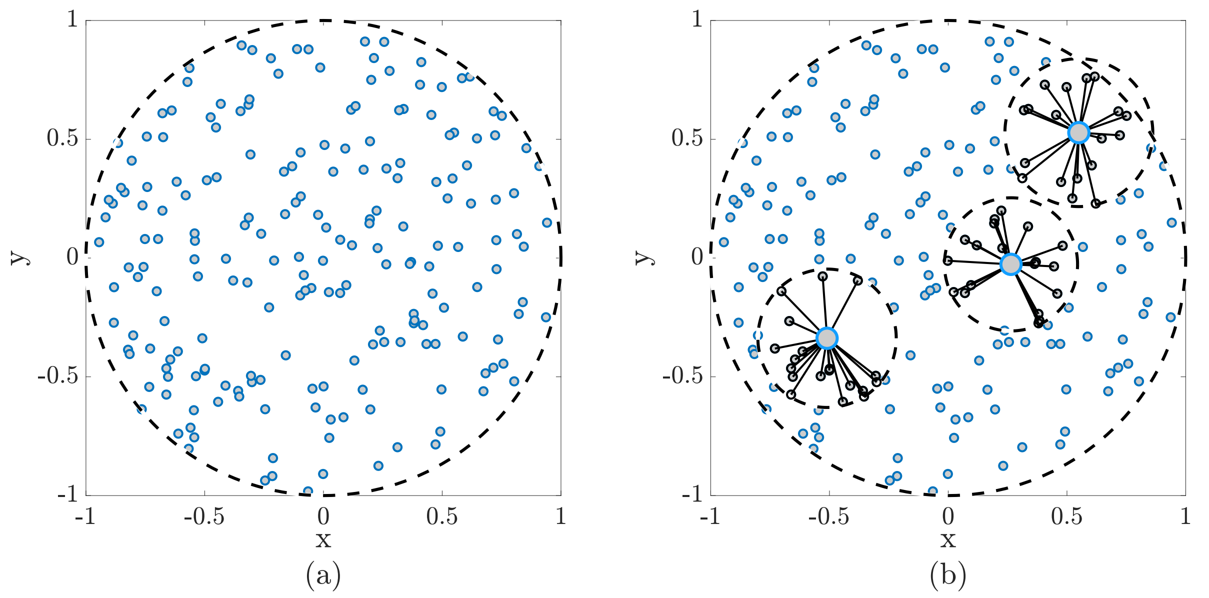


Figure 2.4: (a) The distribution for random participants around the CFL centre server. (b) Example of three participants communicating with neighbours in DFL approach.

$R_r > 0$ is proposed for notational simplicity, but a hexagonal one can also be applied, as the outcomes will differ slightly by a very small constant. The number of participants is distributed according to the Poisson distribution with network intensity λ and area A. In other words, the number of participants within A can be defined as a Poisson random variable with mean λA, where $\lambda > 0$ is the network intensity, and A is the network circular area ($A = 2\pi R_r^2$).

On the one hand, the central point in a two-dimensional bounded space will be the position for the central server of the network in the CFL model as illustrated in Fig. 2.4 (a), and the participants will be randomly distributed within the target area. However, the successful transmission and participation in the learning process will be subject to wireless communication constraints (see Sec. 2.4.1) to approximate the real situation applications.

On the other hand, in DFL, each point (participant) i will define each neighbour based on a one-hop communication link within the device signal coverage that will be a smaller circle with a radius $r_i < R_r$. For instance, the wireless mesh connections for three DFL participants with a Rayleigh fading are illustrated in Fig. 2.4 (b).

The positions of the participants are independently and randomly distributed within

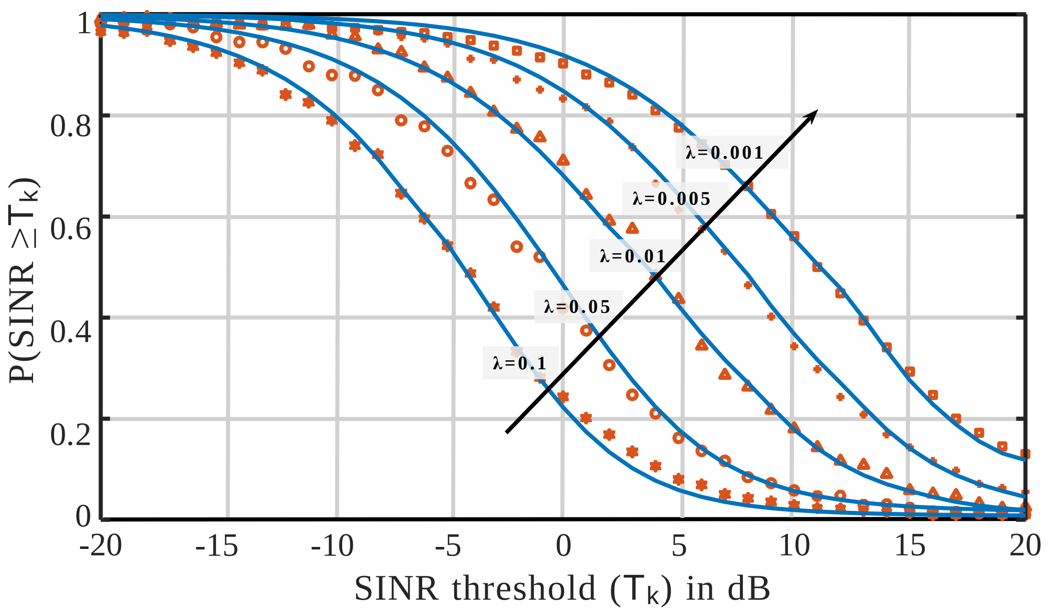


Figure 2.5: The simulation (markers) and theoretical results (solid lines) of the relationship between the SINR threshold and the probability of success for the participants within different network intensities.

the network area (circular area), where the distance for each participant i to the centre point (central server) is $d_{k0} \leq R_r$. In the simulation, the relationship between the SINR threshold T_k and the probability of successful transmission $P(\text{SINR} \geq T_k)$ of the network, participants will be found based on this study analysis as in Eq. (2.1) from Sec 2.4.1 and the total number of participants is a random variable in the network space radius $R_r = 1000\ m$ and the intensity $\lambda = 0.1$.

The participants are assumed to have equal transmit power (0.8 Watts). Only a finite number of participants can simultaneously exchange and update their parameters based on the slotted ALOHA protocol. When the given parameters (i.e., signal power, distance, and interference) were applied in Eq. (2.8), the theoretical results matched the simulation results, and both proved that the number of successful transmitter devices decreased when the SINR threshold increased for different intensity of users in the network, as shown in Fig. 2.5.

In contrast, reducing the SINR threshold allows more participants to involve in the learning process, but the required bandwidth and the system's latency will also increase. Therefore, there is a need to examine the trade-off between the probability of success and the SINR threshold to achieve higher throughput and capacity and acceptable latency.

Consequently, the proposed wireless communication model outcomes in Fig. 2.5 confirm the conclusions of the theoretical analysis in Eq. (2.8) and Eq. (2.16) where a trade-off between the probability of success transmit and the SINR threshold is required to satisfy the FL network target in term of the capacity and the number of users (participants) within the network during the learning process.

2.6.2 CFL Setup and Simulation

The simulation settings are proposed for implementing an FL with a central server deployed in the centre of the target disk area with a radius R_r. It has been assumed that the network has a large-scale massive IoT and edge devices network following the PPP with intensity λ and the total number of participants is N, the distance from any participant i to the server is $r_i < R_r$.

To simplify the simulation, all participants are assumed to have the same transmit power and are all randomly distributed around the centre within the network area. The system will be trained using Python and TensorFlow API frameworks, the well-known MNIST dataset and a local algorithm on the edge devices to perform digit number recognition from handwritten images. While this is a simple and well-understood problem, it is being used here to illustrate the principle. The MNIST dataset contains labelled data samples of 60k images, each with a size of 28×28 pixels. This dataset was shuffled to distribute randomly for the participants, where each participant had 200 random samples. The edge devices will implement a CNN algorithm as a classifier model to evaluate the outcome results and the system performance metrics. Large-scale massive IoT networks are used to validate the algorithm, and thus the network has two cases:

I. Low mobility scenarios where the number of participants' success transmitted in the system can be the same for all iterations until the system convergences.

II. High mobility scenarios where participants move in a wide geometric area and many participants are IN and OUT of the network during the learning processes, such as autonomous vehicles and Unmanned Aerial Vehicle (UAV) networks. Thus, the number of successful transmits is usually different in each learning iteration.

This simulation will implement the first case, low mobility scenarios, and we will leave the second case for subsequent works. In both cases, the number of successful transmit participants in the system is subject to the communication network constraints (described in Sec. 2.4.1) and the effect of network parameters in terms of the devices' intensity and participants' distribution.

Based on the proposed wireless communication model in Sec. 2.6.1, the system will be allocated the target SINR threshold (-10 dB) in order to achieve a higher transmission capacity, the probability of the participant to be in a transmit mode $P_A = 0.5$ and the intensity of users' devices $\lambda = 0.1$ to increase the number of participants. This results in having around 80% probability of successful transmit devices ($P(\text{SINR} \geq T_k)$) in the network as shown in Fig. 2.5.

Consequently, the total number of participants within the target area is 80 devices, but the average number of successful transmit participants in the CFL process was 65 participants.

In this study, the CFL with a central server is designed by using the FedAvg algorithm as a global model optimiser. The main function of the server algorithm is to aggregate and average the participants' parameters to update the new global model at each iteration and then measure the accuracy and the loss of the model outcomes.

The system evaluation for the CFL network optimisation regarding the accuracy and loss used the cost function Cross-Entropy. The simulated system in Sec. 2.6.1 has 65 successful transmit participants as regards the communication constraints in Sec. 2.4.1.

It can be noted from Fig. 2.6 that the procedure moved progressively towards the global minimum dramatically in the first 50 epochs, where the accuracy increased from 17% to 90%. After 200 iterations, the model moved progressively and achieved the convergence state with the accuracy and loss of 98.1% and 0.15, respectively. The latency

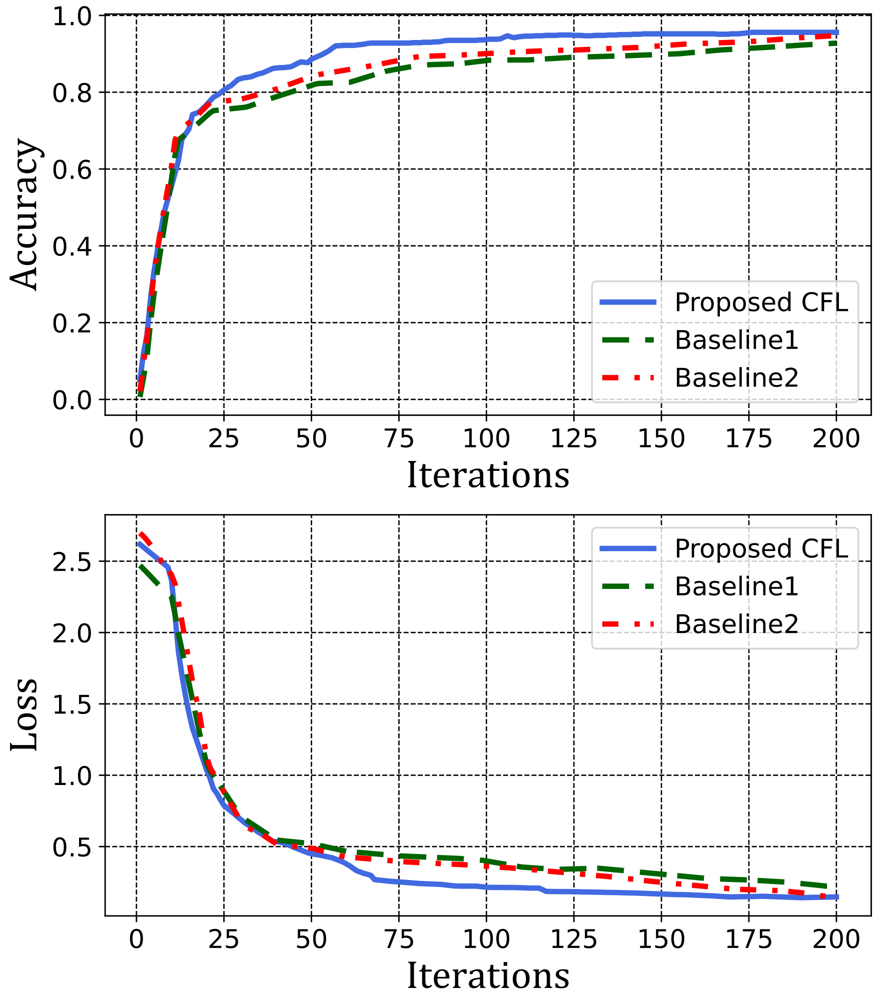


Figure 2.6: The accuracy and loss for the CFL network.

of the model was 1850 seconds, which was the required time to achieve the predefined convergence bound. Despite having applied constraints for the proposed CFL model to build a robust CFL network by considering the communication constraints and real environment scenarios in the simulation, the model was capable of converging faster and achieving slightly higher accuracy and lower loss than the FL baselines that have not considered the communication constraints in practice. In other words, the baselines estimate a fixed number of participants in the learning process without considering the network challenges, which cannot be reliable in real applications. In contrast, our proposed CFL model defines the participants based on the communication model and considers the real applications environment's constraints to obtain a trustworthy and robust network while achieving high accuracy and low loss.

Although the system achieved high accuracy and reached convergence after around 200 iterations in the designed simulation, there is potentially a single point of failure at the central server. The system, in reality, could be completely down if any failure occurs in the central server or the link to the server is blocked (communication bottleneck).

2.6.3 DFL Setup and Simulation

The set-up simulation uses Python and TensorFlow API frameworks to implement and evaluate the design of the DFL system over a WMN for IoT devices. In order to make an accurate comparison, we concentrate on the same previous CFL design in terms of the communication links constraints, and the parameters model optimises, low mobility (fixed) and the same total number of participants in the learning process but without a central server. The system was evaluated based on performance metrics; the validation accuracy, the latency and the convergence speed rate. The simulation settings are listed as follows:

- **Training settings**

The participants trained a classification CNN model on the MNIST datasets. As in CFL, each participant will have 1000 random samples to train the local models, but the parameters in this DFL approach will be shared directly with neighbours to create and update the global models at each device without needing a central server.

The local models of each participant were trained and updated using a stochastic gradient descent (SGD) optimiser with the same learning rate of 0.01 and batch size 32 (hyper-parameters) in CFL simulation.

Furthermore, the designed system also used another two different local models optimiser' algorithms, the Adam algorithm [52] and Root Mean Square Propagation (RMSProp) optimiser [53], to observe how the performance of the DFL model gets affected by implementing different optimiser' methods.

- **Network Settings**

In the simulation, the network is designed to simulate the communication stage between the participants' devices in a peer-to-peer manner over wireless mesh networking, avoiding the communication bottleneck challenge in the central server in the CFL scenario. Each participant will successfully communicate and exchange the parameters with neighbours if the desired transmitters have an SINR over the target threshold and no collision occurs. The number of successful transmits participants is configured to be variable between the participants in an asymmetric manner. For instance, there are two participants, A and B, within the network, and participant A can receive parameter updates from participant B. In contrast, participant B does not have to successfully obtain the parameters update from Participant A unless he satisfies the network communication constraints.

- **Comparison Setting**

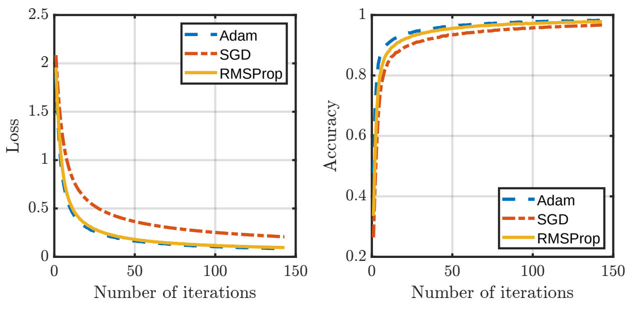
The DFL system outcomes will be compared with the CFL outcomes in terms of system performance. Both DFL and CFL were implemented to train their models on the same factors (i.e., the dataset, the total number of participants within the network and the network geographic area) in order to make them comparable. Thus, the accuracy, loss and latency and convergence speed were measured.

- **Simulation Results**

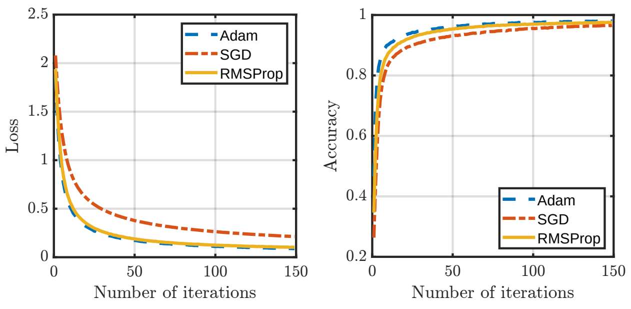
Based on the successful transmit conditions and the network capacity, every device within the network created its one-hop neighbours' group to exchange parameters and performed the CFL model training. The model results have been recorded for some random participants in the DFL simulation as an example to evaluate the system behaviour.

The designed network shows that these random recorded participants are connected successfully with a variant number of neighbours (who meet the communication constraints and they are able to exchange parameters and update their global models).

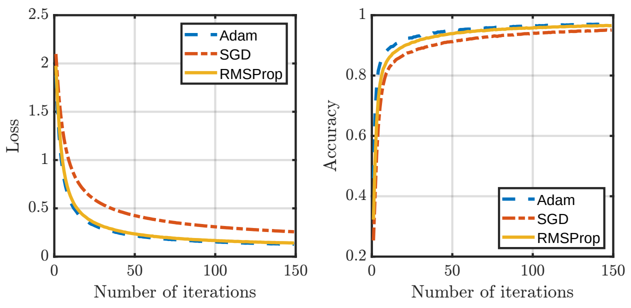
The DFL's simulated system has shown that the classification prediction achieved about 90% accuracy in the first 20 iterations for all these recorded participants with any of the three different model optimises mentioned in the training settings (SGD, ADAM and



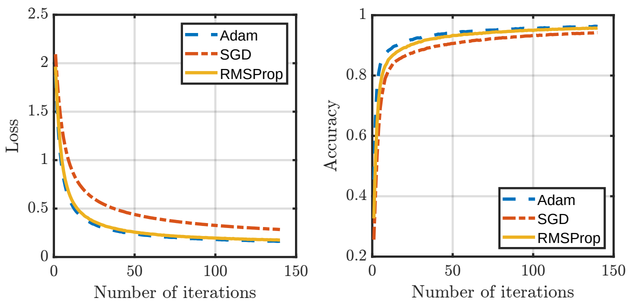
(a) Number of participants = 15



(b) Number of participants = 17



(c) Number of participants = 22



(d) Number of participants = 27

Figure 2.7: The DFL model outcomes (Accuracy and Loss) for four random participants.

RMSprop optimises).

The results and statistics of some participants within the network are illustrated in Fig. 2.7, and Table 2.2. The results for the DFL approach without a central server show high accuracy and low cross-entropy loss in predicting the digit number from the handwritten samples after 135 iterations.

In this study, the wireless mesh networking model in Sec. 2.6.1 is integrated with the DFL model to verify the number of successful participants during the learning process. The system shows that each client (i.e., participant) will be able to communicate with particular neighbours depending on the neighbours' locations, the desired participant transmit power and other devices' power interference.

The latency for each participant varies slightly depending mainly on the number of iterations that each participant is required to achieve the system convergence status. In this study, the results have shown that the average latency for the participants was 420 seconds with the assumption the computation and broadcast time are equal for all participants. The expected system convergence in the DFL model is around 130 iterations on average.

Table 2.2: Simulation results for some random participants of the designed DFL model.

	No. of Succ. Transmit Neighbours	Iterat ions	Accuracy			Cross-Entropy Loss		
Optimiser Algorithm			**ADAM**	**SGD**	**RMSprop**	**ADAM**	**SGD**	**RMSprop**
Participant 1	15	135	0.982	0.966	0.969	0.088	0.214	0.149
Participant 2	17	135	0.979	0.964	0.974	0.094	0.200	0.138
Participant 3	22	135	0.970	0.948	0.965	0.133	0.270	0.237
Participant 4	27	135	0.962	0.951	0.957	0.164	0.218	0.178
Participant 5	13	135	0.979	0.957	0.974	0.131	0.2673	0.2346
Participant 6	24	135	0.981	0.959	0.976	0.132	0.266	0.2341
Participant 7	22	135	0.986	0.971	0.981	0.092	0.197	0.136
Participant 8	19	135	0.974	0.961	0.964	0.125	0.268	0.231
Participant 9	10	135	0.977	0.956	0.963	0.093	0.187	0.211
Participant 10	24	135	0.983	0.951	0.972	0.161	0.256	0.236
Participant 11	18	135	0.975	0.946	0.969	0.171	0.218	0.198

As shown in Fig. 2.7, the system achieved sufficiently high accuracy and low loss in the data predictions. The randomly chosen participants' records show that the participants can learn in parallel and follow similar progress toward convergence in terms of accuracy and loss. The designed system offers the benefits of utilising the DFL framework, where the data never leaves the participants' devices, and privacy restrictions exist. In comparison with the centralised model, the decentralised model achieved competitive results without needing a central server. For instance, participant 1 reached 98.2% accuracy and 0.088 cross-entropy loss after only 135 iterations.

In contrast, after more than 200 iterations, the centralised model achieved the convergence state with accuracy and loss of 98.1% and 0.15, respectively. Based on the latency criteria in the Sec. 2.4.3, CFL has higher latency than DFL as the system outcomes find that the communication cost and the number of iterations of CFL were higher than the DFL approach.

Thus, DFL can reduce the latency and loss and increase the convergence speed in which they can outperform CFL.

From these results, we can conclude that the DFL models over WMN produced significant developments in classification prediction using sufficient datasets. In this study, the DFL framework is combined with the wireless mesh networking using the Slotted-ALOHA protocol to improve communication between the participants during the learning process.

The DFL approach over WMN using the slotted-ALOHA protocol could be very competitive to the CFL. The simulated models prove that the designed DFL system can achieve better latency, more flexibility, and similar accuracy without installing a central server in the network.

2.7 Conclusions

To sum up, DFL reduced the communication cost compared to CFL as the participants' devices communicate directly and send the packets of their parameters and updates with only one-hop neighbours using slotted-ALOHA as the devices' MAC protocol. The network topology was a mesh network topology. In this study, the network communication model simulated the real scenario of the mesh networking topology considering the frequency interference in the network environment and then combined it with the DFL model to train the network efficiently and reliably. The effectiveness of combining the DFL framework and the mesh networking protocol results in a comprehensive improvement in the model performance, and the DFL approach is becoming very competitive compared to CFL.

The following is a summary of this research and future work:

1. Analysis of the wireless communication stage between the participants during the learning process in order to simulate the real scenarios of interaction between the IoT devices, reducing the communication resources and increasing the system flexibility.

2. Implementing the DFL system using the FedAvg algorithm to enforce consensus techniques by sharing local model updates; established gossip methods are also extended by consensus.

3. Emphasis on an experimental IoT setup, considering convergence speed, complexity, communication cost, and average prediction accuracy on the DFL embedded devices.

4. The CFL and DFL algorithms were implemented by considering the communication stage challenges in the simulation to calibrate the real environment scenarios and the real applications.
5. In the future, DFL models need to be designed for high-mobility sensors and devices in the wireless mesh networking system to build a robust system, increase the system flexibility and scalability and enhance the performance for some applications, such as a Driverless Transport System.

References

[1] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*, PMLR, 2017, pp. 1273–1282.

[2] J. Konečnỳ, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," *arXiv preprint arXiv:1610.05492*, 2016.

[3] A. Lalitha, O. C. Kilinc, T. Javidi, and F. Koushanfar, "Peer-to-peer federated learning on graphs," *arXiv preprint arXiv:1901.11173*, 2019.

[4] C. Zhang, Y. Xie, H. Bai, B. Yu, W. Li, and Y. Gao, "A survey on federated learning," *Knowledge-Based Systems*, vol. 216, p. 106 775, 2021, ISSN: 0950-7051. DOI: `https://doi.org/10.1016/j.knosys.2021.106775`. [Online]. Available: `https://www.sciencedirect.com/science/article/pii/S0950705121000381`.

[5] M. Aledhari, R. Razzak, R. M. Parizi, and F. Saeed, "Federated learning: A survey on enabling technologies, protocols, and applications," *IEEE Access*, vol. 8, pp. 140 699–140 725, 2020.

[6] Z. Eghbali and M. Z. Lighvan, "A hierarchical approach for accelerating iot data management process based on sdn principles," *Journal of Network and Computer Applications*, vol. 181, p. 103 027, 2021.

[7] U. Shafi, R. Mumtaz, J. García-Nieto, S. A. Hassan, S. A. R. Zaidi, and N. Iqbal, "Precision agriculture techniques and practices: From considerations to applications," *Sensors*, vol. 19, no. 17, p. 3796, 2019.

[8] W. Y. B. Lim, N. C. Luong, D. T. Hoang, *et al.*, "Federated learning in mobile edge networks: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 3, pp. 2031–2063, 2020.

[9] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 50–60, 2020.

[10] Q. Li, Z. Wen, Z. Wu, *et al.*, "A survey on federated learning systems: Vision, hype and reality for data privacy and protection," *IEEE Transactions on Knowledge and Data Engineering*, 2021.

[11] X. Wang, Y. Han, C. Wang, Q. Zhao, X. Chen, and M. Chen, "In-edge ai: Intelligentizing mobile edge computing, caching and communication by federated learning," *IEEE Network*, vol. 33, no. 5, pp. 156–165, 2019.

[12] L. Lyu, H. Yu, and Q. Yang, "Threats to federated learning: A survey," *arXiv preprint arXiv:2003.02133*, 2020.

[13] S. AbdulRahman, H. Tout, H. Ould-Slimane, A. Mourad, C. Talhi, and M. Guizani, "A survey on federated learning: The journey from centralized to distributed on-site learning and beyond," *IEEE Internet of Things Journal*, vol. 8, no. 7, pp. 5476–5497, 2020.

[14] L. U. Khan, W. Saad, Z. Han, and C. S. Hong, "Dispersed federated learning: Vision, taxonomy, and future directions," *IEEE Wireless Communications*, vol. 28, no. 5, pp. 192–198, 2021.

[15] A. Koloskova, S. Stich, and M. Jaggi, "Decentralized stochastic optimization and gossip algorithms with compressed communication," in *International Conference on Machine Learning*, PMLR, 2019, pp. 3478–3487.

[16] A. G. Roy, S. Siddiqui, S. Pölsterl, N. Navab, and C. Wachinger, "Braintorrent: A peer-to-peer environment for decentralized federated learning," *arXiv preprint arXiv:1905.06731*, 2019.

[17] P. Ramanan and K. Nakayama, “Baffle: Blockchain based aggregator free federated learning,” in *2020 IEEE International Conference on Blockchain (Blockchain)*, IEEE, 2020, pp. 72–81.

[18] N. J. Dabanloo, S. Moharreri, S. Parvaneh, and A. Nasrabadi, “Application of novel mapping for heart rate phase space and its role in cardiac arrhythmia diagnosis,” in *2010 Computing in Cardiology*, IEEE, 2010, pp. 209–212.

[19] W. G. Hatcher and W. Yu, “A survey of deep learning: Platforms, applications and emerging research trends,” *IEEE Access*, vol. 6, pp. 24 411–24 432, 2018.

[20] S. Savazzi, S. Sigg, F. Vicentini, S. Kianoush, and R. Findling, “On the use of stray wireless signals for sensing: A look beyond 5g for the next generation of industry,” *Computer*, vol. 52, no. 7, pp. 25–36, 2019.

[21] S. Samarakoon, M. Bennis, W. Saad, and M. Debbah, “Federated learning for ultra-reliable low-latency v2v communications,” in *2018 IEEE Global Communications Conference (GLOBECOM)*, IEEE, 2018, pp. 1–7.

[22] H. Ye, L. Liang, and G. Y. Li, “Decentralized federated learning with unreliable communications,” *IEEE Journal of Selected Topics in Signal Processing*, vol. 16, no. 3, pp. 487–500, 2022.

[23] D. C. Nguyen, M. Ding, P. N. Pathirana, A. Seneviratne, J. Li, and H. V. Poor, “Federated learning for internet of things: A comprehensive survey,” *IEEE Communications Surveys & Tutorials*, vol. 23, no. 3, pp. 1622–1658, 2021.

[24] Y. Guo, R. Zhao, S. Lai, L. Fan, X. Lei, and G. K. Karagiannidis, “Distributed machine learning for multiuser mobile edge computing systems,” *IEEE Journal of Selected Topics in Signal Processing*, vol. 16, no. 3, pp. 460–473, 2022. DOI: 10.1109/JSTSP.2022.3140660.

[25] V. Mothukuri, R. M. Parizi, S. Pouriyeh, Y. Huang, A. Dehghantanha, and G. Srivastava, “A survey on security and privacy of federated learning,” *Future Generation Computer Systems*, vol. 115, pp. 619–640, 2021.

[26] L. U. Khan, W. Saad, Z. Han, E. Hossain, and C. S. Hong, "Federated learning for internet of things: Recent advances, taxonomy, and open challenges," *IEEE Communications Surveys & Tutorials*, 2021.

[27] Q.-V. Pham, K. Dev, P. K. R. Maddikunta, T. R. Gadekallu, T. Huynh-The, *et al.*, "Fusion of federated learning and industrial internet of things: A survey," *arXiv preprint arXiv:2101.00798*, 2021.

[28] J. Xu, B. S. Glicksberg, C. Su, P. Walker, J. Bian, and F. Wang, "Federated learning for healthcare informatics," *Journal of Healthcare Informatics Research*, vol. 5, no. 1, pp. 1–19, 2021.

[29] S. Chen, D. Yu, Y. Zou, J. Yu, and X. Cheng, "Decentralized wireless federated learning with differential privacy," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 9, pp. 6273–6282, 2022.

[30] C. Che, X. Li, C. Chen, X. He, and Z. Zheng, "A decentralized federated learning framework via committee mechanism with convergence guarantee," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 12, pp. 4783–4800, 2022.

[31] J. Kang, D. Ye, J. Nie, *et al.*, "Blockchain-based federated learning for industrial metaverses: Incentive scheme with optimal aoi," in *2022 IEEE International Conference on Blockchain (Blockchain)*, IEEE, 2022, pp. 71–78.

[32] *Openthread protocol*, Google, 2022. [Online]. Available: `[Online]%20https://openthread.io/`.

[33] P. Pinyoanuntapong, P. Janakaraj, P. Wang, M. Lee, and C. Chen, "Fedair: Towards multi-hop federated learning over-the-air," in *2020 IEEE 21st International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, IEEE, 2020, pp. 1–5.

[34] M. Xin and Y. Wang, "Research on image classification model based on deep convolution neural network," *EURASIP Journal on Image and Video Processing*, vol. 2019, no. 1, pp. 1–11, 2019.

[35] S. A. Zaidi, A. M. Hayajneh, M. Hafeez, and Q. Ahmed, "Unlocking edge intelligence through tiny machine learning (tinyml)," *IEEE Access*, 2022.

[36] P. P. Ray, "A review on tinyml: State-of-the-art and prospects," *Journal of King Saud University-Computer and Information Sciences*, 2021.

[37] H. Zhang, J. Bosch, and H. H. Olsson, "Federated learning systems: Architecture alternatives," in *2020 27th Asia-Pacific Software Engineering Conference (APSEC)*, IEEE, 2020, pp. 385–394.

[38] P. Vanhaesebrouck, A. Bellet, and M. Tommasi, "Decentralized collaborative learning of personalized models over networks," in *Artificial Intelligence and Statistics*, PMLR, 2017, pp. 509–517.

[39] A. Bellet, R. Guerraoui, M. Taziki, and M. Tommasi, "Personalized and private peer-to-peer machine learning," in *International conference on artificial intelligence and statistics*, PMLR, 2018, pp. 473–481.

[40] C. He, C. Tan, H. Tang, S. Qiu, and J. Liu, "Central server free federated learning over single-sided trust social networks," *arXiv preprint arXiv:1910.04956*, 2019.

[41] C. Hu, J. Jiang, and Z. Wang, "Decentralized federated learning: A segmented gossip approach," *arXiv preprint arXiv:1908.07782*, 2019.

[42] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh, "Scaffold: Stochastic controlled averaging for federated learning," in *International Conference on Machine Learning*, PMLR, 2020, pp. 5132–5143.

[43] F. Sattler, S. Wiedemann, K.-R. Müller, and W. Samek, "Robust and communication-efficient federated learning from non-iid data," *IEEE transactions on neural networks and learning systems*, vol. 31, no. 9, pp. 3400–3413, 2019.

[44] N. Shoham, T. Avidor, A. Keren, *et al.*, "Overcoming forgetting in federated learning on non-iid data," *arXiv preprint arXiv:1910.07796*, 2019.

[45] A. Khosravi and Y. S. Kavian, "Broadcast gossip ratio consensus: Asynchronous distributed averaging in strongly connected networks," *IEEE Transactions on Signal Processing*, vol. 65, no. 1, pp. 119–129, 2016.

[46] A. Neumann, E. López, and L. Navarro, "Evaluation of mesh routing protocols for wireless community networks," *Computer Networks*, vol. 93, pp. 308–323, 2015.

[47] M. S. Singh and V. Talasila, "A practical evaluation for routing performance of batman-adv and hwmn in a wireless mesh network test-bed," in *2015 International Conference on Smart Sensors and Systems (IC-SSS)*, IEEE, 2015, pp. 1–6.

[48] https://www.silabs.com/wireless/multiprotocol/mesh-performance [Online], *Webinar: Bluetooth mesh, thread, and zigbee network performance benchmarking*, silicon labs, 2022.

[49] T. Jing, X. Chen, Y. Huo, and X. Cheng, "Achievable transmission capacity of cognitive mesh networks with different media access control," in *2012 Proceedings IEEE INFOCOM*, IEEE, 2012, pp. 1764–1772.

[50] R. T. Ma, V. Misra, and D. Rubenstein, "An analysis of generalized slotted-aloha protocols," *IEEE/ACM Transactions on networking*, vol. 17, no. 3, pp. 936–949, 2008.

[51] M. Haenggi, J. G. Andrews, F. Baccelli, O. Dousse, and M. Franceschetti, "Stochastic geometry and random graphs for the analysis and design of wireless networks," *IEEE journal on selected areas in communications*, vol. 27, no. 7, pp. 1029–1046, 2009.

[52] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[53] T. Tieleman and G. Hinton, "Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude," *COURSERA: Neural Networks for Machine Learning, 4, 26-31*, 2012.

Chapter 3

Decentralised Federated Learning on the Edge over Wireless Mesh Networks

Chapter source: A. Salama, A. Stergioulis, S. Zaidi, and D. McLernon, “Decentralized federated learning on the edge over wireless mesh networks,” IEEE Access, vol. 11, pp. 124 709–124 724, 2023.

Abstract

The rapid growth of Internet of Things (IoT) devices has generated vast amounts of data, leading to the emergence of federated learning as a novel distributed machine learning paradigm. Federated learning enables model training at the edge, leveraging the processing capacity of edge devices while preserving privacy and mitigating data transfer bottlenecks. However, the conventional centralised federated learning architecture suffers from a single point of failure and susceptibility to malicious attacks. In this study, we delve into an alternative approach called decentralised federated learning (DFL) conducted over a wireless mesh network as the communication backbone. We perform a comprehensive network performance analysis using stochastic geometry theory and physical interference models, offering fresh insights into the convergence analysis of DFL. Additionally, we conduct system simulations to assess the proposed decentralised architecture under various network parameters and different aggregator methods such as FedAvg, Krum and Median methods. Our model is trained on the widely recognised EMNIST dataset for benchmarking handwritten digit classification. To minimise the model's size at the edge and reduce communication overhead, we employ a cutting-edge compression technique based on genetic algorithms. Our simulation results reveal that the compressed decentralised architecture achieves performance comparable to the baseline centralised architecture and traditional DFL in terms of accuracy and average loss for our classification task. Moreover, it significantly reduces the size of shared models over the wireless channel by compressing participants' local model sizes to nearly half of their original size compared to the baselines, effectively reducing complexity and communication overhead.

3.1 Introduction

In recent years, the proliferation of Internet of Things (IoT) devices has been remarkable, largely driven by advancements in 5G technology and beyond. The global IoT market is projected to grow to a market value of 1.567 trillion US dollars by 2025 [1]. The architecture of IoT, which incorporates spectrum and energy management mechanisms, is designed to optimise both spectral and energy efficiencies to their maximum potential [2]. Alongside this, breakthroughs in semiconductor technology have enabled the fabrication of transistors with sub-10 nanometre gate lengths, drastically improving processing capacity while reducing power consumption [3]. These developments play a crucial role in the extensive adoption of embedded devices, including smartphones, sensors and tablets. These devices require powerful microprocessors capable of delivering high performance while adhering to stringent energy consumption limitations.

The substantial volume of data generated by embedded sensor devices at the periphery of IoT systems has given rise to the field of Big Data, which focuses on devising effective methods for processing, disseminating, and analysing extensive datasets. In typical IoT setups, data initially collected by sensors at the edge are transmitted through a central network to a cloud server as shown in Fig. 3.1, where various data preprocessing techniques (such as data cleansing, feature extraction, denoising, etc.) are applied. Subsequently, this processed data is employed for training machine learning models tailored to the specific application domain [4]. These trained models serve various machine learning tasks, including classification, clustering, anomaly detection, and regression, facilitating precise and optimal decision-making. The data flow structure described above in IoT systems has benefited from advancements in High-Performance Computing (HPC) systems, enabling real-time and low-latency solutions across diverse industry sectors.

Despite its widespread adoption, the conventional centralised cloud processing architecture has several limitations when considering practical scenarios.

Firstly, the reliance on transferring a large volume of data from edge devices to the cloud

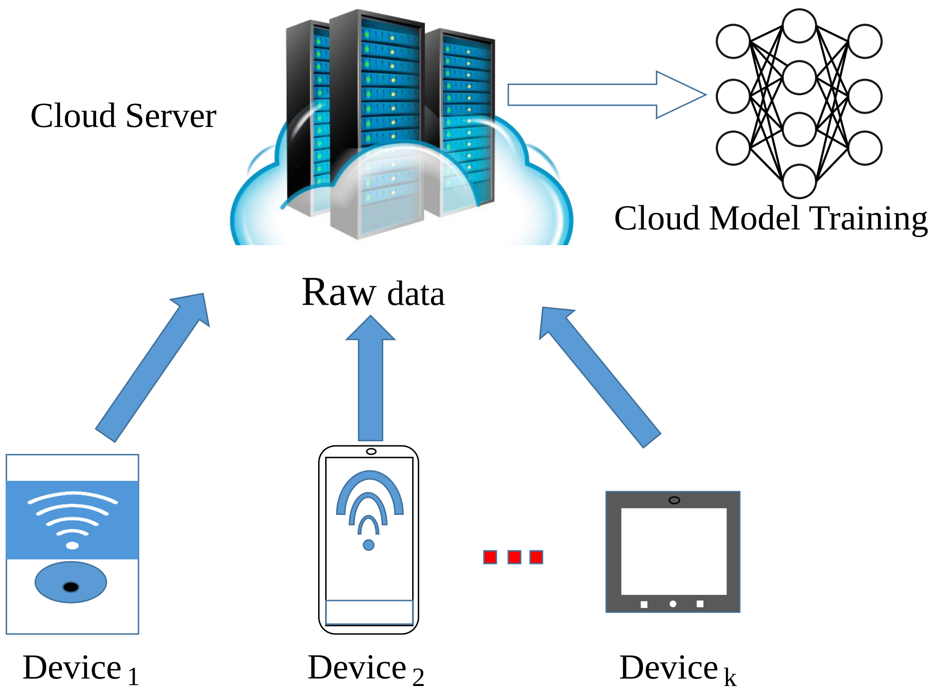


Figure 3.1: Typical data flow and analysis in IoT systems.

for analysis introduces significant challenges related to communication channel capacity and system latency. This becomes particularly problematic for IoT systems that utilise low-power, low-data-rate communication protocols such as Zigbee and LoRa [5].

Secondly, the nature of the data collected by embedded sensor devices often involves privacy-sensitive information, making the transmission of raw data over potentially insecure network connections a significant concern. Data breaches can have severe consequences for the data owners and are strictly regulated by international legislation such as the European General Data Protection Regulation (GDPR) [6].

Furthermore, the increased processing capacity of edge devices remains underutilised as they are primarily used for data gathering and transmission purposes, neglecting their potential for local computation and analysis.

3.1.1 Motivation

In 2017, Google researchers introduced an enhancement to the traditional centralised data flow architecture by proposing Federated Learning (FL) as an innovative distributed model learning framework that operates across the entire system [7], [8]. FL revolutionises the model training process by ensuring that raw data remains localised on the device where it is generated, thereby preserving privacy and enhancing data security. In this approach, a global model is periodically updated by a centralised server in the cloud and shared with edge devices during each training iteration. Each device independently refines the global model using its local dataset and transmits only the updated model parameters back to the cloud server. The server then aggregates these model parameter updates from all participating devices to perform a global model update. This iterative process continues until the global model converges based on a predefined threshold value. Figure 3.2 illustrates the structure of a typical FL system with a central coordinating server.

Although the initial FL architecture represents a substantial advancement, particularly in terms of privacy and overall system efficiency, it has notable limitations, primarily stemming from its centralised design. One significant drawback is the central server, which serves as a singular point of failure, compromising the system's resilience. This vulnerability is especially concerning for time-sensitive IoT applications, as any temporary server downtime can lead to severe disruptions. Additionally, the centralisation of data, even if limited to transmitting model updates, escalates the risk of malicious attacks [9].

In this work, a fully decentralised FL architecture is proposed and analysed to overcome the centralised FL limitations. In our system, there exists no central coordinating entity, such as a cloud server, and the model learning process is fully distributed among network participants, namely the edge devices. More specifically, each individual edge device is responsible for maintaining and updating its local model. Unlike the traditional centralised FL architecture, there is no overarching global model in our approach. During each training iteration, every device shares its current local model parameters with neigh-

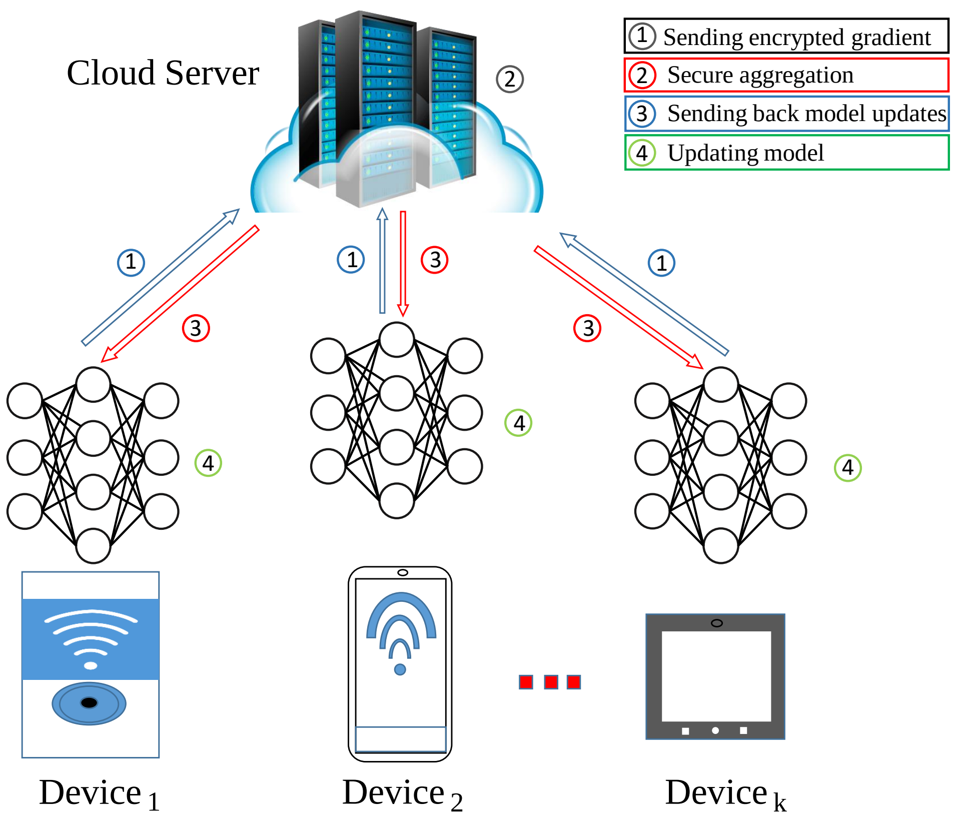


Figure 3.2: Centralised FL system architecture.

bouring devices and receives their respective updates in return. Subsequently, it performs a local model update using its unique dataset. The device then aggregates the received model parameters with its own, calculating the local model parameters for the next iteration. This decentralised learning architecture has been empirically demonstrated to converge to an equivalent central model through the application of gossip training theory in distributed network learning [10] and [11], as illustrated in Fig. 3.3.

In the realm of Decentralised Federated Learning (DFL) on edge devices within wireless mesh networks (WMNs), we believe that our paper makes significant contributions on several fronts. Our primary focus is on reducing communication overhead through the application of cutting-edge compression techniques. Finally, we believe this work makes significant contributions in the area of Federated Learning and network performance analysis.

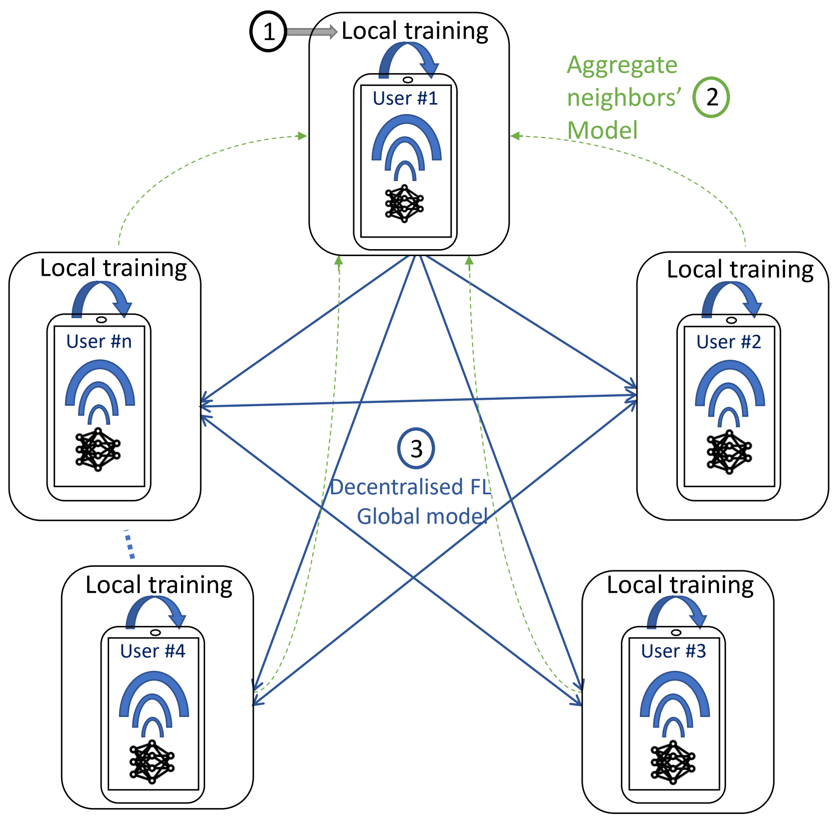


Figure 3.3: Decentralised FL system architecture.

3.1.2 Key Contribution

In this paper, we present a comprehensive description of the DFL architecture within the context of the Internet of Things (IoT) and edge devices while emphasising the significance of the underlying core communication network. Our unique approach is distinct from much of the existing literature, as we aim to provide an all-encompassing perspective on the decentralised FL paradigm while considering the performance implications of the communication network that facilitates data transfer among IoT edge devices. We have chosen wireless one-hop mesh networking as the foundational network infrastructure, aligning with the distributed nature of IoT network learning. The rationale behind this choice is twofold.

Firstly, mesh networking possesses the adaptive capability to reconfigure their routing graph in real-time, particularly when faced with link interruptions [12], [13]. This adaptive characteristic proves invaluable in real-world scenarios where edge devices are often

Table 3.1: An assessment of surveys exploring CFL and DFL studies

.

Ref.	**Year**	**FL Appr-oach**	**Device & Area**	**Various Aggreg-ators**	**Frame-works**	**Appli-cation**	**Focus and Solution Categorisation**
[14]	2020	CFL	Mobile networks	✓	✓	×	A survey on FL and edge computing in mobile networks.
[15]	2021	CFL	IoT devices	×	×	✓	A review of FL in the context of IoT, data distribution, privacy and ML models architectural.
[16]	2021	CFL	IoT devices	✓	×	✓	A systematic review of FL in IoT, detailing limitations and applications.
[9]	2021	CFL	Security & Privacy	×	✓	×	A survey of FL security and privacy, defining secure protocols.
[17]	2022	CFL	FL baselines	✓	✓	?	A survey on the FL baselines with a basic introduction to definitions and architectures.
[18]	2022	CFL	Healthcare	✓	×	✓	An overview of the implement of FL as a tool in the healthcare scenarios.
[19]	2022	CFL	Framework review	✓	✓	?	An overview of frameworks for deploying DFL-based architectures.
[20]	2022	DFL	DLT	×	×	?	A description of the challenges and applications of Blockchain.
[21]	2022	DFL	IoV	×	✓	?	A brief description of FL in fog radio access networks.
[22]	2022	DFL	Wireless comms	×	×	✓	A review of the techniques for adapting FL to distributed environments.
[23]	2022	DFL	UAV devices	×	×	✓	A brief survey on the application of FL in UAV networks.
[24]	2023	DFL	DFL baselines	×	×	✓	A study of optimised DFL models and algorithms focusing on network topologies.
[25]	2023	DFL	DFL baselines	×	?	✓	A comparison of CFL and DFL feder-ation architectures regarding topologies, privacy, and security.
This work	2023	DFL	Edge and IoT devices	✓	✓	✓	Analysed CFL and DFL fundamentals, communication management, geometric network analysis and employed diverse aggregator methods.

deployed in challenging environments, prone to link failures. By harnessing the dynamic nature of the mesh networks, we ensure consistent and reliable data transfer, even in the presence of intermittent connectivity or link disruptions. This resilience is critical to maintaining the integrity of the decentralised FL process, enabling the system to gracefully handle disruptions without compromising learning.

This work combines the DFL architecture with wireless mesh networking, establishing a robust and efficient framework for collaborative learning within IoT systems. Our analysis delves beyond the FL architecture's performance, encompassing the unique attributes of the communication networks. We use one-hop communication mesh networking to ensure efficient data transfer among edge devices. Additionally, we employ different aggregator methods to update the global model and achieve optimum performance. This comprehensive perspective enhances our understanding of the overall system behaviour.

Our research aims to contribute valuable insights into the design and optimisation of DFL systems in IoT environments, promising improved performance and scalability in real-world applications. Moreover, we harness the innate strengths of the mesh networks, such as resilience to single-point failures and heightened data privacy, fortifying the security and reliability of FL within IoT systems. This crucial aspect of our research addresses the mounting complexity and risk factors in large-scale distributed learning applications effectively.

To further enhance our model and minimise communication overhead, we have implemented a state-of-the-art compression technique, as exemplified by the genetic algorithm-based approach [26], to reduce the dimensions of terminal models at the edge. In the case of the Convolutional Neural Network (CNN) model, this reduction entails selecting a subset of convolutional filters and nodes within the dense layers while ensuring that the original models' accuracy levels remain intact.

3.1.3 Organisation

The rest of this report is organised as follows: In Section 3.2, background and related work from literature is presented, and a complete theoretical analysis of the performance of wireless mesh networks and the convergence criteria of the proposed DFL system architecture is carried out. Section 3.4 introduces the methodology for the learning process, network topology, communication methods, and the optimal approach for selecting highly reliable devices to update the global model that is updated using a compressed model at the edge and a range of aggregator methods. In Section 3.5, the simulation setup is described, and the corresponding results are presented along with their interpretation. Section 3.6 presents the results of the simulations and in-depth discussions, offering insights and analysis of the findings. Finally, in Section 3.7 a summary of the work is provided, and future work directions and milestones are proposed.

3.2 Background and Related Work

Ever since its introduction in [7], Federated Learning (FL) has gained considerable attention and has become one of the most extensively researched machine learning paradigms.

The literature surrounding FL is expansive, encompassing diverse architectures, analyses of learning performance, and investigations into data privacy concerns. While the majority of research initially gravitated towards the conventional centralised architecture of FL, recent efforts have increasingly shifted their focus towards decentralised alternatives called decentralised FL (DFL), where basically no central server is needed [27].

Related work on DFL can be distinguished into two main design philosophies. The primer approach to achieving decentralisation involves harnessing blockchain technology, a highly promising avenue. In these blockchain-based systems, participants fall into two categories: standard edge devices and miners. Each edge device establishes communication with nearby miners, who assume the role of model aggregators during particular training rounds. Depending on the employed consensus algorithm, the miner that successfully solves the hashing problem earns the privilege of atomically updating the distributed ledger with the new global model update. In the study [28], the limitations of centralised federated learning (CFL) are addressed by proposing a decentralised federated learning (DFL) approach that eliminates the need for a central server and instead relies on one-hop neighbours for collaboration in the communication network. They use stochastic geometry to model the dynamics of the network topology, MAC protocol, and fading on links, allowing them to evaluate the performance of DFL while preserving privacy and accommodating networking dynamics. However, this study primarily focuses on the evaluation of DFL without considering its application on the edge and evaluating the network intensities for multi-hop wireless mesh networks.

Furthermore, in our research efforts, we have undertaken an extensive assessment of relevant literature and surveys pertaining to both CFL and DFL. This comprehensive evaluation is meticulously presented in Table 3.1, which delves into several facets of

CFL and DFL, including global model aggregation methods, foundational frameworks, application domains, and categorisations based on proposed solutions. To facilitate easy interpretation, symbols have been employed within the table to signify the status of each aspect: a checkmark (✓) indicates full coverage, a question mark (?) denotes partial coverage, and a multiplication symbol (x) signifies that the particular aspect has not been addressed.

In [27], authors aimed at optimising the overall average number of parameter transmissions only in the CFL approach, including shallow and complete transmissions, while maintaining a predefined ratio between them. To offer a thorough analysis of DFL within the context of core communication network performance, Table 3.2 has been compiled to provide an overview of recent developments in CFL and DFL, which serves as an overview of recent developments in the field and their primary focus areas. This table provides a critical evaluation of contemporary advancements in FL network design, spanning multiple dimensions such as resource management, system cost, security, privacy, user distribution analysis, communication network characteristics, FL network intensity, performance, and central server-free approaches.

The works in [29] and [30] investigate the challenges posed by the widespread deployment of Internet of Things (IoT) devices in the 5G era, particularly in the context of software-defined networks (SDNs). It highlights the importance of cache management at the edge of the network and explores emerging edge resources like mobile device clouds and micro-edge data centres. The goal is to optimise content placement based on user demand and cost considerations. The study also addresses security and seamless data delivery in mobile IoT networks and introduces federated learning (FL) as a key framework to harness data and computational resources from end-user devices for training machine learning models. The paper's main focus is on centralised federated learning in the 5G network, leaving potential opportunities in decentralised learning methods, particularly in Ad-hoc networks, relatively unexplored.

Table 3.2: Summary on FL-related topics with our paper's contribution.

Related research	Research area	Assessment of recent developments in the design of FL networks					
		Allocation of resources and cost management	Privacy and security	Analysing the distribution of users	The network communication	FL Network intensity and performance	Central Server-free
[31]	DFL concept	×	✓	×	✓	×	✓
[15]	FL concept	×	✓	×	✓	×	✓
[32]	Distributed ML	✓	×	×	✓	×	×
[9]	Security and Privacy in FL	×	✓	×	×	×	✓
[14]	FL in Edge Networks	✓	✓	×	×	×	×
[16]	FL for IoT	✓	✓	×	✓	×	×
[33]	FL for IIoT	✓	✓	×	×	×	×
[34]	FL for Health Informatics	✓	✓	×	✓	×	×
[35]	decentralised Wireless FL	×	✓	×	✓	×	✓
[36]	DFL framework	✓	✓	×	×	×	✓
[37]	Blockchain-based FL	✓	✓	×	×	×	✓
This work	DFL on the Edge	✓	✓	✓	✓	✓	✓

3.3 Wireless Mesh Network Performance Analysis

In this section, an analytical approach is employed to assess the performance of a wireless mesh network, utilising the physical interference model [38] to quantify the likelihood of successful data transmission between a transmitting node and a receiving node within the network. Our theoretical analysis draws heavily from principles of stochastic geometry and random point processes [39], [40]. According to the physical interference model, the probability of successful transmission hinges on the Signal-to-Interference-and-Noise Ratio (SINR) observed at the receiver. A transmission is classified as successful if the SINR meets or exceeds a predefined threshold value. A notable advantage of the physical interference model is its comprehensive evaluation of total interference emanating from all nodes except the transmitter [41]. Subsequently, we establish the network topology employing the Poisson point process (PPP) theory.

1. **Poisson Point Process**: In our analysis, we assume that the edge devices within the network are distributed based on a stationary homogeneous Poisson Point Pro-

cess (PPP) with an intensity of λ. These devices are distributed within a disk $\mathcal{D} \subset \mathcal{R}^2$ with a radius of R, centred at the origin of the two-dimensional plane $\mathcal{R}^2$. According to the properties of a Poisson point process, the expected number N of devices falling within the disk $\mathcal{D}$ can be calculated as $N = \lambda|\mathcal{D}|$, where $|\mathcal{D}| = \pi R^2$ [39].

An important characteristic of the homogeneous PPP, as per Palm probability theory, is that adding an extra point at the origin in a specific realisation of the process does not affect the distribution of the remaining points in the process (Slivnyak theorem [42]). Consequently, interference statistics can be measured equivalently by assuming that the typical receiver is a point within the process located at the origin [43]. In the network, each device is denoted by i with $1 \leq i \leq N$, where N represents the total number of active devices within the desired receiver coverage area. Additionally, $i \in \varphi$, which φ encompasses all participants within the entire target area.

2. **Successful Transmission Probability**: In the subsequent analysis of the probability of successful transmission, a slotted ALOHA medium access control scheme is considered. In this scheme, each device independently decides to transmit with a probability of p, without coordination with other devices. Additionally, we assume Rayleigh fading for the propagation channel, where the transmission power of each device has a zero mean. The SINR measured at the receiver located at the origin is calculated using the following equation:

$$SINR = \frac{S}{\mathcal{N} + I} \tag{3.1}$$

Here, S represents the signal power emitted by the intended transmitter, $\mathcal{N}$ stands for the noise power, and I corresponds to the cumulative interference power stemming from other transmitters.

For simplicity, we can theoretically assume that the noise power $\mathcal{N}$ is significantly lower than the total interference power. Therefore, we will employ the Signal-to-Interference Ratio (SIR) for the remainder of our analysis, as defined below:

$$SIR = \frac{S}{I} \tag{3.2}$$

The received signal power S_i at the receiver from a transmitter i is [39], [41]:

$$S_i = P_i h r_i^{-\alpha} \tag{3.3}$$

In this context, P_i represents the transmission power of transmitter i, h denotes the fading factor in accordance with the Rayleigh fading model, r_i stands for the distance from transmitter i to the origin, and α characterises the path loss parameter, which reflects the attenuation of signal power with distance. The total interference power I observed at the receiver results from the summation of all S_i values, where i corresponds to all transmitting devices except the intended transmitter. According to the Rayleigh fading model, the received signal power S_i follows an exponential distribution [44], and with the assumption of unit transmission power, its distribution is defined by Eq. (3.4) [39]:

$$f_{S_i}(x) = r_i^{\alpha} \exp\bigg(- r_i^{\alpha} x \bigg), \qquad x \geq 0 \tag{3.4}$$

The successful transmission probability, as per the physical interference model, can be described as the likelihood that the Signal-to-Interference Ratio (SIR) exceeds or equals a predefined threshold:

$$\begin{aligned} p_{succ} &= P(SIR \geq \theta) \\ &= P(\frac{S}{I} \geq \theta) \\ &= P(S \geq \theta I) \end{aligned}$$

$$= \exp(-\theta r^{\alpha} I) \tag{3.5}$$

where r is the distance between the desired transmitter and the receiver.

3. **Laplace Transform of Interference**: The successful transmission probability, as expressed in Eq. (3.5), is equivalent to the Laplace transform of the cumulative interference observed at the receiver when evaluated at $(s = \theta r^{\alpha})$ [39].

$$p_{succ} = \mathcal{L}_I(s)|_{s=\theta r^{\alpha}} = \mathbb{E}\big(\exp(-sI)\big)|_{s=\theta r^{\alpha}} \tag{3.6}$$

Following established analytical methods from stochastic geometry and probability-generating functional, as outlined in references [39], [40], we can deduce a closed-form expression for the successful transmission probability as follows:

$$\begin{aligned}\mathcal{L}_I(s) &= \mathbb{E}\Bigg(\exp(-sI)\Bigg)\\ &= \mathbb{E}\Bigg(\exp(-s\sum_{x\in\Phi} hr_x^{-\alpha})\Bigg)\\ &= \mathbb{E}_{\Phi}\left(\prod_{x\in\Phi}\mathbb{E}_h[\exp(-shr_x^{-\alpha})]\right)\end{aligned}$$

This leads to the following expression for the successful transmission probability:

$$p_{succ} = \exp\left(-\lambda p\pi R^2 s^{\frac{2}{\alpha}}\frac{\frac{2\pi}{\alpha}}{sin(\frac{2\pi}{\alpha})}\right) \tag{3.7}$$

Here, p represents the probability of an individual transmitter deciding to transmit independently (ALOHA), (λ) stands for the intensity of the Poisson Point Process (PPP), R signifies the radius of the PPP disk, (r) denotes the distance between the transmitter and the receiver, and (α) represents the path loss parameter.

Consequently, from Eq. (3.6) and (3.7), we can derive the following expression for the successful transmission probability:

$$p_{succ} = \exp\Bigg(- \lambda p \pi R^2 r^2 \theta^{\frac{2}{\alpha}} \frac{\frac{2\pi}{\alpha}}{sin(\frac{2\pi}{\alpha})}\Bigg) \tag{3.8}$$

3.4 Methodology

Our system model encompasses various critical perspectives to optimise decentralised learning: We start from a system perspective, optimising a decentralised model by collaborating between several distributed edge devices without direct access to their local data. Communication among devices occurs in a peer-to-peer manner, eliminating the need for a central server. From a spatial perspective, we leverage geometric patterns to efficiently manage multi-user communication. Each communication round identifies successful transmitter devices based on interactions with neighbours. Considering convergence, our approach incorporates theoretical analysis to define the target convergence state of the model. Furthermore, we introduce novel aggregator methods and employ Hidden Markov Models (HMM) for device evaluation. Historical performance guides the selection and weighting of edge devices in the learning process.

3.4.1 System Architecture

In a typical Federated Learning (FL) training process, the following three key steps are involved. It's important to distinguish between the *local model*, which is the model trained on each participating device, and the *global model*, which is the model aggregated by the FL server. The following are the main learning steps:

1. **Task Initialisation (Step 1):** At the beginning, the server determines the training task, which refers to the specific application and its data requirements. The server also sets certain hyperparameters for the global model, such as the learning rate. Afterwards, the server shares the initial global model $\mathbf{W}_G^0$ and the task details with selected participants.

2. **Local Model Training and Update (Step 2):** Building on the current global model denoted as $\mathbf{W}_t$ (with t as the iteration index), each participant individually

utilises their local data and device to update their local model parameters, denoted as $\mathbf{W}_t^i$. In each iteration, participant i strives to find optimal parameters, $\mathbf{W}_t$, that minimise the loss function $L(.)$. The loss function varies depending on the problem and the model employed, as illustrated in Table 3.3. In essence, they aim to find $\mathbf{W}_t^{i*}$ such that it minimises $L(.)$. After updating, the local model parameters are then transmitted back to the server.

3. **Global Model Aggregation and Update (Step 3):** In this final step, the server aggregates the local models received from all participants. After aggregating, the server generates updated global model parameters $\mathbf{W}_G^{t+1}$ and sends them back to the respective data owners.

In contrast to traditional Centralised Federated Learning (CFL) systems, the proposed Decentralised Federated Learning (DFL) architecture distinguishes itself by eliminating the need for a central aggregating server. Our target model revolves around a network of edge devices communicating through a wireless mesh infrastructure. The primary objective of this system is the collaborative optimisation of parameters $\mathbf{W}$ for a global model represented as $\hat{y} = f(\mathbf{W}, x)$, where $\hat{y}$ represents the model's predicted output, and x denotes input data. Each individual device possesses its distinct dataset $\mathcal{D}_i$ consisting of input data x_i, and this dataset remains private, not shared with other devices within the network.

The local loss function at each device can be defined as:

$$L_i(\mathbf{W}) = \frac{1}{|\mathcal{D}i|} \sum_{x,y \in \mathcal{D}_i} l\Big(\hat{y} - y; \mathbf{W}, x\Big) \tag{3.9}$$

In this equation, $|\mathcal{D}_i|$ represents the size of the local dataset, and $l(\hat{y} - y)$ is the loss function that quantifies the disparity between the model's predicted output and the actual output corresponding to input x. It is important to note that we assume the local loss function to be both convex and smooth.

At the outset of each training iteration t, let $\mathbf{W_{t_i}}$ represent the local model weights for each device i. Employing its local dataset, each device engages in Stochastic Gradient Descent (SGD) [7] on the local loss function. The device subsequently updates its local model weights using the following equation:

$$\mathbf{W_{t_i+1}} = \mathbf{Wt_i} - \mu \nabla L_i(\mathbf{W_{t_i}}) \tag{3.10}$$

Here, μ denotes the learning rate, carefully selected to ensure the convergence of the SGD algorithm to a minimum.

Model Name	**Loss Function**
Neural Network [45]	(classification) $$L_i(\mathbf{W}) = \frac{1}{\lvert\mathcal{D}_i\rvert} \sum_{x,y \in \mathcal{D}_i} l\Big(\hat{y} - y; \mathbf{W}, x\Big)$$
Linear Regression [46]	Mean Squared Error (MSE): (Regression) $$L_{\mathrm{LR}}(y, \hat{y}) = \frac{1}{2}(y - \hat{y})^2$$
K-means [47]	Sum of Squared Distances: (Clustering) $$L_{\text{K-means}}(x, c) = \sum_{i=1}^{K} \lvert\lvert x - c_i \rvert\rvert^2$$
Squared-SVM [48]	Squared Hinge Loss: (Binary classification) $$L_{\mathrm{SVM}}(y, \hat{y}) = \max(0, 1 - y \cdot \hat{y})^2$$

Table 3.3: Loss functions for different models

In the subsequent phase, each device transmits its recently updated local model weights $\mathbf{W_{t_i+1}}$ to its immediate one-hop neighbours within the wireless mesh network. Simultaneously, it receives updated local model weights from its corresponding one-hop neighbours. Subsequently, each device executes a local aggregation process on these received local model weights, typically involving straightforward averaging. This results in the creation of the initial updated local model weights that will be utilised in the subsequent iteration [49].

The process outlined above facilitates the "diffusion" of each device's local model weight parameters throughout the network during each iteration. Essentially, this diffusion mechanism involves the dissemination of the impact of each device's local training dataset across the network through the transmission of local model weights. Notably, prior research, such as [11], has demonstrated that this collaborative learning network converges to the same global optimum and at a similar convergence rate when compared to a conventional centralised cloud-server approach.

The main objective of DFL is to discover model parameters ($\mathbf{W}$) that minimise the average loss function (also known as an object function or cost function) across all neighbour participating devices as follows:

$$\min_{\mathbf{W}\in\mathbb{R}^d} L(\mathbf{W}_t) = \frac{1}{N}\sum_{i=1}^{N}\zeta_i L_i(\mathbf{W}_t) \tag{3.11}$$

In this context of the loss function in Eq. (3.11), each device indexed as i is assigned a weight denoted by $\zeta_i > 0$. In practical scenarios, these weights ζ_i are typically determined in proportion to the amount of data residing on each respective device. This means that devices with more data contribute more significantly to the overall objective, as represented by the optimisation problem expressed in Eq. (3.11). Furthermore, we assume that the edge devices are capable of running the model within a certain time slot at each epoch.

3.4.2 Communication System

In our model, the Wireless Mesh Networks (WMNs) approach is proposed as a fundamental communication technique that allows devices to share their model parameters in a peer-to-peer manner. Our proposal results from the popularity that WMN have gained due to their cost-effectiveness, which makes them an attractive option for wireless connectivity in the DFL network. WMNs exhibit dynamic self-organisation and self-configuration, allowing network nodes to establish and maintain mesh connections autonomously. This characteristic bestows several advantages upon WMNs, including low initial expenses, efficient network upkeep, resilience, and consistent service coverage [12]. In addition, this low-cost WMN infrastructure is well-suited for establishing a DFL network that can span across community networks, metropolitan areas, municipalities, and enterprise networks.

3.4.3 Learning Convergence Criterion

Consider a scenario with N edge devices participating in the learning network. For this analysis, we make the assumption that these edge devices are distributed according to a homogeneous PPP (Poisson Point Process) with an intensity measure denoted as λ. Furthermore, these devices are confined within a circular region $\mathcal{D}$ centred at the origin and having a radius of R. Within the scope of a typical receiver positioned at the origin, the probability of successful data transmission from a transmitter located at a distance r from the origin can be determined using Eq. (3.8).

The devices that successfully transmit data to the receiver also follow a homogeneous PPP, but with an intensity measure of λp_{succ} [45]. Consequently, the number of devices that succeed in transmitting their data to the receiver, denoted as $\tilde{N}$, can be expressed as:

$$\tilde{N} = |\mathcal{D}|\lambda p_{\text{succ}} = \pi R^2 \lambda \exp\left(-\lambda p \pi R^2 r^2 \theta^{\frac{2}{\alpha}} \frac{\frac{2\pi}{\alpha}}{\sin\left(\frac{2\pi}{\alpha}\right)}\right) \tag{3.12}$$

The distribution of the number m of devices successfully transmitting can be described as [45]:

$$P(m = l) = \frac{\exp(-\tilde{N})\tilde{N}^l}{l!} \tag{3.13}$$

Let $m_{t,j}$ represent the number of devices successfully transmitting their updated local model weights to the receiving device j during training iteration t, resulting in the global model $\mathbf{W}_j$. Additionally, let N_j denote the total number of training iterations out of t, in which at least one device successfully transmitted local model weights to receiver j.

We can now introduce the convergence condition for the model training procedure within the DFL network, adapted from [45] and configured to suit the described decentralised architecture:

$$\arg\min_{\mathbf{W}_j} \left(\frac{1}{N_j} \sum_{j=1}^{N_j} \|\nabla L_j(\mathbf{W_t})\| \right) \leq \epsilon_0 \tag{3.14}$$

The learning convergence condition can be expressed as follows: The network achieves convergence after R training iterations when the maximum expectation of the average gradient, taken across all participating devices denoted as N_j, does not exceed a predefined convergence threshold ϵ_0. This expectation is calculated with respect to the distribution of the input dataset. In essence, this convergence criterion ensures that if even the device with the poorest performance, in terms of the expected average gradient after R rounds, meets the convergence threshold, then all other devices should also meet it. In such a scenario, the DFL network is considered to have converged to the optimal model weight parameters.

3.4.4 Device Selection and Models Aggregator Method

In the context of FL, participant selection is a crucial aspect as it determines which edge clients or devices in the network will contribute to the collaborative model training process. In the literature, different methodologies are used to evaluate the distributed devices and choose the most appropriate group based on the required purpose. In FL, Hidden Markov Models (HMMs) [50], which are probabilistic models widely used in various fields, can be utilised to make well-informed choices concerning participant selection by modelling the past behaviour and performance of devices and thus regularly assign the weights (ζ_i) in Eq. (3.11) for each connected device.

In the DFL approach, the master devices (i.e., main devices) responsible for aggregating models from other neighbours at iteration t must meet specific specifications and requirements to efficiently manage the learning process during that iteration.

Initially, the central server in CFL or main device in the DFL approach initialises a global model w_0 randomly. Subsequently, in each communication round, the following sequence of steps is executed to achieve the learning objective, as illustrated in Fig. 3.2:

1. Step I: Broadcast Latest Model. The central server (in CFL scenario) or the main device (in DFL scenario) disseminates the most recent global model w_t to all clients (neighbours) (typically in cross-silo FL) or a subset of clients (N_t) chosen for participation in the current training round (commonly in cross-device FL).

2. Step II: Clients Compute Local Updates. Each client (i.e., edge device) utilises its compressed proposal model to calculate the model update based on its local dataset by performing multiple iterations of gradient descent; $w^i_{t+1} \leftarrow w^i_{t+1} - \eta \nabla_w L(w^i_t, D_i)$, with η representing the learning rate.

3. Step III: Aggregate Client Updates. The server or the main device updates the global model by combining the local updates using a specific aggregation rule $A(\cdot)$: $(w_{t+1} \leftarrow A(\{w^i_{t+1} : i \in \varphi\})$.

The most widely-used aggregation rule for communication-efficient Federated Learning (FL) is Federated Averaging (FedAvg) [7], which aggregates the client updates by computing a weighted average:

$$w_t \leftarrow \frac{1}{N_t} \sum_{i \in \varphi} w^i_t \tag{3.15}$$

However, FedAvg is not fault-tolerant, and even a single faulty/malicious client can prevent the global model from converging [51]. Although this work does not specifically address malicious attacks and model protection, it is important to mention that there are existing robust aggregation techniques designed for these purposes:

Krum, as described in [52], operates in each communication round by selecting m local model updates out of the total available N_t updates to compute the global model update. This selection is based on comparing the similarity between these local updates. Assuming we have f clients out of N_t are malicious, Krum assigns a score to each local model update w_i by calculating the sum of Euclidean distances between w_i and the m nearest neighbouring local updates among $N_t - f - 2$. The m local model updates with the smallest scores are then chosen, and their average is calculated to determine the global model update.

Median [51]: The Median aggregation method is a coordinate-wise aggregation rule that operates independently on each model parameter. To determine the ith parameter of the global model update, the server arranges the ith parameter values from the submitted N_t local model updates in ascending order and selects the median value. Median aggregation can achieve an order-optimal statistical error rate when dealing with strongly convex loss functions.

Various other aggregation methods have been proposed apart from the previously mentioned ones. For example, Bulyan [53] employs an iterative approach with aggregation rules like Krum for enhanced robustness, but it suffers from computational inefficiency and lacks scalability. Zeno [54] assigns scores to updates and aggregates the top $N_t - b$ updates with the highest scores, where N_t represents the total number of clients, and b is a predefined hyperparameter, typically set equal to or greater than the number of malicious clients.

Another recent approach uses variations of auto-encoders to project client updates into a latent space for malicious update detection, but it relies on the unrealistic assumption of having access to data matching the client's private data distribution for training. Some studies focus on achieving robust federated learning by identifying and blocking malicious clients through adaptive model quality estimation [55] or clustered federated learning [56].

However, it is important to note that some of these methods have predominantly been tested in the context of centralised federated learning, specifically in the straightforward cross-silo scenario. Their applicability to the more complex and dynamic cross-device scenarios, which are characteristic of DFL approaches, has seen limited exploration. This limitation is part of our work to investigate.

3.5 Model Simulation

The Simulation section of this work encompasses the following components:

(i) Simulation of a Wireless Mesh Network: We simulate a wireless mesh network in which participants are distributed according to a Poisson Point Process (PPP), as discussed in our theoretical analysis of wireless mesh network performance. This simulation allows us to model communication dynamics and evaluate network performance across various scenarios.

(ii) Simulation of the baseline Centralised Federated Learning (CFL) architecture: Our simulations replicate the baseline centralised FL architecture. In this setup, a multi-layered Convolutional Neural Network (CNN) is subjected to compression using the state-of-the-art genetic algorithm-based approach (SOTA compression method). The EMNIST benchmarking dataset [57] is employed for training and validation. The primary objective is to minimise communication overhead while enhancing the performance of the compressed CNN model for categorical digit classification with the existence of a central server. This improvement is achieved by enabling collaborative learning among multiple participants within the CFL framework.

(iii) The simulations evaluated the proposed DFL architecture, which incorporates a state-of-the-art compression method. Specifically, a multi-layered Convolutional Neural Network (CNN) was compressed using a genetic algorithm-based approach at the edge. Additionally, various aggregation methods, such as Krum and Median, were employed, mirroring the approach used in the centralised setup. These simulations facilitated a performance comparison between the centralised and decentralised architectures, allowing us to assess the effectiveness and potential advantages of the DFL approach, which has no need for a central server and network infrastructure, taking into account the communication overhead and the complexity. Importantly, this work introduces the novel application of Krum and Median aggregation methods within the realm of DFL.

Through these simulations, we aim to gain insights into the performance, accuracy, and efficiency of both CFL and DFL architectures within the context of wireless mesh networks. The simulation results will provide valuable information for understanding the feasibility and practical implications of implementing DFL in real-world scenarios.

Moreover, the primary objective of the simulation component in this project was to explore the relative performance of two FL architectures, (a) conventional centralised (CFL) and (b) fully decentralised (DFL), while considering the success probability of each data transmission between any two network participants. Notably, this work represents the first instance of providing system simulations where each communication step is completed with a specific success probability, accounting for the underlying communication network's performance within the physical interference model. These results measure the realistic performance of practical FL systems, avoiding the assumption of faultless communications.

To incorporate the communication success probability parameter into the algorithms for both FL architectures, we encountered challenges. Existing FL software libraries, such as TensorFlow Federated [58], and PySyft [59], do not offer the flexibility to define precisely how that network participants communicate during the learning process. Therefore, we developed a fully customised software solution based on the FedML FL and torch frameworks, which provided the most flexibility in defining and implementing custom FL network designs [60]. Specifically, after acquiring average successful transmission probabilities for various wireless mesh networking configurations through corresponding simulations, we integrated this parameter into the FL systems (i.e., CFL and DFL systems) simulation. This approach effectively incorporates the communication network aspect into the learning procedure.

In the following subsections, we provide further detailed descriptions of the simulation setups in other related aspects and present the corresponding results.

3.5.1 Wireless Mesh Network Simulation

To simulate the wireless mesh network that constitutes the backbone for communications across both centralised and DFL architectures, we assume that the participating edge devices are placed randomly in a bounded two-dimensional area of a circle according to a PPP. The communication medium is characterised by an inverse square path loss law and by the presence of Rayleigh fading modelled by an exponential random variable.

In the case of the proposed decentralised architecture, it is assumed that each edge device transmits and receives local model updates only from one-hop neighbours with regard to the wireless mesh network connectivity graph. The simulation of this one-hop neighbourhood is implemented by considering a disk $\mathcal{D}$ of radius r_{oh} around the typical receiver and including a communication link for every transmitting device that falls inside this area. In other words, each edge device transmits to other devices with a distance less than r_{oh}. Moreover, the medium access control (MAC) scheme is Slotted Aloha with the probability of transmission p in each slot. It is further assumed that there are k available frequency levels spanning the allocated bandwidth, which are used for concurrent transmissions from transmitters to receivers, with $k = \lambda p \pi r_{oh}^2$ equal to the mean value of the number of transmitting devices falling inside the one-hop neighbourhood disk area of the typical receiver. Therefore, the desired one-hop-neighbouring transmitting devices do not interfere with each other with respect to the typical receiver.

As previously stated, the aim of the network simulation is to find the mean value of the successful transmission probability between a transmitting node placed inside the area of the circle and the typical receiver situated without loss of generality at the origin of the two-dimensional plane. In order to calculate the target probability, the following procedure is followed:

1. Generate a Poisson distributed for a random number of edge devices inside a disk of radius R with a mean value equal to the process intensity λ times the area of the disk $A = \pi R^2$.

2. Split all generated edge devices into transmitters and receivers for a specific round of communications, i.e. perform thinning of the main PPP with parameter p.

3. For a typical receiver placed at the origin, find the number of transmitters that are less than r_{oh} distance away.

4. Calculate the Signal to SINR for each of the transmitting devices, considering as interference all transmitters that are outside the one-hop neighbourhood radius. If the SINR is above the threshold γ the transmission is considered successful.

5. Repeat the above steps for N rounds and calculate the successful transmission probability for a specific threshold γ as the total number of successful transmissions over the total number of transmissions.

3.5.2 Centralised Federated Learning (CFL)

In simulating the fundamental CFL architecture as proposed in [7], the central server node is assumed to be positioned at the origin of the two-dimensional plane, as previously described in our wireless mesh network simulation setup. Within this setup, the edge devices participating in the learning process are located within a one-hop neighbourhood region of a disk, utilising some distinct frequency bands for simultaneous transmission. Each training iteration involves the following sequence of actions:

- The central server disseminates the global model to all participating devices.
- Every edge device employs Stochastic Gradient Descent with its own local dataset to adjust the weights of the convolutional neural network and subsequently uploads these updated weights to the central server.
- Finally, the server consolidates the received local models from each edge device using the Federated Averaging, Krum and Median aggregator aggregators algorithm, applying straightforward one of these aggregators methods for the global model update.

Following is a description of the convolutional neural network used for classifying handwritten digits trained on the EMNIST dataset. The input of the neural network is a 28×28 pixels image depicting a handwritten digit in grayscale. The model is compressed using the output layer consisting of 10 outputs corresponding to each handwritten digit, whereas the genetic algorithm is used to minimise the overall model size. The EMNIST benchmarking dataset [57] contains 60K grayscale images of handwritten digits. For simulation purposes, the dataset is split into random-sized parts and allocated among the participating edge devices.

3.5.3 Decentralised Federated Learning (DFL)

In order to facilitate a meaningful comparison of results, the simulation setup for the fully decentralised FL system mirrors the configuration of the baseline centralised system, maintaining consistent setup parameters such as learning rate, stochastic gradient descent batch size, and the density of participating devices. Analytically, for a given edge device density (the total number of participating devices in the learning network), the objective is to train the convolutional neural network with the EMNIST benchmark dataset evenly distributed among all participating devices. Below, we outline the typical training iteration process for the fully decentralised algorithm:

- Initialisation: At the onset of each training round, each local edge device possesses the current convolutional neural network weights.
- Local Model Update: Each device independently performs stochastic gradient descent to update its local model weights using its local dataset.
- Communication and Model Exchange: Following the identification of one-hop neighbours within the wireless mesh network connectivity graph for the specific training iteration (as previously noted, this graph is dynamically reconfigurable), each device initiates a broadcast. During this broadcast, it shares its intermediate updated local model with these neighbouring devices and, in turn, receives updated local models from them.
- Aggregation with Robust Methods: Each edge device aggregates the received local

models with its own, employing a straightforward non-weighted average of each model weight, akin to Federated Averaging for the centralised case. Additionally, our work extends beyond traditional aggregation methods and evaluates the model using various robust aggregator methods, such as Krum and Median. This multi-faceted approach aims to enhance performance and resilience in the face of adversarial behaviour, contributing to the robustness of the DFL framework.

This training iteration cycle, orchestrated collaboratively by the participating edge devices, enables the fully DFL system to continually refine its global model. Importantly, it mirrors the core concept of FL, leveraging the collective knowledge of decentralised devices while preserving data privacy and security. The synchronisation and aggregation of local models among neighbouring devices foster collaborative learning without the need for a centralised server, emphasising the robustness and decentralised nature of this approach.

Furthermore, the powerful benefit in our DFL model, particularly in high-intensity networks, is designed to leverage the collective computational power of edge devices efficiently. By distributing tasks, minimising data transfer, and promoting parallel processing, this approach inherently leads to reduced latency, making it well-suited for scenarios where low-latency responses are essential.

3.6 Results and Discussion

In this section, we provide a detailed overview of the simulation parameters and present the results of our wireless mesh network simulation.

For the wireless mesh network simulation, we selected a disk radius of $R = 500$ meters, defining the total simulation area. The intensity of the Poisson Point Process (PPP), representing the density of participating edge devices, was varied, with values $\lambda = 10^{-3}, 5 \times 10^{-3}, 10^{-2}, 5 \times 10^{-2}, 10^{-1}$, resulting in average numbers of participating devices $(\tilde{N})$ is varied.

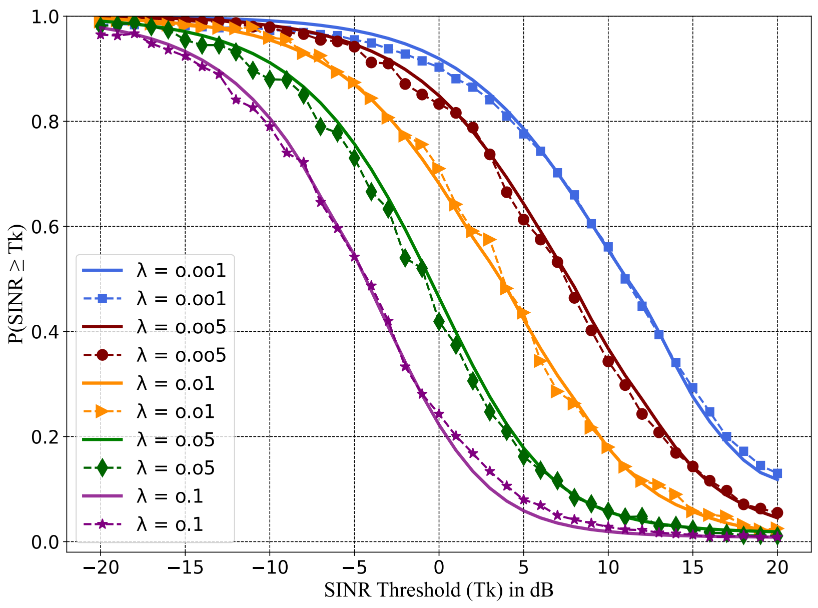


Figure 3.4: The theoretical (solid lines) and simulation (dashed with markers) successful transmission probability as a function threshold values γ in dB.

To emulate the behaviour of devices following the slotted Aloha Medium Access Control (MAC) scheme, we set the probability of a transmitting device deciding to transmit in a specific slot to $p = 0.3$. The one-hop neighbourhood disk radius was set to $r_{oh} = 200$ meters.

We conducted SINR calculations for several threshold (γ) values, spanning from -20 dBm to +20 dBm, with $N_{sim} = 10^5$ simulation rounds for each threshold setting. Additionally, we assumed that each device transmitted with the same power level of $P_{tx} = 1$ Watt.

The diverse set of simulation parameters employed here enables us to comprehensively explore the performance of our wireless mesh network under various conditions. By systematically varying λ and γ, we gain insights into the network's robustness, scalability, and reliability, shedding light on its behaviour across different device densities and signal-to-noise environments.

This detailed analysis serves as a strong foundation for our subsequent discussion and allows us to draw meaningful conclusions about the suitability of our proposed approach for real-world scenarios.

With regards to the FL network training parameters for both centralised and decentralised

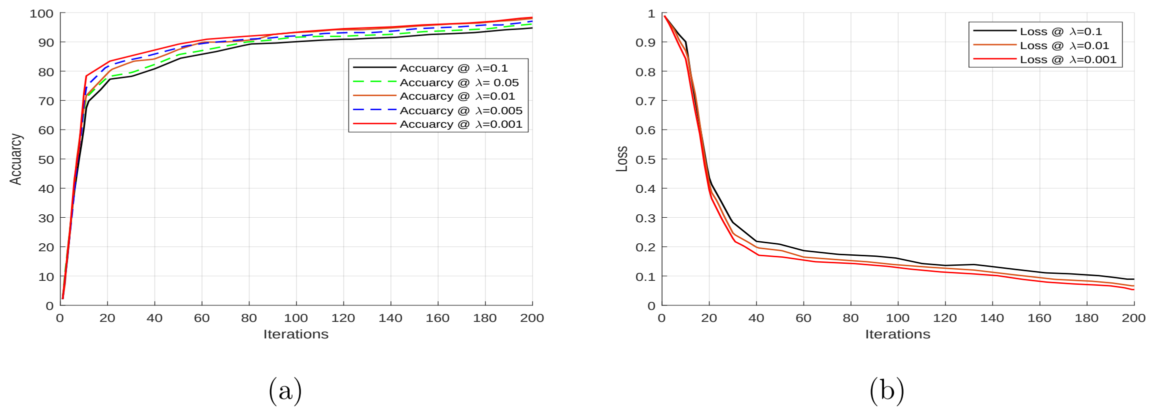


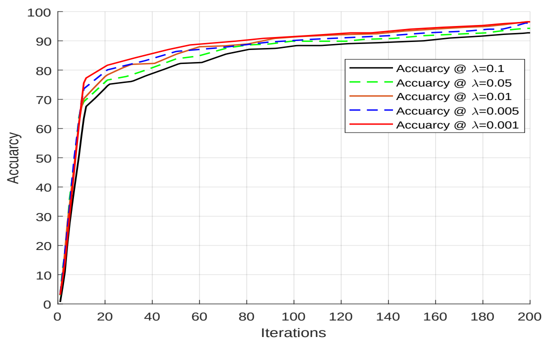
Figure 3.5: Performance metrics for CFL for various network intensities, (a) Model accuracy and (b) Model cross-entropy loss

architectures, the EMNIST dataset was initially split into training and validation sets, comprising 50,000 and 10,000 samples, respectively. In configuring the stochastic gradient descent algorithm's hyperparameters, we selected a learning rate (μ) of 0.015 and a batch size (n_{batch}) of 32 samples. The performance metrics under consideration include model accuracy and cross-entropy loss, specifically tailored for categorical data.

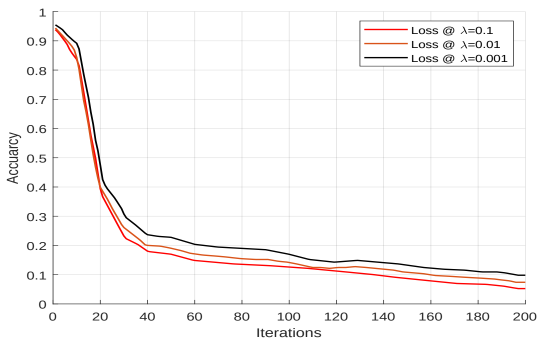
In Figure 3.4, we present the calculated successful transmission probability as a function of varying γ threshold values, encompassing different PPP intensity values in both theoretical and simulation contexts. A discernible trend emerges, showcasing that as the participating device density increases, the resulting mean value of the successful transmission probability experiences a non-linear decrease. This relationship can be attributed to the escalating total interference power at the typical receiver, resulting from more devices transmitting data, driven by a constant slotted Aloha transmission probability (p).

It's noteworthy that, in both CFL and DFL systems, we adopted an SINR threshold (γ) value of -16 dBm as the criterion for deeming a transmission successful. This choice of threshold ensures that our evaluation remains consistent across different scenarios, providing a clear basis for comparative analysis.

Below in Table 3.4, the convolutional neural network model accuracy and cross-entropy loss for centralised and DFL architectures are presented for increasing model training epochs and device density $\lambda = 0.01$.



(a)



(b)

Figure 3.6: Performance metrics for the proposed DFL model for various network intensities, (a) Model accuracy and (b) Model cross-entropy loss

The simulations were conducted in 200 epochs (iterations), and the average values for each metric are presented here. In the case of decentralisation, performance metrics are derived by averaging the individual metrics of each participating device across the entire network. It is worth noting that in the centralised scenario, the model achieves a convergence threshold of over 95% model accuracy after an average of 152 training epochs. Conversely, in the decentralised scenario, the predefined threshold is reached after an average of 173 epochs. This represents a 13.8% increase in the required number of training iterations, consequently impacting the overall system latency to the same degree.

The increase in training epochs can be attributed to the diffusion delay introduced by the decentralised architecture. In the decentralised approach, the contributions of each device's local dataset to the local model update are not immediately transferred across the network. This is in contrast to the centralised approach, where a single aggregator collects all individual contributions in each round to update the global model. Thus, this trade-off becomes evident when transitioning to a fully decentralised solution with a realistic model of the core communication network.

Despite the trade-off of increased training epochs, the advantages of data security and resilience make DFL a compelling choice for collaborative machine learning in decentralised settings. This phenomenon highlights the practical modelling of the core communica-

Table 3.4: FL simulation results

Iterations	CFL		DFL	
	Accuracy	**Loss**	**Accuracy**	**Loss**
10	0.7987	0.8743	0.7764	0.9321
20	0.8453	0.4422	0.8351	0.5276
30	0.8662	0.2917	0.8497	0.3344
40	0.8877	0.2587	0.8723	0.2897
50	0.9012	0.2492	0.8903	0.2632
75	0.9249	0.2314	0.9122	0.2471
100	0.9388	0.2265	0.9276	0.2384
125	0.9464	0.2075	0.9304	0.2172
150	0.9497	0.1794	0.9415	0.1964
175	0.9753	0.1623	0.9511	0.1742
200	0.9892	0.1428	0.9759	0.1515

tion between devices in DFL scenarios, where the network topology and communication mechanisms play an important role in shaping the learning dynamics. Understanding and addressing these challenges are essential steps toward optimising DFL frameworks for real-world applications.

Figures 3.5a, 3.5b, 3.6a, and 3.6b illustrate the accuracy and cross-entropy loss of both architectures concerning varying network intensities (λ). These charts clearly demonstrate an inverse relationship between model accuracy and network device density. This outcome emphasises the significance of considering the communication network's performance in the analysis of an FL system. Interestingly, an increased number of devices participating in the learning process, while expected to enhance convergence, actually leads to reduced model accuracy due to elevated interference from additional transmitters.

Furthermore, it is noteworthy that the rate of increase in accuracy until the 90% threshold is nearly identical between the baseline centralised solution and the proposed DFL system. This observation is of particular significance, indicating that by slightly adjusting the convergence threshold criterion, there's no compromise in performance when transitioning to the proposed DFL framework.

Our model employs a genetic algorithm-based approach to optimise model size, making it suitable for resource-constrained IoT and wearable devices. This model compression con-

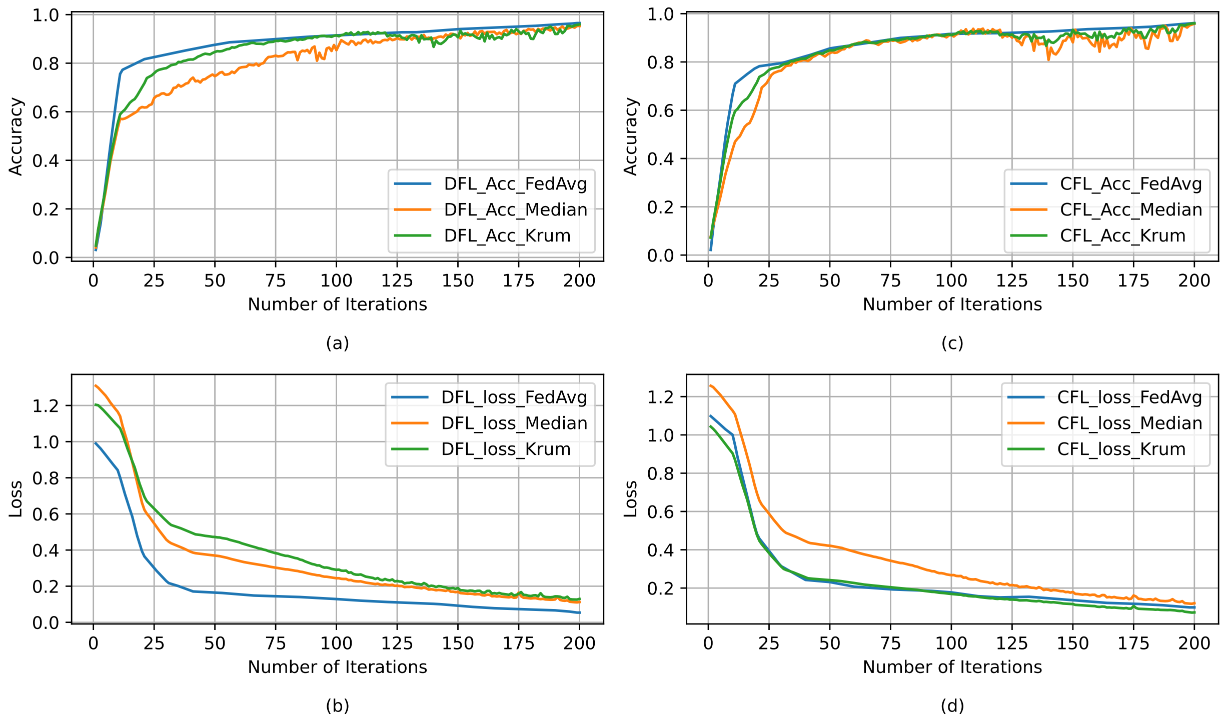


Figure 3.7: Comparative analysis of DFL and CFL performance metrics using various aggregator methods.

tributes to reducing the complexity as the model size decreases, enhancing its applicability in such constrained environments. Despite this reduction and local model compression to minimise communication overhead and complexity, our approach excels in achieving high model performance. It competes effectively with traditional DFL models, as demonstrated in Fig. 3.7. This highlights the efficiency and promise of our approach in striking a balance between model size and performance.

Our study conducts a comprehensive evaluation of the proposed model, considering various aggregator methods applicable across a spectrum of use cases. Notably, we integrate Median and Krum aggregator methods into the DFL framework, making our work a pioneering effort in this regard. Fig. 3.7 presents a comparative analysis of DFL and CFL performance metrics over multiple training iterations, employing a diverse set of aggregator methods, including FedAvg, Median, and Krum. Subplot (a) provides insight into DFL's accuracy trends, while subplot (c) showcases CFL's accuracy trends. Subplots (b) and (d) delve into the corresponding loss trends for DFL and CFL, respectively. This in-depth analysis offers a window into the intricacies of training dynamics and convergence behaviour inherent to both approaches.

Lastly, our results consistently demonstrate that the DFL model consistently achieves accuracy rates exceeding 93% and exhibits lower loss across all our proposed aggregator methods. This makes it a versatile and suitable model for various purposes. These findings emphasise the inherent advantages of DFL, especially in decentralised settings where concerns about data privacy and distribution are paramount. Moreover, they underscore DFL's potential as a robust and versatile framework perfectly suited for collaborative applications across edge devices.

3.7 Conclusions and Future Work

In conclusion, our study explores a contemporary approach to distributed machine learning, providing an alternative to traditional Internet of Things (IoT) model learning systems. We prioritise data security by keeping raw edge device data local and sharing only model parameters. Recognising limitations in CFL systems, such as single points of failure and susceptibility to malicious attacks, we embrace a DFL architecture.

Our work addresses DFL framework limitations through convergence analysis and comprehensive performance assessment of the communication network, utilising wireless mesh networking. Our theoretical analysis employs stochastic geometry to derive a closed-form equation for successful transmission probability in wireless mesh networks. We present a detailed DFL architecture description and training procedure, establishing a learning convergence criterion. Simulations compare our DFL architecture to the conventional CFL model. Using practical slotted Aloha wireless mesh networks and the EMNIST dataset for handwritten digit classification, we extract the successful transmission probability, accounting for potential transmission failures.

Results demonstrate that our decentralised architecture closely matches centralised counterparts in terms of accuracy and average loss. Importantly, our study integrates geometric analysis and diverse aggregator methods (Krum and Median) over compressed models, achieving high performance while significantly reducing the communication overhead. This approach highlights the practicality of decentralised architectures and offers an efficient framework for future IoT systems, potentially scalable in real-world applications.

Future work for this project will focus on the following areas:

(i) Investigation of suitable networking protocols: Further research will explore networking protocols specifically tailored for wireless mesh networks, such as the Thread protocol. The aim is to identify protocols that support low-power, low-data rate transmissions, which are particularly well-suited for IoT system solutions. This research will contribute to the development of more efficient and optimised communication mechanisms within wireless mesh networks.

(ii) Exploration of blockchain-enabled solutions: Future research could investigate the integration of blockchain technology into the compressed DFL architecture. As blockchain technology matures, it presents opportunities for enhancing data security, privacy, and trust in the context of FL. The exploration of blockchain-enabled solutions can contribute to the development of more robust and resilient DFL systems, providing additional layers of data integrity and transparency.

References

[1] L. S. Vailshery, "Iot market size worldwide 2017-2025," *Statista*, Jan. 2021. [Online]. Available: `https://www.statista.com/statistics/976313/global-iot-market-size/`.

[2] A. Afzal, S. A. R. Zaidi, M. Z. Shakir, *et al.*, "The cognitive internet of things: A unified perspective," *Mobile Networks and Applications*, vol. 20, pp. 72–85, 2015.

[3] J. M. Shalf and R. Leland, "Computing beyond moore's law," eng, *Computer (Long Beach, Calif.)*, vol. 48, no. 12, pp. 14–23, 2015, ISSN: 0018-9162.

[4] R. Krishnamurthi, A. Kumar, D. Gopinathan, A. Nayyar, and B. Qureshi, "An overview of iot sensor data processing, fusion, and analysis techniques," *Sensors*, vol. 20, no. 21, 2020, ISSN: 1424-8220. DOI: `10.3390/s20216076`. [Online]. Available: `https://www.mdpi.com/1424-8220/20/21/6076`.

[5] A. Cilfone, L. Davoli, L. Belli, and G. Ferrari, "Wireless mesh networking: An iot-oriented perspective survey on relevant technologies," *Future Internet*, vol. 11, no. 4, 2019, ISSN: 1999-5903. DOI: `10.3390/fi11040099`. [Online]. Available: `https://www.mdpi.com/1999-5903/11/4/99`.

[6] P. Voigt and A. Von dem Bussche, "The eu general data protection regulation (gdpr)," *A Practical Guide, 1st Ed., Cham: Springer International Publishing*, vol. 10, no. 3152676, pp. 10–99, 2017.

[7] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*, PMLR, 2017, pp. 1273–1282.

[8] J. Konečný, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, *Federated learning: Strategies for improving communication efficiency*, 2017. arXiv: 1610.05492 [cs.LG].

[9] V. Mothukuri, R. Parizi, S. Pouriyeh, Y. Huang, A. Dehghantanha, and G. Srivastava, "A survey on security and privacy of federated learning," *Future Generation Computer Systems*, Oct. 2020. DOI: 10.1016/j.future.2020.10.007.

[10] D. Kempe, A. Dobra, and J. Gehrke, "Gossip-based computation of aggregate information," in *44th Annual IEEE Symposium on Foundations of Computer Science, 2003. Proceedings.*, 2003, pp. 482–491. DOI: 10.1109/SFCS.2003.1238221.

[11] A. H. Sayed, S.-Y. Tu, J. Chen, X. Zhao, and Z. J. Towfic, "Diffusion strategies for adaptation and learning over networks: An examination of distributed strategies and network behavior," *IEEE Signal Processing Magazine*, vol. 30, no. 3, pp. 155–171, 2013. DOI: 10.1109/MSP.2012.2231991.

[12] I. F. Akyildiz, X. Wang, and W. Wang, "Wireless mesh networks: A survey," *Computer Networks*, vol. 47, no. 4, pp. 445–487, 2005, ISSN: 1389-1286. DOI: https://doi.org/10.1016/j.comnet.2004.12.001.

[13] D. Benyamina, A. Hafid, and M. Gendreau, "Wireless mesh networks design — a survey," *IEEE Communications Surveys Tutorials*, vol. 14, no. 2, pp. 299–310, 2012. DOI: 10.1109/SURV.2011.042711.00007.

[14] W. Y. B. Lim, N. C. Luong, D. T. Hoang, *et al.*, "Federated learning in mobile edge networks: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 3, pp. 2031–2063, 2020.

[15] D. C. Nguyen, M. Ding, P. N. Pathirana, A. Seneviratne, J. Li, and H. V. Poor, "Federated learning for internet of things: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 3, pp. 1622–1658, 2021.

[16] L. U. Khan, W. Saad, Z. Han, E. Hossain, and C. S. Hong, "Federated learning for internet of things: Recent advances, taxonomy, and open challenges," *IEEE Communications Surveys & Tutorials*, 2021.

[17] P. Boobalan, S. P. Ramu, Q.-V. Pham, *et al.*, "Fusion of federated learning and industrial internet of things: A survey," *Computer Networks*, vol. 212, p. 109 048, 2022.

[18] M. Joshi, A. Pal, and M. Sankarasubbu, "Federated learning for healthcare domain-pipeline, applications and challenges," *ACM Transactions on Computing for Healthcare*, vol. 3, no. 4, pp. 1–36, 2022.

[19] L. Witt, M. Heyer, K. Toyoda, W. Samek, and D. Li, "Decentral and incentivized federated learning frameworks: A systematic literature review," *IEEE Internet of Things Journal*, 2022.

[20] Y. Qu, M. P. Uddin, C. Gan, Y. Xiang, L. Gao, and J. Yearwood, "Blockchain-enabled federated learning: A survey," *ACM Computing Surveys*, vol. 55, no. 4, pp. 1–35, 2022.

[21] M. Billah, S. T. Mehedi, A. Anwar, Z. Rahman, and R. Islam, "A systematic literature review on blockchain enabled federated learning framework for internet of vehicles," *arXiv preprint arXiv:2203.05192*, 2022.

[22] R. Gupta and T. Alam, "Survey on federated-learning approaches in distributed environment," *Wireless personal communications*, vol. 125, no. 2, pp. 1631–1652, 2022.

[23] D. Saraswat, A. Verma, P. Bhattacharya, *et al.*, "Blockchain-based federated learning in uavs beyond 5g networks: A solution taxonomy and future directions," *IEEE Access*, vol. 10, pp. 33 154–33 182, 2022.

[24] J. Wu, S. Drew, F. Dong, Z. Zhu, and J. Zhou, "Topology-aware federated learning in edge computing: A comprehensive survey," *arXiv preprint arXiv:2302.02573*, 2023.

[25] H. Chen, H. Wang, D. Jin, and Y. Li, "Advancements in federated learning: Models, methods, and privacy," *arXiv preprint arXiv:2302.11466*, 2023.

[26] M. Agarwal, S. K. Gupta, and K. Biswas, “Genetic algorithm based approach to compress and accelerate the trained convolution neural network model,” *International Journal of Machine Learning and Cybernetics*, pp. 1–17, 2023.

[27] H.-S. Lee, “Device selection and resource allocation for layerwise federated learning in wireless networks,” *IEEE Systems Journal*, vol. 16, no. 4, pp. 6441–6444, 2022.

[28] A. Salama, A. Stergioulis, A. M. Hayajneh, S. A. R. Zaidi, D. McLernon, and I. Robertson, “Decentralized federated learning over slotted aloha wireless mesh networking,” *IEEE Access*, vol. 11, pp. 18 326–18 342, 2023. DOI: 10.1109/ACCESS.2023.3246924.

[29] V. Balasubramanian, M. Aloqaily, M. Reisslein, and A. Scaglione, “Intelligent resource management at the edge for ubiquitous iot: An sdn-based federated learning approach,” *IEEE network*, vol. 35, no. 5, pp. 114–121, 2021.

[30] A. Mahmod, G. Caliciuri, P. Pace, and A. Iera, “Improving the quality of federated learning processes via software defined networking,” in *Proceedings of the 1st International Workshop on Networked AI Systems*, 2023, pp. 1–6.

[31] H. Ye, L. Liang, and G. Y. Li, “Decentralized federated learning with unreliable communications,” *IEEE Journal of Selected Topics in Signal Processing*, vol. 16, no. 3, pp. 487–500, 2022.

[32] Y. Guo, R. Zhao, S. Lai, L. Fan, X. Lei, and G. K. Karagiannidis, “Distributed machine learning for multiuser mobile edge computing systems,” *IEEE Journal of Selected Topics in Signal Processing*, vol. 16, no. 3, pp. 460–473, 2022. DOI: 10.1109/JSTSP.2022.3140660.

[33] Q.-V. Pham, K. Dev, P. K. R. Maddikunta, T. R. Gadekallu, T. Huynh-The, *et al.*, “Fusion of federated learning and industrial internet of things: A survey,” *arXiv preprint arXiv:2101.00798*, 2021.

[34] J. Xu, B. S. Glicksberg, C. Su, P. Walker, J. Bian, and F. Wang, “Federated learning for healthcare informatics,” *Journal of Healthcare Informatics Research*, vol. 5, no. 1, pp. 1–19, 2021.

[35] S. Chen, D. Yu, Y. Zou, J. Yu, and X. Cheng, "Decentralized wireless federated learning with differential privacy," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 9, pp. 6273–6282, 2022.

[36] C. Che, X. Li, C. Chen, X. He, and Z. Zheng, "A decentralized federated learning framework via committee mechanism with convergence guarantee," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 12, pp. 4783–4800, 2022.

[37] A. Islam, A. Al Amin, and S. Y. Shin, "Fbi: A federated learning-based blockchain-embedded data accumulation scheme using drones for internet of things," *IEEE Wireless Communications Letters*, vol. 11, no. 5, pp. 972–976, 2022.

[38] P. Cardieri, "Modeling interference in wireless ad hoc networks," *IEEE Communications Surveys Tutorials*, vol. 12, no. 4, pp. 551–572, 2010. DOI: 10.1109/SURV.2010.032710.00096.

[39] M. Haenggi, R. K. Ganti, *et al.*, "Interference in large wireless networks," *Foundations and Trends® in Networking*, vol. 3, no. 2, pp. 127–248, 2009.

[40] J. G. Andrews, A. K. Gupta, and H. S. Dhillon, *A primer on cellular network analysis using stochastic geometry*, 2016. arXiv: 1604.03183 [cs.IT].

[41] T. Jing, X. Chen, Y. Huo, and X. Cheng, "Achievable transmission capacity of cognitive mesh networks with different media access control," in *2012 Proceedings IEEE INFOCOM*, 2012, pp. 1764–1772. DOI: 10.1109/INFCOM.2012.6195549.

[42] M. Haenggi, *Stochastic geometry for wireless networks*. Cambridge University Press, 2012.

[43] D. Stoyan, W. S. Kendall, S. N. Chiu, and J. Mecke, *Stochastic geometry and its applications*. John Wiley & Sons, 2013.

[44] B. Sklar, "Rayleigh fading channels in mobile digital communication systems .i. characterization," *IEEE Communications Magazine*, vol. 35, no. 7, pp. 90–100, 1997. DOI: 10.1109/35.601747.

[45] Z. Lin, X. Li, V. K. N. Lau, Y. Gong, and K. Huang, *Deploying federated learning in large-scale cellular networks: Spatial convergence analysis*, 2021. arXiv: `2103.06056 [cs.IT]`.

[46] D. C. Montgomery, E. A. Peck, and G. G. Vining, *Introduction to linear regression analysis*. John Wiley & Sons, 2021.

[47] J. A. Hartigan and M. A. Wong, "Algorithm as 136: A k-means clustering algorithm," *Journal of the royal statistical society. series c (applied statistics)*, vol. 28, no. 1, pp. 100–108, 1979.

[48] J. A. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," *Neural processing letters*, vol. 9, pp. 293–300, 1999.

[49] S. Savazzi, M. Nicoli, and V. Rampa, "Federated learning with cooperating devices: A consensus approach for massive iot networks," *IEEE Internet of Things Journal*, vol. 7, no. 5, pp. 4641–4654, May 2020, ISSN: 2372-2541. DOI: `10.1109/jiot.2020.2964162`.

[50] S. R. Eddy, "Hidden markov models," *Current opinion in structural biology*, vol. 6, no. 3, pp. 361–365, 1996.

[51] D. Yin, Y. Chen, R. Kannan, and P. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," in *International Conference on Machine Learning*, PMLR, 2018, pp. 5650–5659.

[52] P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," *Advances in neural information processing systems*, vol. 30, 2017.

[53] R. Guerraoui, S. Rouault, *et al.*, "The hidden vulnerability of distributed learning in byzantium," in *International Conference on Machine Learning*, PMLR, 2018, pp. 3521–3530.

[54] C. Xie, S. Koyejo, and I. Gupta, "Zeno: Distributed stochastic gradient descent with suspicion-based fault-tolerance," in *International Conference on Machine Learning*, PMLR, 2019, pp. 6893–6901.

[55] L. Muñoz-González, K. T. Co, and E. C. Lupu, "Byzantine-robust federated machine learning through adaptive model averaging," *arXiv preprint arXiv:1909.05125*, 2019.

[56] F. Sattler, K.-R. Müller, T. Wiegand, and W. Samek, "On the byzantine robustness of clustered federated learning," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2020, pp. 8861–8865.

[57] G. Cohen, S. Afshar, J. Tapson, and A. Van Schaik, "Emnist: Extending mnist to handwritten letters," in *2017 international joint conference on neural networks (IJCNN)*, IEEE, 2017, pp. 2921–2926.

[58] M. Abadi, A. Agarwal, P. Barham, *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *arXiv preprint arXiv:1603.04467*, 2016.

[59] T. Ryffel, A. Trask, M. Dahl, *et al.*, *A generic framework for privacy preserving deep learning*, 2018. arXiv: `1811.04017 [cs.LG]`.

[60] C. He, S. Li, J. So, *et al.*, *Fedml: A research library and benchmark for federated machine learning*, 2020. arXiv: `2007.13518 [cs.LG]`.

Chapter 4

Enhancing Byzantine-Resilience in Decentralised Federated Learning for Dynamic Networks

Chapter source: A. Salama, S. A. Zaidi, D. McLernon, and M. M. Qazzaz, "Enhancing Byzantine-Resilience in Decentralised Federated Learning for Dynamic Networks," IEEE Transactions on Machine Learning in Communications and Networking, (Submitted in March, and still under review.

Abstract

Decentralised Federated Learning (DFL) has emerged as a promising approach for privacy preservation in dynamic networks. Unlike traditional data sharing, it facilitates the direct exchange of model parameters among users in a peer-to-peer paradigm. DFL plays a pivotal role in distributed machine learning (ML), presenting a dynamic interplay between communication and ML performance in wireless networks. Nevertheless, Byzantine participants in DFL networks can seriously disrupt the system, potentially leading to model divergence. This study addresses this challenge by enhancing DFL's performance, especially in Byzantine-device scenarios. The purpose is to establish a resilient model over ad hoc networks while employing the CSMA/CA protocol for data transmissions.

Our proposed DFL Byzantine-resilient model, tailored for wireless networks, identifies and eliminates untrusted devices using our novel DFL reputation score technique and leverages ingenious aggregation techniques and spatial analysis methods to maintain throughput. We present two essential metrics to evaluate the DFL network performance: 1) robustness against adversarial attacks and 2) attainable high accuracy and low loss, considering complexity and network dynamics. Through comprehensive simulation utilising the well-known MNIST dataset as a benchmark dataset for such classification problems, even in the existence of up to 35% Byzantine devices within the network, our model was able to achieve 94% accuracy and low loss. Our model acclimates effectively to various adversarial attacks, improving security, convergence rate, and performance. This improvement offers a reliable solution for collaborative learning in distributed networks without a central infrastructure.

Future research may explore scalability and optimisation in scenarios involving a multi-

tude of participating devices, further enhancing DFL systems.

4.1 Introduction

The rise of Intelligent Connected Vehicles (ICVs) is transforming the Internet of Vehicles (IoV), with various online gadgets and edge devices like smartwatches, house assistants, security systems, and autonomous vehicles (AVs) joining the network. Technologies like vehicle-to-everything (V2X) communication through dedicated short-range communication (DSRC) and cellular V2X (C-V2X) have facilitated seamless connectivity among vehicles.

ICVs face challenges in tasks like driving trajectory prediction, traffic flow prediction, and intelligent V2X communication within dynamic environments. Data-driven machine learning (ML) solutions have gained traction to address these complexities, enhancing ICVs' capabilities in predicting driving trajectories, forecasting traffic flow, and enabling intelligent V2X communication.

Equipped with sensors like cameras, GPS, and LiDAR, ICVs and AVs generate real-time data, providing valuable inputs for ML algorithms. However, centralised training approaches are impractical due to communication costs and privacy concerns associated with transferring large volumes of distributed data to a cloud server. Conventional centralised cloud processing architectures face limitations in communication channel capacity, system latency, and privacy-sensitive data transmission over potentially insecure network connections.

Federated Learning (FL) [1] is a promising solution, coordinating the collaborative training of neural network (NN) models across multiple end devices in a distributed manner as shown in Fig. 4.1. Challenges persist in the realm of ICVs, encompassing factors like vehicle mobility, constrained computing and communication resources, and notably, the potential threat posed by Byzantine attackers and untrusted devices within the IoV environments.

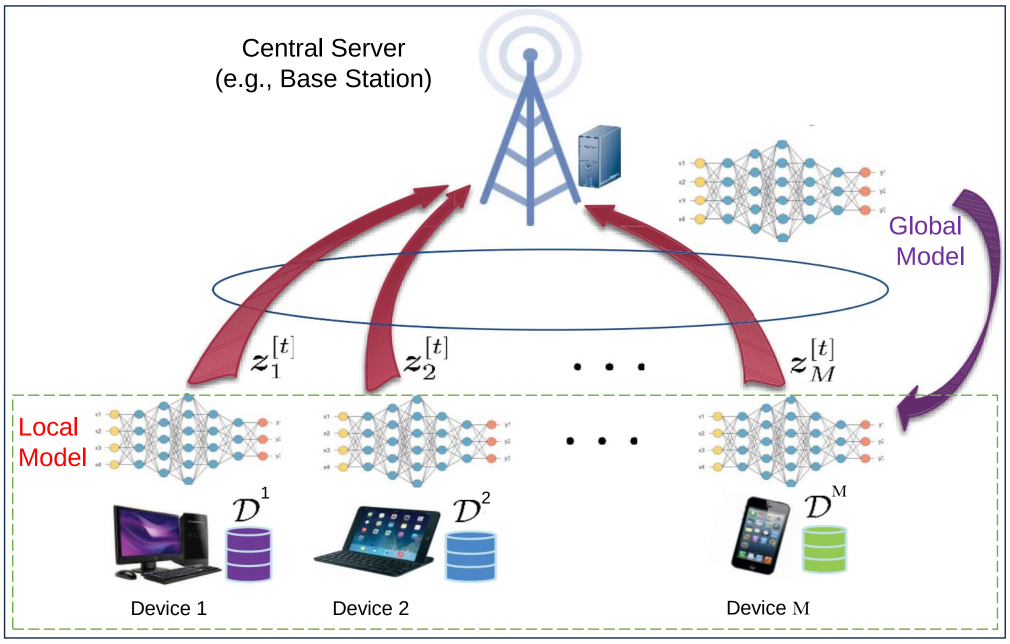


Figure 4.1: An example of a traditional FL framework.

To overcome these challenges, a DFL Byzantine-resilience approach efficiently facilitates real-time data sharing among vehicles without relying on centralised systems. DFL addresses FL limitations, ensuring privacy and security in dynamic environments like AVs and ICV networks. This collaborative strategy presents benefits such as enhanced traffic flow management and optimised parking space allocation. To illustrate both the transformative potential and the challenges associated with this DFL approach, we provide a theoretical analysis and simulations of the DFL framework in the presence of Byzantine and malicious devices in order to optimise real-world network performance.

4.1.1 Motivation

FL effectively reduces server workload and data transmission needs, addressing privacy concerns by enabling ML model training on local data. Centralised Federated Learning (CFL), depicted in Fig. 4.1, poses challenges such as communication bottlenecks and a single point of failure. Moreover, deploying a central server can be complex and costly. This research aims to overcome these limitations by employing DFL, eliminating the need for a central server through peer-to-peer communication.

DFL implementation on wireless nodes introduces challenges influenced by propagation, topology, and medium access dynamics. In DFL, models are shared in a defined neighbourhood in a peer-to-peer manner, accelerating ML task learning locally without a central server. However, simultaneous uplink transmissions may cause interference, extending convergence time for ML algorithms.

To address these challenges, co-designing communication medium access control (MAC) protocols (e.g., CSMA/CA) with DFL is crucial. Optimising the design space could yield significant performance gains. CSMA/CA, the primary MAC protocol in 802.11 networks, may suffer interference in DFL model transmission, impacting real-environment applications.

DFL for AV networks faces robustness concerns, vulnerable to Byzantine attacks during training. Byzantine nodes exhibit potentially malicious behaviour, leading to issues like model corruption or deliberate attacks. Despite Byzantine-resilient FL exploration, decentralised scenarios receive less attention.

Thus, efforts are made to address Byzantine attacks in DFL networks in this work, introducing a Byzantine optimisation strategy to:

(i) Detect Byzantine nodes or clients using anomaly detection methods.

(ii) Mitigate impact through device-weighted and reputation score mechanisms and robust aggregation methods.

(iii) Maintain model accuracy despite Byzantine faults.

This paper proposes a DFL model to enhance the efficiency and application of Autonomous Vehicles (AVs) in the V2X network. The model executes learning processes locally across AV users, preserving privacy by gathering only model parameters from neighbours. The DFL framework benefits traffic flow monitoring and road safety in the V2X topology. Figure 4.2 illustrates a V2X DFL network layout, featuring both trusted and Byzantine clients.

The proposed approach promises performance and safety enhancements even with untrusted devices, extending benefits beyond AV users to pedestrian safety and road in-

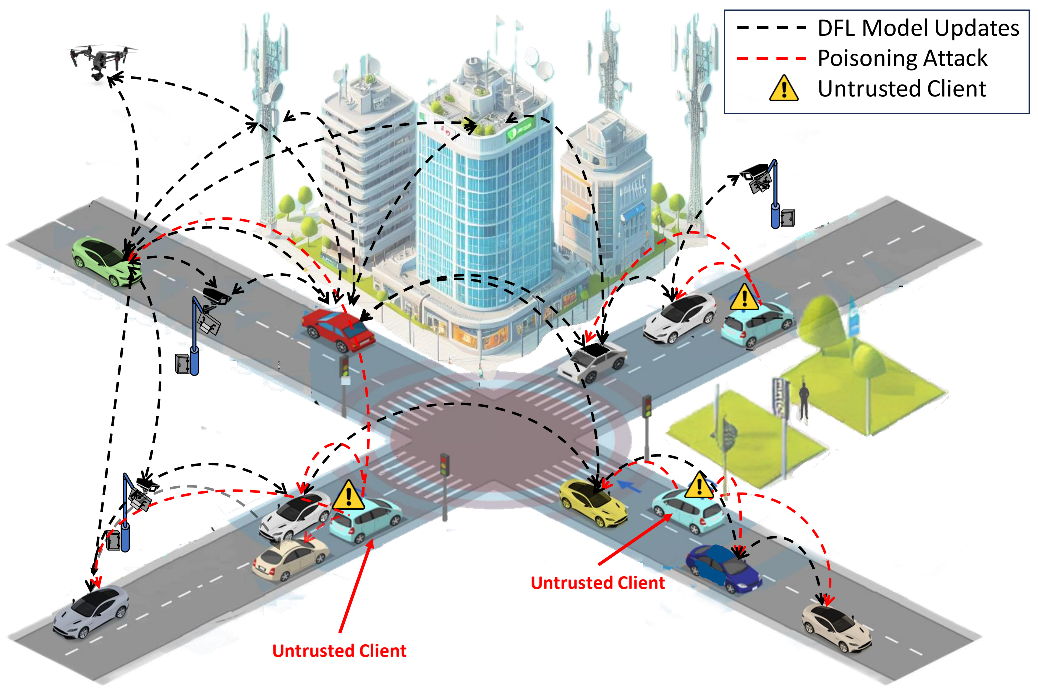


Figure 4.2: An example of V2X DFL network layout that contains trusted and untrusted clients.

frastructure. DFL presents critical benefits in AV applications, including Self-Driving, Advanced Driver Assistance Systems (ADAS), Computer Vision, and the following:

i **Preserving Privacy**: DFL protects user information by keeping local data on devices, addressing privacy concerns.

ii **minimise Latency**: DFL reduces communication costs, specifically latency, by avoiding large data offloading and the need for a central server in dynamic V2X or V2V networks.

iii **Improved Learning Quality**: DFL accelerates training, convergence rates, and scalability by leveraging significant dataset resources from sensors in a large-scale AV network. This surpasses traditional centralised FL methodologies.

4.1.2 Key Contribution

Our research introduces an improved DFL model that is able efficiently to address malicious attacks on Intelligent Connected Vehicles (ICVs) within Autonomous Vehicle (AV) networks. A novel DFL reputation score model is developed, leveraging client activity records as weights in the aggregation process. The robustness of our model is rigorously tested against various scenarios and attack types.

1. We propose an intuitive approach to identify malicious clients attacking the DFL model in ICVs within AV networks. The rarity of certain patterns of gradient similarity across multiple parameters and clients serves as a detection criterion.
2. A reputation score model is employed to track client activity, influencing the aggregation process. Our unique reputation model enhances convergence using a weighted averaging approach.
3. Our DFL model is evaluated on the proposed AV network, outperforming baseline models against adaptive white-box attacks. Resilience is maintained up to a 35% malicious client ratio.
4. The DFL model mitigates communication bottlenecks by decentralising the learning process in AV networks, distributing communication load, and eliminating dependence on a central server. This enhances model resilience in adverse conditions or malicious attacks.

Additionally, our model optimises the communication between clients using CSMA/CA protocols, minimising collisions and interference rates in DFL implementation for AV networks. DFL surpasses standard methodologies through geometric models and frequency allocation mechanisms, providing accurate wireless communication evaluations.

4.1.3 Paper Organisation

In this paper, we organise our content into several sections to ensure a systematic presentation of our research. Section 4.2 provides the foundational knowledge about DFL in dynamic networks, highlighting its importance and addressing challenges posed by Byzantine participants. Following that, Sec. 4.3 thoroughly details the architecture of our DFL network and the methodologies employed to create a resilient model in dynamic wireless networks. Moving forward, Sec. 4.4 presents our extensive simulation outcomes and provides a comprehensive analysis of our proposed DFL Byzantine-resilient model's performance across various scenarios and network dynamics. Finally, in Sec. 4.5, we summarise our key findings, draw conclusions, discuss the implications of our research within the DFL context, and outline potential avenues for future research.

4.2 Background

This section offers a concise overview of the leading defences in the field of DFL with Byzantine attacks. Recent demonstrations have revealed their susceptibility to state-of-the-art *untargeted model poisoning attacks*, as demonstrated in prior work [2]. In this type of malicious activity, the attacker aims to degrade the overall performance of the model without specifically targeting specific predictions.

Setup and Notations. Our analysis considers a total of φ workers, categorised into a set of total workers denoted as $\mathcal{G} \subseteq \{1, \ldots, \varphi\}$. Our primary objective is to minimise the following function:

$$f(\mathbf{W}) := \frac{1}{|\mathcal{G}|} \sum_{i \in \mathcal{G}} f_i(\mathbf{W}), \tag{4.1}$$

where f_i represents the loss function associated with worker i, operating under a distinct and heterogeneous data distribution ξ_i.

The stochastic gradient computation by a worker i using a minibatch ξ_i is defined as $g_i(\mathbf{W}, \xi_i) := F_i(\mathbf{W}; \xi_i)$. Here, F_i represents the function that computes the gradient, and g_i is the resulting gradient value.

It's important to note that stochastic gradients are unbiased estimators of the true gradients, meaning that:

$$\mathbb{E}_{\xi_i}[g_i(\mathbf{W}, \xi_i)] = f_i(\mathbf{W}), \tag{4.2}$$

where the expectation is taken over the distribution of minibatches ξ_i. Additionally, the variance of these gradients is bounded:

$$\mathbb{E}_{\xi_i}[\|g_i(\mathbf{W}, \xi_i) - f_i(\mathbf{W})\|^2] \leq \sigma^2. \tag{4.3}$$

To ensure data consistency among workers, we assume a certain threshold (ζ) to constrain the heterogeneity across their data as follows:

$$\mathbb{E}_{j \sim \mathcal{G}} \left\| \nabla f_j(\mathbf{W}) - \nabla f(\mathbf{W}) \right\|^2 \leq \zeta^2, \quad \forall\, \mathbf{W}. \tag{4.4}$$

When clarity permits, we refer to the gradient computed at time step t as g_t or simply g_i.

Byzantine Attack Model. In the Byzantine attack model, a subset of Byzantine workers is represented by $\mathcal{B} \subset \mathcal{G}$, while the remaining workers are classified as good. This yields $\mathcal{G} = \mathcal{B} \biguplus \mathcal{N}$, where $\mathcal{N} = \{1, 2, \ldots, N_n\}$ is the trustee clients subset. To quantify the impact of Byzantine elements, we introduce δ as the fraction of Byzantine workers, where $|\mathcal{B}| =: q \leq \delta N_n$. Byzantine workers possess the capability to deviate from the established protocol, transmitting arbitrary updates to the collaborative neighbours in the DFL network. Remarkably, they can collaborate and may even possess complete knowledge of the states of all other workers.

4.2.1 Decentralised Federated Learning (DFL)

In continuation of the introduction, this section outlines a fully decentralised architecture for FL. The proposed system is investigated in terms of the updating process of model weight parameters and their transmission across network devices.

In our decentralised system, a network of edge dynamic devices (e.g., AV clients) is envisioned to be connected through a wireless mesh infrastructure. Unlike conventional FL setups, the DFL architecture has no central aggregation server. The objective of this system is to collaboratively optimise the parameters $\mathbf{W}$ of a global model (e.g., $\hat{y} = f(\mathbf{W}, x)$, where $\hat{y}$ represents the model's predicted output and x denotes the input data). Each device (i.e., AV client) possesses its private dataset $\mathcal{D}_i$ of input data x_i, which remains inaccessible to other devices within the network. The local loss function at each device is defined as [3]:

$$L_i(\mathbf{W}) = \frac{1}{|\mathcal{D}i|} \sum_{x,y \in \mathcal{D}_i} l\Big(\hat{y} - y; \mathbf{W}, x\Big) \tag{4.5}$$

Here, $|\mathcal{D}_i|$ denotes the size of the local dataset, and $l(\hat{y} - y)$ quantifies the difference between the model's predicted output and the actual output for input x. The local loss function is assumed to be both convex and smooth [3].

During each training iteration k, the local model weights $\mathbf{W_{k_i}}$ of each device i at the iteration's start are considered. Employing its local dataset, each device conducts a specific aggregation rule on the local loss function and updates its local model weights such as *Stochastic Gradient Descent* [1], which is a common aggregation rule in FL paradigm, and can be implemented by applying the following equation:

$$\mathbf{W_{k_i+1}} = \mathbf{W_{k_i}} - \mu \nabla L_i(\mathbf{W_{k_i}}) \tag{4.6}$$

In this context, μ represents the learning rate which is chosen to facilitate the convergence of the SGD algorithm towards a minimum.

Afterwards, each device transmits its updated local model weights $\mathbf{W_{k_i+1}}$ to its neighbours within the accessible wireless network. Additionally, it receives the updated local model weights from its corresponding neighbours. A local aggregation process is then employed, involving a simple averaging mechanism for these updated local model weights. This step generates the initial updated local model weights for the subsequent iteration [4].

The above process yields the diffusion of local model weight parameters across the network during each iteration. This diffusion process involves sharing the impact of each device's local training dataset with the network through the transmission of local model weights. Research conducted in [5] demonstrates that this cooperative learning network attains convergence towards the identical global optimum with a comparable convergence rate in contrast to traditional CFL methodologies.

Therefore, this study aims to combine DFL and Byzantine-resilient mechanisms into a single intelligent system, enhancing the robustness of networks for various practical applications. The goal is to ensure the privacy and security of participants' information.

Network security significantly influences the learning process in DFL, yet researchers often overlook it. This paper extensively addresses this aspect and proposes using a Byzantine-resilient DFL network to bolster security during the learning process. The rationale behind this approach is that DFL algorithms can effectively update the edge terminals with parameters via reliable aggregation rules. This enables the creation of a global model capable of analysing end-users data accurately, all while preserving data locally and without sharing it with a central server or other devices on the network. Essentially, the data remains safeguarded within individual devices, leading to personalisation, assured Quality of Service (QoS), and improved performance for several Autonomous Vehicles (AVs) applications.

4.2.2 Byzantine-resilient Aggregation Rules

This work introduces an overview of various key aggregation methods implemented in FL and outlines their inherent strengths and potential weaknesses. For instance, **FedSGD** [1] represents a rudimentary aggregation approach, implementing a weighted mean aggregation of gradients. The weights are proportional to the quantity of data samples each client maintains. Notably, FedSGD's vulnerability lies in its susceptibility to a solitary malicious client sending amplified malevolent gradients. In contrast, both **Trimmed Mean** and **Median** methodologies [6] operate independently to aggregate parameters. Specifically, the Trimmed Mean technique omits the maximum of users who can behave maliciously (B_{max}) at both lower and higher extremes for each parameter. Conversely, the Median method extracts the median from each parameter update across all received gradients from the clients. These aggregation techniques are susceptible to what is known as the Full-Trim attack [2]. Meanwhile, the **Krum** technique [7] designates a single local model to become the subsequent global model. The selection is based on the client that exhibits the smallest Euclidean distance from its nearest neighbours, which is then chosen as the local model. The Full-Krum attack [2] is specifically designed to compromise this approach. **Bulyan** [8] employs a combination of the previous techniques. It utilises Krum [7] in an iterative fashion to choose a specific number of models and then applies the Trimmed Mean technique to the chosen ones.

Notably, the Full-Trim attack is also effective against Bulyan. **FABA** [9] functions in an iterative manner, eliminating models that are the farthest from the mean of the remaining unfiltered models. This process occurs B_{max} times before the mean of the residual gradients is returned. In an attempt to defend against Sybil clone poisoning attacks, **FoolsGold** [10] identifies clients with similar cosine similarity as malicious. Such clients have their reputation penalised, and a weighted mean of the gradients is returned, weighed by this adjusted reputation.

In addition, **FLTrust** [11] seeks to establish client trust by presuming server access to a clean, albeit small, validation dataset. The method returns a weighted mean of the gradients that are weighed by this acquired trust. However, it is worth noting that it may not be realistic to access such a clean dataset, particularly considering the non-iid nature of the local datasets at the clients.

Nevertheless, FL offers a variety of methodologies to aggregate client-provided data and navigate potential threats. Each method comes with unique advantages and its own set of potential vulnerabilities to malicious interference, particularly in a decentralised FL approach where there is no central server that can monitor the connected devices' behaviour. Thus, an understanding of various protection techniques is crucial in developing a robust and secure DFL system, particularly in scenarios where malicious attacks are a concern.

Notably, [12] introduces a novel approach in their FLAIR framework, computing reputation scores using flip-scores of local gradients. Their results in terms of accuracy and loss have shown superior performance relative to other reputation score methods. Inherent to the structure of the DFL model is the absence of a central server. Consequently, strategies like the one implemented in [12], which relied heavily on a central server for evaluating participating clients based on their behaviour during training, become unfeasible.

4.2.3 Threat Scenario: Cutting-Edge Model Poisoning Attack

In the context of DFL algorithms, the following discussion in this work delves into a critical threat model. This model considers scenarios where adversaries compromise a subset of distributed devices collaborating to optimise learning in a peer-to-peer manner in the DFL network, and thus, the adversaries target this network in order to impact the integrity of the DFL process via *data poisoning attacks* such as Sybil-based attacks [13]. This threat model's implications become particularly significant when considering the potential consequences of compromised devices and the nuances of advanced model poisoning attacks. To elucidate this further, we explore these state-of-the-art untargeted model poisoning attacks: the Full-Trim attack [2], the Full-Krum attack [2] and the Shejwalkar attack [14].

The principal focus of our study centres on advanced untargeted model poisoning attacks, specifically examining the Full-Trim attack, the Full-Krum attack and the Shejwalkar attack. These attacks fall within the category of directed deviation attacks (DDAs), which possess the capability to circumvent all recognised defence mechanisms. DDAs are characterised by their strategic manipulation of the local models on compromised decentralised devices. This manipulation is executed systematically, frequently involving the resolution of constrained optimisation problems. The ultimate objective is to induce the global model to deviate in a direction contrary to the anticipated benign global model updates.

The exploration of Byzantine resilient systems and reputation score techniques is widespread in the literature. Studies such as [15] and [16] use reputation scores as an incentive for clients to maintain system fidelity. Conversely, the works of [10] and [17] calculate reputation scores based on the cosine similarity between client gradients, while [11] employs cosine similarity in comparison with trusted gradients obtained from a clean validation dataset at the server.

In light of this, the present work advocates for the amalgamation of the reputation-based approach with the Byzantine-resilient decentralised stochastic optimisation [18] methodology. This proposition aims to construct a robust Byzantine-resilient DFL framework capable of efficiently mitigating Byzantine influences. It accomplishes this by dynamically determining optimal aggregation weights for each participating client (e.g., Autonomous Vehicles) within the DFL network during the learning process at each iteration. This strategy serves as a promising countermeasure against various untargeted model poisoning attacks, mitigating the potential adverse effects of Byzantine devices and thereby optimising the overall DFL model performance.

4.3 System Model

The key goal of this work is to explore techniques and algorithms that enhance the resilience of DFL systems against Byzantine attacks, maintain the integrity of the learning process and protect the privacy of participants' data in a decentralised and distributed setting. In our system model, We make the assumption that the attacker or Byzantine possesses control over certain participants' devices and can manipulate the local model parameters sent from these devices to the main device (i.e., master device) during the learning process. In our model, the main device is the device that has been chosen by the group based on its capability and availability to aggregate local models from neighbours at that certain iteration in order to train and update the global model of the DFL model.

In the proposed network, the attacker's knowledge of the aggregation rule used by the main device can vary, as they may or may not be aware of it.

In the context of the DFL learning process, our version of DFL involves a random number of AVs. Each AV has its local data. At each round, each AV communicates with its neighbours' users (i.e., users within its reachable area). They all should share the same model architecture and implement the same ML algorithm. Within this AV group, there is a potential of having several users considered Byzantine users $\mathcal{B}$. The assumption in our model is that up to a maximum of B_{max} users can behave maliciously in any given round, where the total number of Byzantines $\mathcal{B} \leq B_{max}$. To tackle the Byzantine issue in the DFL network, we utilise dynamic aggregation rules that consider the behaviour of neighbouring devices in each iteration. This includes both the device's behaviour in prior iterations and the similarity of model parameters in the current iteration.

4.3.1 Byzantine-Resilient DFL

Byzantine devices refer to faulty or malicious participants in the DFL network that can intentionally send incorrect or misleading updates to compromise the learning process.

The system design is based on an undirected graph G, represented as $G := (\mathcal{N} \cup \mathcal{B}, \mathcal{E})$, where $\mathcal{N}$ and $\mathcal{B}$ correspond to sets of trust and Byzantine devices, respectively. $\mathcal{E}$ refers to a subset of the Cartesian product $(\mathcal{N} \cup \mathcal{B}) \times (\mathcal{N} \cup \mathcal{B})$, which constitutes the edges or connections among vehicle devices and does not include any self-links.

Importantly, the trusted devices are unaware of both the number and identities of Byzantine devices within the network. In this context, a terminal AV or device link $(n, v) \in \mathcal{E}$ given the undirected nature of the graph, propose a bi-directional communication link between vehicle n and vehicle v. For any specific device n, we can denote the sets of its trusted and Byzantine neighbours as $N_n := \{v|(v, n) \in \mathcal{E},\ v \in N\}$ and $B_n := \{v|(v, n) \in \mathcal{E},\ v \in \mathcal{B}\}$, respectively. $\mathcal{N}$ and $\mathcal{B}$ represent the total numbers of trusted and Byzantine devices, defined as $\mathcal{N} := |\mathcal{N}|$ and $\mathcal{B} := |\mathcal{B}|$. Similarly, for any device n, $N_n = |N_n|$ and $B_n = |B_n|$ ascertain the number of its trusted and Byzantine neighbours' links, respectively.

In our proposed design, we have made an assumption that there is a number of Byzantine attackers with control over certain participants' devices, where the maximum number of Byzantine clients is capped at $B_{max} < \frac{v_n}{3}$. This assumption suggests that when trusted clients outnumber Byzantine ones, the Byzantine group will face a challenging recovery process, and then the model will be able to work efficiently and reliably. Here, v_n signifies the total number of the neighbours connected to participant n.

At the steady state, participants engage in a process where they train their local models using their respective datasets. Subsequently, they send a matrix of updated parameters to a master device during a randomly selected time slot. The master device then aggregates these local models to create a global model. In this context, the aggregated parameters are treated differently and independently. The individual user ID consistently assigns a weight that reflects the degree of trustworthiness and reliability of the device. This weight is determined by the device's reputation score within the network.

However, it is crucial to acknowledge a potential limitation of this approach. A Byzantine neighbour within the network possesses the capability to maliciously manipulate the model updates. For instance, such a neighbour could nullify the weighted average by sending 0s values for the model weights update ($\mathbf{W}$), effectively rendering the model meaningless. Conversely, they could intentionally send messages with infinitely large elements, leading to an inflated global model. Compounding the concern, the impact of a malicious user is not confined to direct interactions but can also affect honest neighbours throughout the information diffusion process [19] via intermediate users. This creates a challenge in ensuring the integrity and security of the learning process, especially in the presence of adversarial elements.

Therefore, our model is thoughtfully crafted to effectively eliminate any devices that transmit either 0s or weight values exceeding a specified threshold deemed impractical for the learning process. Such devices are not counted as active participants in the learning process to ensure the integrity and accuracy of the global model's construction. Furthermore, to manage the behaviour of the remaining devices, a sophisticated algorithm is employed, which incorporates two distinct rewards score values denoted as $\mathcal{W}(m, t)$.

These scores serve a dual purpose: the first applies penalties to B_{max} devices that most exhibit excessive or insufficient scores, while the other rewards devices based on their individual scores during each iterative phase.

At each time point t, the primary user, denoted as n, assigns the score $\mathcal{W}(n_m, t)$ cumulatively to every neighbouring user m. These scores increase by 1 if the device receives a reward and 0 if the device receives a penalty, as shown in Eq. (4.7). This reward and penalty mechanism considers the probability of the device being trustworthy, which is based on the model similarity derived from the pairwise cosine similarities between the parameter updates of the master device and each neighbouring participant.

To clarify, penalties are selectively imposed on a subset of B_{max} clients displaying the most insufficient scores, while the remaining clients are rewarded. Here, the identity of the master model aggregator AV is represented by n, and m symbolises the active connected neighbours to n at time t.

The reward and penalty mechanism is defined as follows:

$$\mathcal{W}(m,t) := \begin{cases} 0, & \text{if } \mathbb{P}(m|\mathcal{N}) < \gamma \quad (\text{ Penalised } m) \\ +1, & \text{if } \mathbb{P}(m|\mathcal{N}) \geq \gamma \quad (\text{ Rewarded } m) \end{cases} \tag{4.7}$$

$$\mathbb{P}(m|\mathcal{N}) = \frac{\mathbb{P}(\mathcal{N}|m) \cdot \mathbb{P}(m)}{\mathbb{P}(\mathcal{N})} \tag{4.8}$$

where $\mathbb{P}(m|\mathcal{N})$ denotes the probability trustworthiness of a device m based on prior beliefs about trustworthiness and observed evidence such as device behaviour, history reputation score and model similarity, and (γ) is the trust threshold that defines the minimum cumulative score required for a device to be considered trustworthy.

In this context, further refinement of the analysis is required. Assuming $\mathbb{P}(m)$ is the prior probability that device m is trustworthy, $\mathbb{P}(\mathcal{N})$ is the probability that a device is trusted, and $\mathbb{P}(\mathcal{N}|m)$ is the likelihood that a device is trusted given its behaviour, we can rewrite the inequality penalty condition accordingly:

$$\frac{\mathbb{P}(\mathcal{N}|m) \cdot \mathbb{P}(m)}{\mathbb{P}(\mathcal{N})} < \gamma \tag{4.9}$$

Given that $\mathbb{P}(\mathcal{N})$ can be expressed as a number of iterations over all devices i, and assuming independence between devices, it becomes:

$$\frac{\mathbb{P}(\mathcal{N}|m) \cdot \mathbb{P}(m)}{\sum_i \mathbb{P}(\mathcal{N}|i) \cdot \mathbb{P}(i)} < \gamma \tag{4.10}$$

In the context of the Byzantine-resilient DFL paradigm, a pivotal element emerges in the form of reputation scores which are allocated to each neighbour, denoted as $\mathcal{R}(m,t)$. These scores play a critical role in assessing the contributions of each neighbour device (m) within the network, all of which are connected to the master client n. Notably, these reputation scores are intricately linked to the individual client IDs (m) as well as the progression of time (t). This dynamic link to identity and temporal evolution gives these scores the ability to evolve, adapt, and consistently mirror the behaviour and engagement of network devices.

$$\mathcal{R}(m,0) \geq 0,$$

$$\mathcal{R}(m,t) := \alpha\mathcal{R}(m,t-1) + \frac{\mathcal{W}(m,t)}{\mathcal{W}(t)}, \quad t > 0 \tag{4.11}$$

where $\mathcal{W}(t)$ is the summation of all weighted scores for all participants at time t.

This model articulates the progression of reputation scores within the DFL network over time and ensures that the participants are engaged with respect to their quality of contributions and behaviour, thereby nurturing a dynamic environment of trust and collaboration. An important element in Eq. (4.11) is the parameter α, which takes values between 0 and 1. This parameter delicately modulates the equilibrium between the preceding iteration's reputation score, $\mathcal{R}(m, t-1)$, and the current session's weighted contribution, $\mathcal{W}(m, t)$. A higher α value leans more towards valuing the anterior reputation score, which is beneficial for reinforcing consistent, positive participation over time. In contrast, a lower α value diminishes the weight of past scores, favouring a more significant impact from recent activities on the reputation. Thus, α navigates the delicate balance between preserving reputation continuity and adaptability, with higher values anchoring the reputation in historical performance and lower values enabling smooth recalibrations to acknowledge recent actions.

In our framework, we apply the softmax function to normalise the reputation score $\mathcal{R}(m, t)$, which reflects the quality of contributions and behaviour of participants in the DFL network. With each iteration, summing these normalised scores from all interconnected devices (i.e., participant nodes) results in a total of one. This approach effectively represents the changing dynamics of reputation scores. Specifically, the normalised reputation score for a node m at a given time t is represented as $\hat{\mathcal{R}}(m, t)$, which is achieved through softmax normalisation, as illustrated below:

$$\hat{\mathcal{R}}(m, t) = \frac{e^{\mathcal{R}(m,t)}}{\sum_{\substack{\text{all connected}\\ \text{neighbours to } n}} e^{\mathcal{R}(t)}}, \quad t > 0. \tag{4.12}$$

This formulation clarifies the mechanism through which participant reputation is refined and adapted over successive iterations, essentially influencing the network's overall trust dynamics.

Furthermore, we examine the general format of the robust aggregation rule denoted as $\mathcal{A}_n$, designed for an honest worker n within the set $\mathcal{N}$. This rule is expressed as follows:

$$\begin{aligned} &\mathcal{A}_n(\mathbf{W}_n, \{\tilde{\mathbf{W}}_{m,n}\}_{m \in N_n \cup B_n}) \qquad (4.13)\\ &:= (1 - r_n)\mathcal{A}(\mathbf{W}_n, \{\tilde{\mathbf{W}}_{m,n}\}_{m \in N_n \cup B_n}) + r_n \mathbf{W}_n \end{aligned}$$

In this context, $\mathcal{A}$ represents a *base aggregation method* (e.g., FedSGD and Krum, see

Sec. 4.2.2 shared across all honest workers, and $r_n \in [0, 1)$ denotes an individual-specific constant. The base aggregator, working with the available model weights, generates a D-dimensional vector $\mathbb{R}^D$. However, it is possible for the base aggregator's output to lack model information $\tilde{\mathbf{W}}_{m,n}$ from neighbours m that an honest worker n should rely on. To address this, we explore a convex fusion of the base aggregator's output and $\mathbf{W}_n$, controlled by r_n as shown in Eq. (4.13). Thus, a lower value of r_n signifies a greater reliance on the base aggregator.

Here, we examine several well-known base aggregation methods as baselines in the context of Byzantine-resilient optimisation. These serve as illustrative instances. For ease of notation, we define the input of aggregator $\mathcal{A}$ as $\{\mathbf{W}_1, \ldots, \mathbf{W}_{\mathcal{S}}\}$, and the total number of devices excluding the master device denotes as $\mathcal{S} = N_n + q$, where $|\mathcal{N}| = N_n$, $|\mathcal{B}| =: q \le \delta N_n$ and δ represents the fraction of Byzantine workers as discussed in Sec. 4.2.

The *Coordinate-wise median (CooMed)* aggregation algorithm calculates the median value for each individual coordinate $d = 1, \ldots, D$ as follows [6]:

$$\text{CooMed}(\mathbf{W}_1, \ldots, \mathbf{W}_S)_d = \text{Median}([\mathbf{W}_1]_d, \ldots, [\mathbf{W}_S]_d) \tag{4.14}$$

where $[\mathbf{W}]_d$ refers to the d-th element of vector $\mathbf{W}$.

The *Geometric median (GeoMed)* identifies a point that minimises the sum of distances to all input vectors [20], [21]:

$$\text{GeoMed}(\mathbf{W}_1, \ldots, \mathbf{W}_{\mathcal{S}}) = \arg\min_{\mathbf{W}_n} \sum_{m=1}^{\mathcal{S}} \|\mathbf{W}_n - \mathbf{W}_m\|_2. \tag{4.15}$$

Another common aggregator method is called *Robust Federated Averaging* (RFA) [22]. This method calculates the geometric median between the latest global model (V) with the current neighbours' model updates as follows:

$$\text{RFA}(\mathbf{W}_1, \ldots, \mathbf{W}_m) := \arg\min_{V} \sum_{i=1}^{m} \|V - \mathbf{W}_i\|_2. \tag{4.16}$$

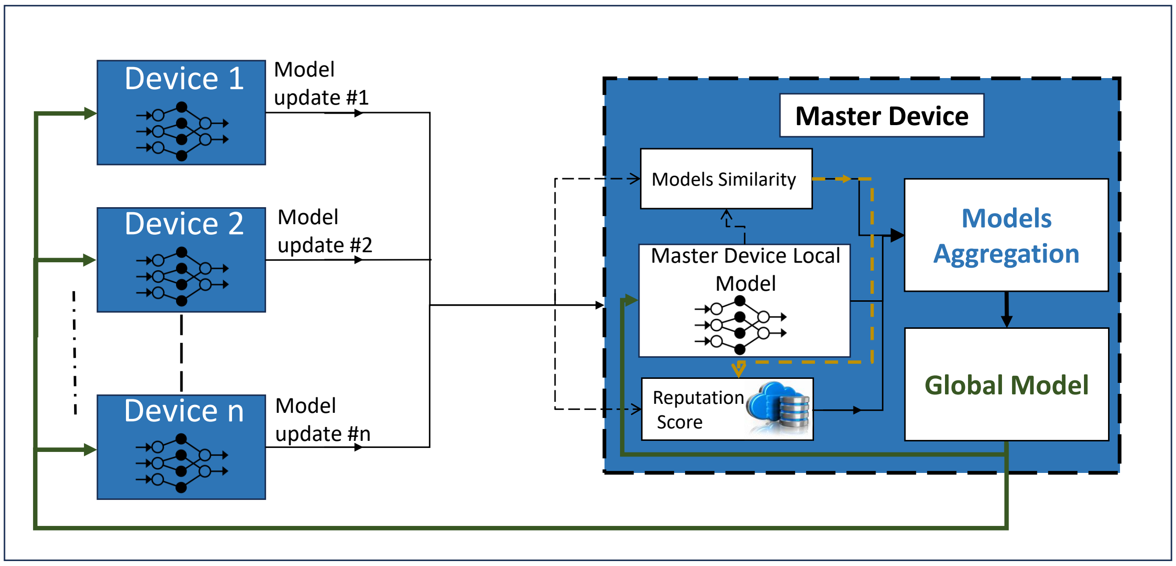


Figure 4.3: Illustration of the proposed aggregation and model update layout in DFL.

The RFA method, as detailed above, employs the geometric median to integrate model updates from various devices in an FL network. By minimising the sum of Euclidean distances ($\|V - \mathbf{W}_i\|_2$) between the global model (V) and each local model update ($\mathbf{W}_i$), RFA effectively finds a central point that represents the most robust consensus among the distributed models. While the geometric median lacks a closed-form solution, the RFA work in [22] approximates it using multiple iterations of the smoothed Weiszfeld algorithm [23]. This method employs an iterative approach, iteratively refining the geometric median estimate. Importantly, each iteration of this technique demands a relatively efficient $\mathcal{O}(n)$ computational effort.

Furthermore, the *Krum* method, pioneered by Blanchard et al. [7], proves to be a valuable tool when it is possible to estimate the number of Byzantine workers accurately, represented as q. This method identifies the input vector with the shortest distance to the $\mathcal{S} - q - 2$ nearest vectors in the dataset, offering a robust solution in such scenarios.

Thus, for $n \neq m$, let $m \to n$ denotes that the aggregated models $\mathbf{W}_m$ from subset m are transmitted to and received by the master device n. These models are selected from a set of $\mathcal{S} - q - 2$ closest vectors to $\mathbf{W}_n$. Then, the base aggregation of models in the Kurm method is described as follows:

$$\mathrm{Krum}(\mathbf{W}_1, \ldots, \mathbf{W}_S) = \underset{\mathbf{W}_{n,m}}{\arg\min} \sum_{m \in \mathcal{U}} \|\mathbf{W}_n - \mathbf{W}_m\|_2^2 \tag{4.17}$$
$$|\mathcal{U}| = \mathcal{S} - q - 2.$$

Nevertheless, the methodologies of Krum and aggregation techniques entail notable computational intricacies. These intricacies either demand the computational capabilities of a potent central server or impose substantial processing burdens on the master device within the context of DFL. Such resource-intensive operations have the potential to undermine the efficacy of the resultant model in real-world applications.

In our design, we introduce a novel aggregation rule that aligns with the principles of DFL. This aggregation rule incorporates a predefined threshold (γ) to optimise the weighted score at each iteration, as illustrated in Eq. (4.7). This predefined threshold is meticulously calibrated in advance, and the master devices have the flexibility to fine-tune it in response to real-time network conditions as the operation progresses. Moreover, our aggregation rule enables the model to regulate the influence of connected neighbour devices, taking into account their reputation scores and the correlation between the participant model and the local model of the master device n, as shown in Fig. 4.3. By factoring in reputation scores, we aim to reduce the adverse impact of potentially malicious and Byzantine devices within the DFL network at each iteration, denoted as t. As a result, our robust aggregation rule $\mathcal{A}_{n,t}(.)$ applied by the master device n at time t is fine-tuned to address suspicious devices by utilising the weighted score in Eq. (4.7) and the reputation score in Eq. (4.11). The rule can be summarised as follows:

$$\mathcal{A}_{n,t}(\mathbf{W}_n, \{\tilde{\mathbf{W}}_{m,n}\}_{m \in N_n \cup B_n}) \tag{4.18}$$
$$:= (1 - r_n)\mathcal{A}(\mathbf{W}_n, \{\hat{\mathcal{R}}(m,t) \times \tilde{\mathbf{W}}_{m,n}\}) + r_n \mathbf{W}_n$$

Our approach utilises the integration of the proposed reputation score $\mathcal{R}(m,t)$ with various aggregation methods in the DFL model, including the Federated Averaging (FedAvg)

[1] algorithm introduced by McMahan et al. [1]. While FedAvg calculates the anticipated mean value of aggregated models for global updates, our novel approach introduces a unique twist. We incorporate individual normalised reputation scores $\hat{\mathcal{R}}(m,t)$, resulting in a weighted aggregation scheme as depicted in Eq. (4.19). This novel approach enhances the aggregation process by considering the reputation of individual models based on the client history records and received model correlation and similarity to the master devices, as shown in Fig. 4.3. Additionally, we introduce a parameter r_n that determines the degree of influence exerted by the updated global model $\mathcal{A}_{n,t}$ for the master client n at each time sequence t. This parameter plays a pivotal role in shaping the influence dynamics within our aggregation process, as elaborated in Eq. (4.18).

$$\mathcal{A}(.) = \sum_{m=1}^{v_n} \hat{\mathcal{R}}(m,t) \times \tilde{\mathbf{W}}_{m,n} \tag{4.19}$$

As a consequence, the global model within our Byzantine-resilient DFL framework undergoes iterative updates. These updates are guided by the aggregator rule, as defined in Eq. (4.18). This rule is systematically applied to the selected master device, denoted as n, during each iteration, resulting in the update of the global model. Thus, the aggregator method mechanism for updating the global model can be expressed as follows:

$$\mathcal{A}_{n,t}(.) := (1 - r_n) \sum_{m=1}^{v_n} \hat{\mathcal{R}}(m,t) \times \tilde{\mathbf{W}}_{m,n} + r_n \mathbf{W}_n \tag{4.20}$$

While the reputation score and similarity approach may initially lead to a slower rate of updating the global model updates, it ensures consistent progress and provides significant enhancements in model performance, including accelerated convergence throughout the entire training process. In addition, the influence of neighbours' updates can be controlled by adjusting the value of r_n, which is a constant value $0 < r_n < 1$. A smaller r_n emphasises the base aggregator's contribution, while a larger r_n gives more weight to the local update of the master device model.

Furthermore, the aggregation mechanism that is used to update the global model can be implemented using a variety of existing aggregation methods as a baseline to evaluate the improvement that our approach can achieve. The choice to employ various alternative aggregation methods, including the well-known FedAvg [1] and the state-of-the-art Krum method, serves the purpose of evaluating the performance enhancement of our models in comparison to alternative approaches. This evaluation primarily focuses on aspects of model efficiency, taking into consideration both complexity and power consumption, particularly in the context of our design for the classification benchmark datasets.

Consequently, our proposed system does not rely on a central server and employs an approach that simplifies a robust DFl model and reduces its dependence on external factors. As a result, the DFL model can collaboratively update parameters in real-time applications, even when in the presence of Byzantine devices, and does not require a connection to a central server to monitor and manage the training process. This setup fosters a robust and flexible peer-to-peer approach.

4.3.2 DFL Learning Criteria

In the DFL approach, all participating devices must possess an identical machine-learning model, for instance, a Convolutional Neural Network (CNN) that contains a shared weight parameters matrix, denoted as $\mathbf{W}$, which is considered as an optimisation variable for the model. The primary goal of the model design is to reduce the cross-entropy loss function's discrepancy between the expected and predicted outputs, expressed as:

$$\arg\min_{\mathbf{W}} F(\mathbf{W}) \triangleq \frac{1}{v_n} \sum_{(m)} f_m(\mathbf{W}) \tag{4.21}$$

In this context, the notation $F(\mathbf{W})$ represents the global loss function, and $f_m(\mathbf{W})$ corresponds to the local loss for the neighbour devices, where m represents the set of active AVs neighbours or devices that successfully transmit to the master device n during each iteration of the training process. Furthermore, v_n denotes the length of the active devices set m, defined as:

$$m = 1, 2, \ldots.., v_n.$$

The local loss function for device m is derived from the cross-entropy loss of the local training dataset $\mathfrak{D}_{(m)}$, expressed as:

$$f_m(\mathbf{W}) = \frac{1}{D} \sum_{i=o}^{D} l(h_{\mathbf{W}}(X_m^i), y_m^i), \tag{4.22}$$

Here, i signifies a subset of the total count of local training datasets, D corresponds to the length of the participant datasets $\mathfrak{D}(m)$ divided by the batch size as follows $(\frac{|\mathfrak{D}(m)|}{Bm})$, and $l(h_{\mathbf{W}}(X_m^{(i)}), y_m^{(i)})$ represents the cost function for the weights matrix $\mathbf{W}$ evaluated on a hypothesis $h_{\mathbf{W}}(X_m^i)$ with data samples X_m^i. For example, the hypothesis for the simple linear regression is defined as $h_{\mathbf{W}}(X) = \mathbf{W}_0 + \mathbf{W}_1 X$.

During the t^{th} step of the DFL process, each device m possesses a local parameters weights matrix $\mathbf{W}^{(t)}$ that is revised to optimise model accuracy and find a fitting solution to the problem, repeating the models' exchange process until the model achieving the convergence state ε as follows:

$$\frac{1}{v_n}(\sum_{m=1}^{v_n} \nabla f_m(\mathbf{W}_m^t)) < \varepsilon \tag{4.23}$$
$$\text{s.t.} \quad v_n >> 1$$

Then, a chosen master device at time t aggregates the local models from the connected devices, which will be defined based on a spatial and communication model at a time t (i.e., iteration), to update the global model based on the proposed aggregation techniques in Sec. 4.3.1 and then shared the updated global model with others in the network for a new iteration in DFL learning process. These proposed techniques (i.e., aggregation

Algorithm 4.1 Byzantine Resilient DFL Network

```
1: $\varphi = \mathcal{N} \cup \mathcal{B}$        –»Total number of participants
2: All participants have initial weights with $\mathbf{W}(0)$
3: for iteration (time) $t = 1, 2, \ldots..R$ do
4:     for each Master device $(n)$ do in parallel
5:         Aggregate models and define $\mathcal{N}$ and $\mathcal{B}$ devices
6:         for all $n$'s active nodes $m = 1, 2, \ldots..v_n$ do
7:             $\mathcal{W}(m,t) := \begin{cases} \text{Penalty } m, & \text{if } \mathbb{P}(m|\mathcal{N}) < \gamma \\ \text{Reward } m, & \text{if } \mathbb{P}(m|\mathcal{N}) \geq \gamma \end{cases}$
8:             for $i = 1, \ldots$ samples of dataset $\mathfrak{D}_{(m)}$, where
9:                 $D = \frac{|\mathfrak{D}_{(m)}|}{B_m}$ , where $B_m$ is Batch size do
10:                $\nabla f_m(\mathbf{W}^t_m) = \frac{1}{D} \sum_{i=1}^{D} \nabla l(h_{(\mathbf{W}^t_m)}(X^i_m, y^i_m))$
11:                $\mathbf{W}^t_{(m)} \longleftarrow \mathbf{W}^t_{(m)} - \eta^t_m \nabla f_m(\mathbf{W}^t_m; B_m)$
12:            end for
13:        end for
14:        Master device $n$ at iteration $t$ receives $\mathbf{W}^t_m$ and
15:        $\nabla f_m(\mathbf{W}^t_m)$ from $v_n$ active neighbours clients:
16:        while $\frac{1}{v_n}(\sum_{m=1}^{v_n} \nabla f_m(\mathbf{W}^t_m)) > \varepsilon$ & $v_n > 1$:
17:            (i.e., Convergence state $\varepsilon$ is not achieved yet)
18:            $\mathcal{R}(m,t) := \alpha\mathcal{R}(m,t-1) + \mathcal{W}(m,t), \quad t > 0$
19:            $\hat{\mathbf{W}}^t_n \longleftarrow (\mathcal{A}_{n,t}(\mathbf{W}_n, \{\tilde{\mathbf{W}}_{m,n}\})$
20:                $:= (1 - r_n)\mathcal{A}(\mathbf{W}_n, \{\hat{\mathcal{R}}(m,t)\tilde{\mathbf{W}}_{m,n}\}) + r_n\mathbf{W}_n$
21:            $\mathcal{A}(.) = \sum_{m=1}^{v_n} \hat{\mathcal{R}}(m,t) \times \tilde{\mathbf{W}}_{m,n}$
22:            $\mathbf{W}^{(t+1)}_n \longleftarrow \hat{\mathbf{W}}^t_n$ do
23:        end while
24:        return end the training (Gradient Convergence)
25:    end for
26:    Master device broadcast $\hat{\mathbf{W}}^t_m$, and then go to Step 3
27: end for
```

methods using DFL reputation score and model similarity) give our approach the privilege of offering a significant advantage by removing potentially untrustworthy models and devices from consideration. It achieves this by placing the highest trust in master devices and others whose model parameters have demonstrated strong correlations over time. This correlation-based approach builds a reputation score database, ensuring the reliability of trusted participants.

The summary of the implemented algorithm is given in Algorithm 4.1.

4.3.3 Spatial and Communication Models

We consider that wireless nodes (i.e., autonomous vehicles) are randomly scattered in a 2-D plane. Their location can be modelled in different models such as cluster processes or queuing theory-based models that can be used for more accurate representation in these scenarios, but for simplicity, one of the most representative distribution models that can be used for vehicle networks, or more specifically, Vehicular Ad Hoc Networks (VANETs), is the Poisson Point Process (PPP) model [24].

The Poisson Point Process model is widely used to represent the spatial distribution of AVs and other related participants in a network due to its simplicity and analytical tractability [24]. It assumes that AVs are randomly and independently distributed across the network, following a Poisson process.

Each point in the PPP represents an AV or another related participant, and the process determines the probability of finding a participant in any given area. This will be particularly useful for analysing and simulating wireless communication within our V2X network, including some other factors such as signal coverage, interference, and connectivity.

The density of the PPP is represented by λ, with an assumption that each device collaboratively shares model parameters with its neighbours.

In the proposed model, it is assumed that there exist multiple subchannels available for communication between AVs and other related participants during the learning process.

These subchannels allow for multiple AVs to transmit and receive data simultaneously, thus improving the efficiency and speed of communication. The Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) protocol is utilised in this multi-user communication scenario. Under this protocol, an AV can share parameters with its neighbouring AVs only when no other AV within the same subchannel is transmitting. If a transmission is already in progress, the AV must wait and then attempt to transmit again after a random backoff time. This multi-user, multi-channel communication setup allows for effective and efficient learning processes, where parameters can be shared, and global model updates can be propagated rapidly across predefined channels.

In this study, we focus on a typical receiver that is linked to the desired transmitter. For the small-scale path-loss model, we adopt a Rayleigh fading channel combined with a single slope large-scale path-loss. Consequently, the signal received from the desired transmitter (i.e., the receiver connected to typical neighbours) is represented as $(P_k h_{ko} d_{ko}^{-\alpha})$ [25]. Here, P_k denotes the power of the signal transmitted by the desired transmitter device k, and $\alpha \geq 2$ represents the path-loss exponent. Additionally, d_{k0} and h_{ko} indicate the distance and fading coefficient, respectively, for the channel between device k and the target neighbours.

To assess the real-world interference scenario within the network and explicitly quantify the number of AV clients that successfully transmit, we can employ a physical interference model based on the widely recognised signal-to-interference-plus-noise ratio (SINR). We express the probability of successful transmission (P_s) for transmitter k during the DFL training process as follows:

$$P_s = \mathbb{P}(SINR \geq T) = \mathbb{P}\left(\frac{P_k h_{k0} d_{k0}^{-\alpha}}{\sum_{i \in \varphi} I_i + N_0} \geq T\right) \tag{4.24}$$

where P_s represents the probability ($\mathbb{P}$) of successful transmission, which is determined by the probability of SINR exceeding a predefined threshold T. Here, I_i and N_0 represent the interference from other devices i and the noise in the channel, respectively. The noise is assumed to be significantly lower in comparison to interference. Consequently, the equation can be expressed as follows:

$$P_s = \mathbb{P}(SINR \geq T) = \mathbb{P}\left(\frac{P_i h_{i0} d_{i0}^{-\alpha} a_i}{\sum_{i \in \varphi} I_i} \geq T\right) \tag{4.25}$$

The interference originating from device i in the network is $I_i = P_i h_{i0} d_{i0}^{-\alpha} a_i$ with $i = 1, 2, ..., v_n$. The binary variable a_i indicates whether the device is transmitting (i.e., $a_i = 1$) or ready to receive ($a_i = 0$). The summation of interference ($\sum_{i \in \varphi} I_i$) encompasses interference from surrounded active devices (v_n) within the desired coverage area. The set φ, defined as the union of $\mathcal{N}$ and $\mathcal{B}$, represents the total number of AV clients across the entire target area.

Furthermore, various communication techniques can be employed to enhance communication reliability among devices. One of our proposed approaches is the utilisation of MultiPath TCP (MPTCP) [26], where devices can share model updates using diverse sub-flows for reliable transmission. It considers factors like network conditions, available bandwidth, and congestion levels to select the most suitable path for each subflow. This is particularly beneficial in cases where a single network path might be experiencing congestion or unreliability. MPTCP allows for improved fault tolerance and load balancing by distributing model updates across multiple network paths. This technique contributes to the robustness of the communication infrastructure within the network, enhancing the reliability of model sharing and collaborative learning among connected devices.

The DFL approach is mainly tailored for dynamic wireless networks lacking a centralised server, which traditional FL relies on to handle the aggregation of model updates across the network [27]. In the DFL scenario, devices communicate peer-to-peer to construct an optimised global model. Consequently, the design of the DFL network must prioritise maximising communication reliability and minimising the risk of unauthorised devices launching attacks or poisoning the model. This is an essential objective that ensures the dependability and effectiveness of the DFL algorithm.

4.3.4 Performance Metrics

In this work, we introduce a framework tailored for optimising a classification problem within a V2X network. Consequently, our assessment of the DFL model's effectiveness is based on two widely recognised performance metrics frequently utilised in ML models:

(a) **Categorical Cross-Entropy Loss**: This loss function is a main model metric that is mainly applied to adjust the model weights during the training process and is commonly used for classification problems. Typically, the cross-entropy loss function, also known as logistic or log loss, is prevalent in such cases. It evaluates the probability of each prediction against the binary actual output, assigning higher loss to larger deviations and minimal loss to smaller ones, with a total loss of 0 indicating a perfect model. The function is given by:

$$Loss = -\sum_{j=1}^{m}\sum_{i=1}^{n} y_{(i,j)} \log(p_{(i,j)}), \tag{4.26}$$

where $y_{(i,j)}$ and $p_{(i,j)}$ are the actual output and predicted probability for class i and sample j, respectively.

(b) **Accuracy**: It is a common metric for examining the performance of various ML models. Accuracy is determined in the following manner:

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}}. \tag{4.27}$$

It quantifies the proportion of correctly classified instances, encompassing true positives and true negatives, among all instances in the dataset. In DFL, accuracy plays a significant role in improving system performance by identifying and addressing potential issues with the model's predictions. The accuracy value is computed by comparing the model's predictions to the actual labels in the dataset and dividing the number of correct predictions by the total number of instances in the dataset.

(c) **Model Complexity**: Estimating the complexity of a DFL model involves evaluating various aspects, such as the model architecture, the learning process, and the communication patterns between the distributed devices. The model's size plays a crucial role in model architecture complexity, as larger models with more parameters tend to be more complex. Furthermore, communication complexity is another crucial factor to consider, where an increase in the number of distributed devices participating in the learning process and the number of communication rounds needed for training can escalate complexity. As such, it is essential to consider a trade-off balancing between the complexity of the DFL model and the target accuracy during the model's configuration.

4.4 Simulation Results

In this section, we assess the effectiveness and performance of our introduced Byzantine-resilient DFL model through comprehensive simulation experiments. The simulation of malicious users involves executing various attack types, which are detailed in Sec. 4.2.3. These malicious users are programmed to randomly employ different attack strategies to disrupt the model's performance.

In the proposed network, a cellular frequency reuse technique (i.e., N frequencies for the whole network) is used over a large-scale network of uniform cells in order to increase the capacity without increasing its allocated bandwidth during the learning process. The simulated network consists of random participants of both AV clients (i.e., trusted clients) and other untrusted devices (i.e., militia and Byzantine devices) that have a Poisson distribution within the target areas. Initially, the potential of having data collisions during the learning process in our large-scale network is mitigated through the utilisation of CSMA/CA as the fundamental communication protocol. This protocol adeptly manages multi-user communications across different network intensity levels denoted by λ in our network simulation.

Since the participants' devices possess diverse computational capacities, the complexity of the deployed DFL model also varied accordingly. Due to restricted processing capabilities, a significant increase in device numbers consistently raised the complexity of the DFL

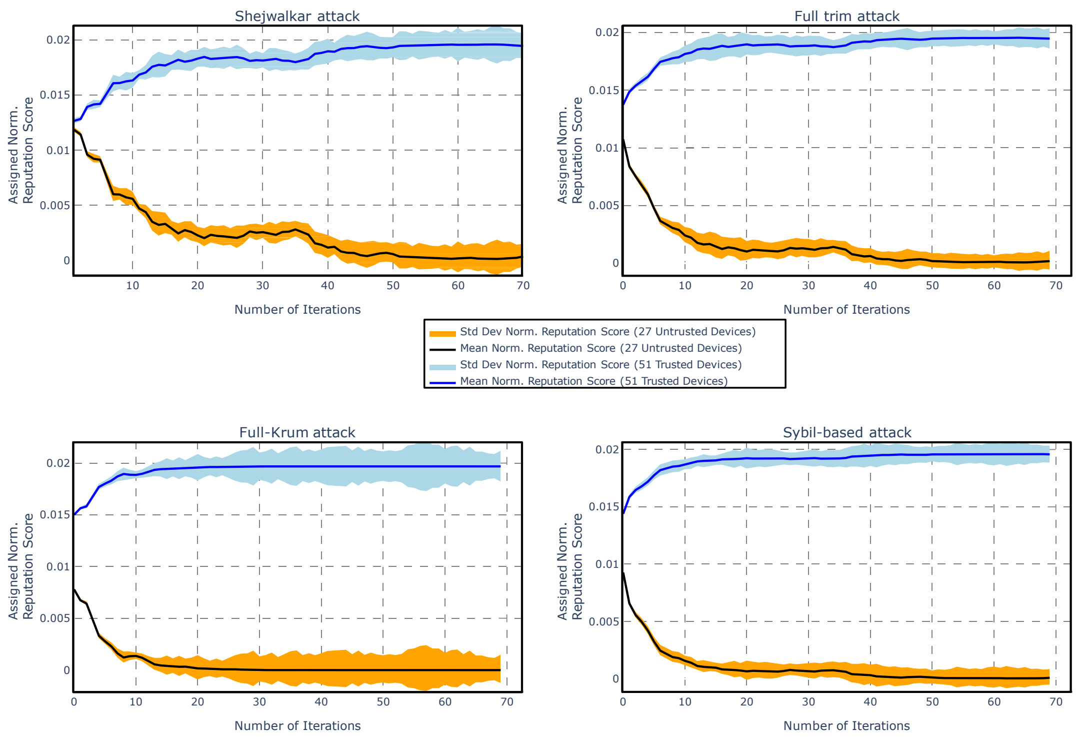


Figure 4.4: The average and standard deviation of the assigned normalised reputation score for untrusted and trusted devices during the DFL learning process under various attacks.

model, potentially exceeding the capabilities of some devices.

In general, as the number of trusted devices increased, the amount of trained data also increased, resulting in a steady enhancement of recognition accuracy. Thus, there is a necessity to strike a balance between the number of collaborating devices and the achievable accuracy in the model's design. Moreover, the CNN model, a widely acclaimed tool for image classification [28], has been selected as the local machine learning (ML) model of our classification problem using the MNIST dataset for benchmarking the model (i.e., the summary details of MNIST is described in Table 4.1.

Clients leverage this robust tool to train their models with their local datasets, benefiting from our optimisation efforts in terms of the hyperparameters setting, such as model size and layer count. This optimisation is geared towards achieving an acceptable level of accuracy while simultaneously maintaining low computational requirements and complexity at the edge, thereby streamlining the overall complexity of the proposed DFL model.

To simplify matters, considering that each device shares an identical local model archi-

Table 4.1: The benchmark dataset details

Dataset	**# Train**	**# Test**	**Feature**	**Class**	**Model**
MNIST	60,000	10,000	784	10	CNN

Table 4.2: Comparison of the complexity and performance improvement in our model over the Krum model at various network intensities.

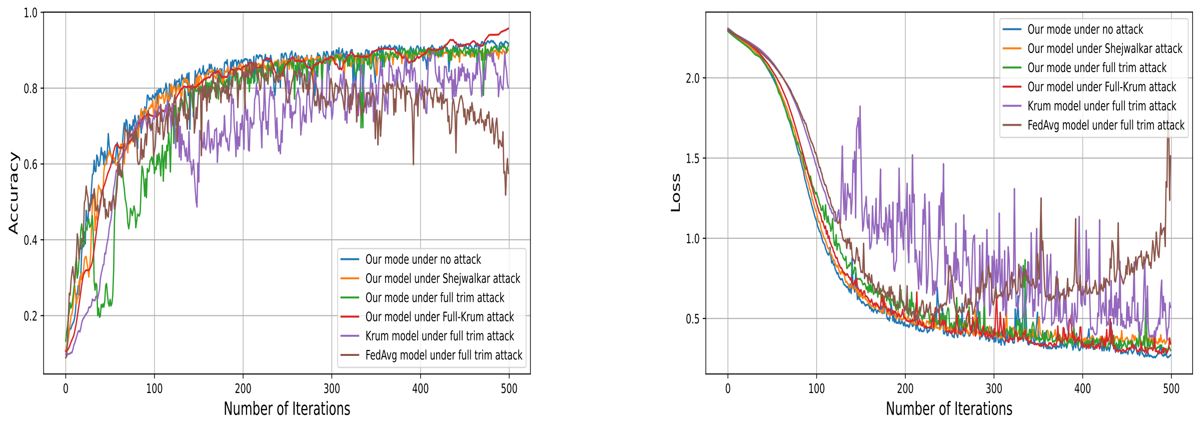
Network Intensity	Krum Model				Our Model			
	Func. Calls	Time (s)	Accuracy	Loss	Func. Calls	Time (s)	Accuracy	Loss
0.01	19549602	26.5	86.5	0.098	19548731	29.838	94.3	0.034
0.05	21037066	27.94	87.2	0.068	20992056	28.997	92.2	0.051
0.1	22986612	33.094	88.1	0.0.52	22796069	31.568	93.9	0.063
0.15	25036039	39.941	86.3	0.095	24600214	33.356	91.5	0.057
0.2	27185901	45.218	90.2	0.073	26404326	34.298	92.7	0.052
0.25	29435537	53.261	89.8	0.090	28208340	38.05	93.1	0.049
0.3	31785116	69.608	88.9	0.080	30012534	39.525	92.5	0.063
0.35	34234619	85.177	87.5	0.108	31816619	43.361	91.8	0.081
0.4	36784225	90.731	85.7	0.209	33620909	43.805	93.5	0.105

tecture, complexity can be assessed in terms of the learning process (i.e., the aggregation and optimisation model processes computation) function in the number of communication epochs necessary to reach the predefined convergence threshold ε that is described in Eq. (4.23).

In this work, we introduced a novel score reputation model that contributed significantly to the convergence of our DFL model. This reputation model was instrumental when employing a weighted averaging approach during the model aggregation process. By assigning reputation scores to individual clients based on their historical behaviour and performance, we not only improved convergence but also added an additional layer of security.

The reputation score model excels in gauging the trustworthiness of clients within the DFL network by diligently monitoring and archiving their historical activities. This model played a significant role in enhancing the security and reliability of the DFL system. By maintaining a record of client behaviour, we were able to detect anomalies and potentially malicious activities.

Unlike certain existing defences that perform well only under specific configurations, such as specific combinations of attacks and models, our DFL model has demonstrated a robust defence mechanism that was adaptable across various aggregation methods against



(a) (b)

Figure 4.5: (a) Accuracy and (b) loss comparison between our DFL model and various baseline models under different attack scenarios in the proposed network.

various state-of-the-are Byzantine attacks. Importantly, our DFL model has maintained its resilience even when up to 35% of the clients within the network were potentially malicious, as our model eliminates the negative effect of the suspension and untrusted devices using our proposed score reputation and weighted mechanisms to minimise the untrusted devices' contribution, which are described in Sec. 4.3.1 and the assigned weights for the network devices during the DFL training are shown in Fig. 4.4.

In this study, we conducted an extensive evaluation of model complexity under diverse network intensity scenarios. Table 4.2 serves as a crucial tool for visually illustrating the performance of our model when compared to the Krum model. This state-of-the-art FL model uses the Krum aggregator method for processing the models' aggregations. This performance evaluation encompasses various levels of network intensity and user counts. The metric function calls quantify the frequency with which system functions are invoked and executed throughout the operation of our model. This metric offers valuable insights into the computational and complexity demands placed on the model. Importantly, our model showcases significant reductions in complexity and execution time, as vividly demonstrated in the table, especially under high-intensity network conditions. These findings underscore the distinct advantages and suitability of our proposed model when dealing with substantial network intensity scenarios.

These enhancements in our model are primarily attributed to the well-tuned model facilitated by the reputation score derived from the model similarity and weighted score mechanism in Eq. 4.7, which optimises the model performance by enabling a flexible and reliable system. The implementation of the accumulated trust level, determined by assessing model similarity between the master device and participant neighbours, along with the novel DFL reputation scoring system, had several positive impacts on our model performance and can be summarised as follows:

1. Enhanced Security: The score reputation weights allowed us to identify and flag potentially malicious clients, contributing to the overall security of the DFL network. By addressing security concerns proactively, we reduced the vulnerability of the system to adversarial attacks.
2. Improved Model Convergence: Our reputation scoring model played a significant role in enhancing the convergence of our DFL model by amplifying the value of incorporating trusted model updates during the learning process. This enhancement is instrumental in ensuring the effective integration of model updates, ultimately resulting in improved model performance in key metrics such as accuracy and loss, as shown in Fig. 4.5.
3. Comparative Performance and Robustness: In our evaluation of the DFL model, we conducted a performance analysis that involved comparing our model with various baseline models, including networks with Byzantine devices, while employing different aggregation methods such as Krum and FedAvg. As illustrated in Fig. 4.5, our model consistently outperformed others, exhibiting steady improvement and delivering high performance under these challenging conditions.

These results signify a substantial advancement in the field of DFL security and performance. Our aggregation method and reputation scoring system not only contribute to the convergence of the DFL model but also provide robust defences against adversarial attacks, making our DFL system a reliable and secure choice for collaborative learning in distributed networks. Lastly, the optimisation in our DFL framework enhances model performance by facilitating a system that is flexible, computationally efficient, and highly reliable.

4.5 Conclusions

The adoption of a reputation-scoring system greatly enhances the convergence of our DFL model. By giving precedence to reliable model updates, our methodology facilitates efficient integration, leading to enhanced model effectiveness in accuracy and loss metrics. A thorough comparative analysis against a state-of-the-art baseline model was performed to evaluate our DFL model. These assessments underscored the model's exceptional robustness. Significantly, the model's efficacy was tested in various conditions, including networks compromised by Byzantine devices, and compared using different aggregation techniques like Krum and FedAvg. Across the board, the model consistently surpassed its peers, showing continuous progress and high-performance levels.

To sum up, this work introduces a robust and secure DFL model designed to excel in challenging environments. By prioritising trusted updates and showcasing superior performance, this approach contributes to the advancement of secure DFL systems. These findings present promising opportunities for the deployment of DFL in real-world applications, as no central server and infrastructure are required, even in the presence of adversarial actors.

In the future, further investigation into the scalability of our model is warranted, especially in scenarios with a large number of participating devices. Optimising resource allocation and communication protocols in such contexts can be a valuable research direction.

References

[1] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, “Communication-efficient learning of deep networks from decentralized data,” in *Artificial intelligence and statistics*, PMLR, 2017, pp. 1273–1282.

[2] M. Fang, X. Cao, J. Jia, and N. Gong, “Local model poisoning attacks to {byzantine-robust} federated learning,” in *29th USENIX security symposium (USENIX Security 20)*, 2020, pp. 1605–1622.

[3] Z. Lin, X. Li, V. K. N. Lau, Y. Gong, and K. Huang, *Deploying federated learning in large-scale cellular networks: Spatial convergence analysis*, 2021. arXiv: `2103.06056 [cs.IT]`.

[4] S. Savazzi, M. Nicoli, and V. Rampa, “Federated learning with cooperating devices: A consensus approach for massive iot networks,” *IEEE Internet of Things Journal*, vol. 7, no. 5, pp. 4641–4654, May 2020, ISSN: 2372-2541. DOI: `10.1109/jiot.2020.2964162`.

[5] A. H. Sayed, S.-Y. Tu, J. Chen, X. Zhao, and Z. J. Towfic, “Diffusion strategies for adaptation and learning over networks: An examination of distributed strategies and network behavior,” *IEEE Signal Processing Magazine*, vol. 30, no. 3, pp. 155–171, 2013.

[6] D. Yin, Y. Chen, R. Kannan, and P. Bartlett, “Byzantine-robust distributed learning: Towards optimal statistical rates,” in *International Conference on Machine Learning*, PMLR, 2018, pp. 5650–5659.

[7] P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," *Advances in neural information processing systems*, vol. 30, 2017.

[8] R. Guerraoui, S. Rouault, *et al.*, "The hidden vulnerability of distributed learning in byzantium," in *International Conference on Machine Learning*, PMLR, 2018, pp. 3521–3530.

[9] Q. Xia, Z. Tao, Z. Hao, and Q. Li, "Faba: An algorithm for fast aggregation against byzantine attacks in distributed neural networks," in *IJCAI*, 2019.

[10] C. Fung, C. J. Yoon, and I. Beschastnikh, "The limitations of federated learning in sybil settings," in *23rd International Symposium on Research in Attacks, Intrusions and Defenses (RAID 2020)*, 2020, pp. 301–316.

[11] X. Cao, M. Fang, J. Liu, and N. Z. Gong, "Fltrust: Byzantine-robust federated learning via trust bootstrapping," *arXiv preprint arXiv:2012.13995*, 2020.

[12] A. Sharma, W. Chen, J. Zhao, Q. Qiu, S. Bagchi, and S. Chaterji, "Flair: Defense against model poisoning attack in federated learning," *ACM ASIA Conference on Computer and Communications Security*, 2023.

[13] C. Fung, C. J. Yoon, and I. Beschastnikh, "Mitigating sybils in federated learning poisoning," *arXiv preprint arXiv:1808.04866*, 2018.

[14] V. Shejwalkar and A. Houmansadr, "Manipulating the byzantine: Optimizing model poisoning attacks and defenses for federated learning," in *NDSS*, 2021.

[15] J. Kang, Z. Xiong, D. Niyato, S. Xie, and J. Zhang, "Incentive mechanism for reliable federated learning: A joint optimization approach to combining reputation and contract theory," *IEEE Internet of Things Journal*, vol. 6, no. 6, pp. 10 700–10 714, 2019.

[16] Y. Zhang, M. J. Wainwright, and J. C. Duchi, "Communication-efficient algorithms for statistical optimization," *Advances in neural information processing systems*, vol. 25, 2012.

[17] S. Awan, B. Luo, and F. Li, “Contra: Defending against poisoning attacks in federated learning,” in *Computer Security–ESORICS 2021: 26th European Symposium on Research in Computer Security, Darmstadt, Germany, October 4–8, 2021, Proceedings, Part I 26*, Springer, 2021, pp. 455–475.

[18] Z. Wu, T. Chen, and Q. Ling, “Byzantine-resilient decentralized stochastic optimization with robust aggregation rules,” *arXiv preprint arXiv:2206.04568*, 2022.

[19] J. Peng, W. Li, and Q. Ling, “Byzantine-robust decentralized stochastic optimization over static and time-varying networks,” *Signal Processing*, vol. 183, p. 108 020, 2021.

[20] Y. Chen, L. Su, and J. Xu, “Distributed statistical machine learning in adversarial settings: Byzantine gradient descent,” *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, vol. 1, no. 2, pp. 1–25, 2017.

[21] C. Xie, O. Koyejo, and I. Gupta, “Generalized byzantine-tolerant sgd,” *arXiv preprint arXiv:1802.10116*, 2018.

[22] K. Pillutla, S. M. Kakade, and Z. Harchaoui, “Robust aggregation for federated learning,” *IEEE Transactions on Signal Processing*, vol. 70, pp. 1142–1154, 2022.

[23] E. Weiszfeld and F. Plastria, “On the point for which the sum of the distances to n given points is minimum,” *Annals of Operations Research*, vol. 167, pp. 7–41, 2009.

[24] M. Khabazian and M. M. Ali, “A performance modeling of vehicular ad hoc networks (vanets),” in *2007 IEEE Wireless Communications and Networking Conference*, IEEE, 2007, pp. 4177–4182.

[25] T. Jing, X. Chen, Y. Huo, and X. Cheng, “Achievable transmission capacity of cognitive mesh networks with different media access control,” in *2012 Proceedings IEEE INFOCOM*, IEEE, 2012, pp. 1764–1772.

[26] M. Bagnulo, *Threat Analysis for TCP Extensions for Multipath Operation with Multiple Addresses*, RFC 6181, Mar. 2011. DOI: `10.17487/RFC6181`. [Online]. Available: `https://www.rfc-editor.org/info/rfc6181`.

[27] A. Salama, A. Stergioulis, A. M. Hayajneh, S. A. R. Zaidi, D. McLernon, and I. Robertson, "Decentralized federated learning over slotted aloha wireless mesh networking," *IEEE Access*, vol. 11, pp. 18 326–18 342, 2023.

[28] M. Xin and Y. Wang, "Research on image classification model based on deep convolution neural network," *EURASIP Journal on Image and Video Processing*, vol. 2019, no. 1, pp. 1–11, 2019.

Chapter 5

Enhancing Reliability and Efficiency in Collaborative Learning through Decentralised Federated Learning using MPTCP

Chapter source: A. Salama, S. A. Zaidi, D. McLernon, and M. M. Qazzaz, "Enhancing Reliability and Efficiency in Collaborative Learning through Decentralised Federated Learning using MPTCP," IEEE Open Journal of the Communications Society (Submitted in April, and still under review).

Abstract

Establishing reliable connectivity in dynamic Decentralised Federated Learning (DFL) networks presents significant challenges due to frequent topological changes and unpredictable participant movements. Continuous communication, essential for the learning process within the DFL framework, is greatly enhanced by adopting multipath connectivity facilitated by Multipath TCP (MPTCP). This approach addresses the inherent limitations of serving access technologies that often lead to frequent disconnections as participants move across coverage areas. To ensure robust communication in such dynamic environments, we propose an advanced subflows management algorithm that is seamlessly integrated with our topology optimisation algorithm, termed the Dynamic Network Topology Algorithm (DNTA). DNTA employs a dynamic and sophisticated network selection strategy, utilising Reinforcement Learning (RL) algorithms to discern the most effective network configuration.

In contrast to existing solutions, our proposed approach introduces a DFL model that dynamically leverages the 4 available subflows of MPTCP, ensuring adaptive and efficient network usage to enhance the learning process. Through extensive system simulations, we evaluate our decentralised framework against state-of-the-art federated learning (FL) benchmark models, leveraging the CIFAR-10 dataset to benchmark image classification performance. Our findings indicate that our decentralised setup achieves up to 96.5% accuracy and a loss of only 0.038%, outperforming the benchmark models. It also significantly enhances latency (less than 15 ms) and data throughput on wireless channels. By optimally utilising parallel subflows of the proposed multipath scheme, our approach significantly mitigates outages and connection instability by up to 35% improvement in

comparison to a single-path approach, underscoring its potential for real-world applications in the evolving landscape of distributed learning.

5.1 Introduction

In the dynamic landscape of collaborative Machine Learning (ML) and smart collaboration across mobile and wearable devices, efficient and reliable communication is essential. The transport layer plays a pivotal role in ensuring these communication demands are met. However, as these devices become increasingly prevalent, traditional transport protocols struggle to meet the demands of enhanced connectivity and reliability required for dynamic networks [1]. In response, Multipath TCP (MPTCP) emerges as a transformative solution [2], optimising the use of multiple network paths can improve data flow and resilience for advanced collaborative learning applications [2], such as DFL [3] for dynamic environments. MPTCP has been officially recognised by the Internet Engineering Task Force (IETF) RFC 6182 as a standard tool for bandwidth utilisation [4]. This approach not only enhances data transmission efficiency but also supports the sustainable integration of emerging technologies by ensuring robust and adaptable network connections [5] [6].

Furthermore, as we progress towards increasingly autonomous and sophisticated collaborative applications nowadays [7], the necessity for advanced resource management strategies becomes paramount, especially with the increasing dependence on cellular and ad-hoc networks for these applications. This shift underscores the need to move beyond traditional centralised architectures to address vulnerabilities such as single points of failure and concentrated cyber threats.

Therefore, DFL emerges as a flexible scheme to overcome some limitations in the conventional FL framework, such as communication bottleneck [3], where devices collaborate in a peer-to-peer manner in order to optimise a global model without the need for a central server. This enables resilient and intelligent collaborative systems as the data processing is distributed across multiple devices (see Sec. 5.3.4 for more details on the DFL learning process). This distribution is vital for further mitigating the risks associated with centralised systems and enhancing data privacy and security. Despite its potential, the reliability of DFL networks in mobile environments is yet to be fully realised, often hampered by issues such as random packet loss, network congestion, and user mobility [8] [9] [10]. In this regard, MPTCP can be adopted to optimise the DFL network by providing various available subflows and aggregating bandwidth across these network subflows [2]. As a result, utilising MPTCP can significantly enhance the management of multiple data streams on mobile devices, reduce costs and improve network efficiency, which leads to enhancing the DFL user experience in terms of learning performance and accuracy.

In response, this research focuses on enhancing the reliability and connectivity of the DFL network through the strategic use of MPTCP. By integrating MPTCP, this study aims to ensure robust and efficient data handling in mobile networks, particularly enhancing load balancing, throughput and resilience against package drops and link failures during the learning process in the traditional single-path protocols. This approach not only supports the operational demands of modern mobile and edge devices but also addresses critical concerns of data privacy, low latency, and resource efficiency in decentralised learning environments.

The remainder of this work is structured as follows. Section 5.2 provides an overview of MPTCP technology and its advantages for enhancing the DFL framework. In Section 5.3, we detail the system architecture, focusing on a theoretical exploration of client communication within the network and the evaluation metrics employed for DFL enhanced by MPTCP. Section 5.4 describes the simulation setup and parameters chosen to assess the performance of DFL when integrated with MPTCP, contrasting these findings against benchmark conventional FL models based on traditional TCP connections under a similar network design. The paper concludes with a summary of our findings and the advancements achieved through this research in Sec. 5.5.

5.2 Background and Motivation

In a DFL scenario involving multiple clients communicating in a peer-to-peer manner, the communication of model parameters of the DFL participants typically relies on the Transmission Control Protocol (TCP) to ensure reliable transmission. The DFL process can be categorised into three phases:

1. Stable Phase: Model parameters of the DFL user are transmitted to the target neighbours without issues during this phase. There are no delays or packet losses in the transmission process.

2. Congestion phase: In the congestion phase, delays in model update transmission may occur due to excessive data traffic. Potential packet loss can occur as a result of buffer overflow. To address packet loss, TCP initiates the retransmission of lost model updates. However, even with prompt retransmission, clients may experience an additional delay of at least one Round-Trip Time (RTT).

3. Disconnected phase: The disconnected phase encompasses scenarios such as connection loss, retransmission timeout (RTO), and the appearance of three duplicated ACKs. During this phase, the transmission of model updates faces significant delays or may fail entirely.

Issues in a DFL network using a single-path TCP scenario arise when continuous transmission becomes impractical, requiring a client to either wait for an extended period or promptly establish another connection. Disconnections of DFL users are more prevalent in dynamic wireless communication networks, multi-hop networks, and other wireless access networks. Several researchers have conducted studies to enhance and optimise DFL for various scenarios [11]. Nonetheless, these studies have not explored the use of MPTCP for managing the utilisation of multiple network communication interfaces simultaneously. This gap highlights the need for a multipath approach that can effectively manage and control subflows, thereby improving reliability and connectivity without compromising the users' privacy.

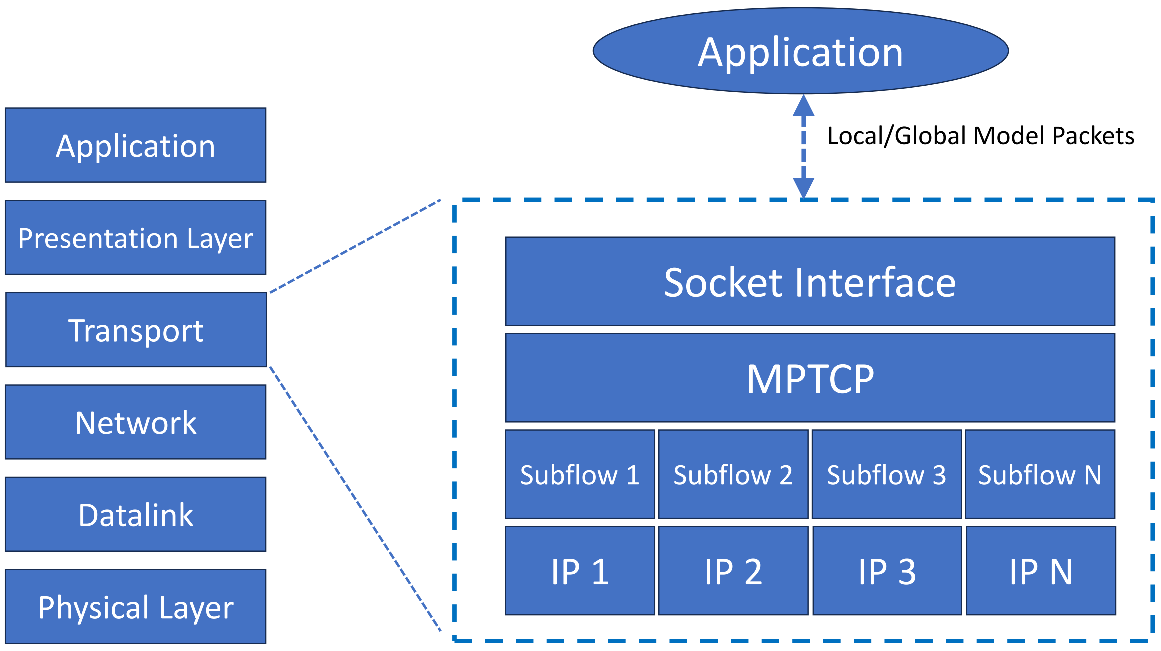


Figure 5.1: An overview of MPTCP stack for DFL

Therefore, MPTCP employs multiple connections to mitigate congestion and disconnected phases [12] in the DFL networks. This strategy aims to minimise the impact of these challenges, especially when operators significantly impact the learning process.

In addition, various multipath protocols of the transport layer are available for implementation in DFL networks, including multipath quick UDP internet connection protocol (MPQUIC) [13], concurrent multipath transmission based on stream control transmission protocol (CMTSCTP) [14], and multipath transmission control protocol (MPTCP) [15]. In particular, MPTCP, an extension of the widely adopted TCP, has achieved the IETF standard status [15] and has been seamlessly integrated by major smartphone manufacturers such as Apple, Samsung, and Huawei [16], [17], [18]. This integration makes it feasible for real-world applications, allowing for seamless coordination during the DFL model updates, resilience in the face of network failures, and the effective aggregation of bandwidth from multiple network interfaces.

MPTCP approach is proposed to operate for the establishment of multiple connections simultaneously for the DFL users by utilising multiple available interfaces, which the number of these interfaces can be denoted as N, as shown in Fig. 5.1. Based on network

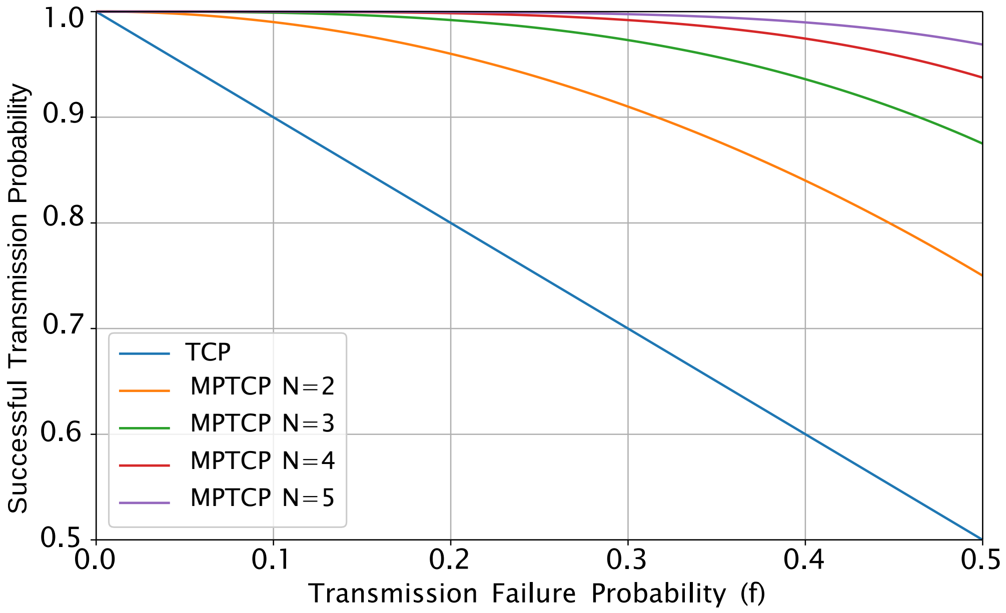


Figure 5.2: Successful Transmission Probability with Multipath Transmission

policy and design, these N interfaces generate $N-1$ replicated control packets, each of which is simultaneously transmitted through its respective interface [19]. Upon successful transmission of any control packet, MPTCP proceeds to transmit the next one in sequence. Assuming an average transmission failure probability of P_{f_i} for each process, the overall successful transmission probability P_{success} of the system can be computed as:

$$P_{\text{success}} = 1 - P_{f_1} \times P_{f_2} \times P_{f_3} \times \ldots \times P_{f_N} = 1 - \prod_{i=1}^{N} P_{f_i}. \tag{5.1}$$

This equation is commonly observed in parallel systems. In the case of MPTCP, where multiple interfaces are utilised simultaneously and assuming equal transmission failure probabilities P_f for each interface, Eq. (5.1) simplifies to:

$$P_{\text{success}} \approx 1 - {P_f}^N. \tag{5.2}$$

The analysis presented in Eq. (5.2) is depicted in Fig. 5.2. For example, if the probability of transmission failure for a single connection is 8%, the theoretical analysis indicates a successful control packet transmission probability of 0.920 with a single TCP connection in the DFL network. When employing MPTCP with two subflows, the probability increases to 0.9872, and with three subflows, it further rises to 0.99744. These encouraging findings demonstrate that even with just two connections, a significant reduction in packet losses can be achieved in standard DFL network communication. Thus, it can be inferred that as the possible number of interfaces (N) that the network can adopt increases, the successful transmission probability of the system (P_{success}) also increases, and this underscores the motivation behind our proposed model DFL over MPTCP.

Therefore, this study aims to enhance the DFL framework by integrating MPTCP mechanisms into a unified intelligent system in several ways:

1. **Improved Reliability:** MPTCP allows data to be transmitted over multiple paths simultaneously. In the context of DFL, this can enhance reliability by mitigating the impact of packet losses or failures on a single path. Using multiple paths, the system becomes more robust against network disruptions, ensuring that model updates can be successfully transmitted even under challenging network conditions [20].

2. **Reduced Latency:** The use of MPTCP results in lower latency because multiple network paths can be used at the same time [21]. For DFL, it is important to have a low latency. With low latency, convergence is faster and more efficient learning can be achieved since edge devices may communicate through various wireless signal strengths and potential communication delay scenarios.

3. **Bandwidth Aggregation:** The aggregation of bandwidth from several subflows can be realised by MPTCP, which leads to more wider available bandwidth, and therefore the model updates to be delivered in a DFL setup would find enough capacity [20]. This improvement is more beneficial, especially when dealing with large models and situations where quick transmission is necessary for successful cooperation at edge devices.

4. **Adaptability to Network Changes:** DFL systems operate in highly dynamic

environments where network conditions can change rapidly. In such adaptive scenarios, the ability to dynamically select paths for data transmission from various sources is crucial. MPTCP plays a significant role in ensuring resilience in these situations. This is particularly important given that the movement of edge devices can result in a completely new topology of signal strengths or changes in network topologies. These changes reflect the evolving realities of our surroundings [5].

Consequently, the main motivation for developing our DFL framework is to enhance privacy and security for all clients. By leveraging the capabilities of MPTCP, our framework enables efficient collaboration across clients, achieving superior network and learning performance. Effective communication within the network is crucial for attaining robust collaborative learning outcomes, such as high accuracy and minimal latency. These advancements are particularly vital for real-world applications requiring rapid and reliable data processing, such as Autonomous Vehicles (AV), where decision-making speed and data integrity are paramount.

However, it is crucial to acknowledge that the benefits of MPTCP are contingent upon compatibility with the existing physical bandwidth capacities of the network infrastructure. This ensures that the aggregated bandwidth effectively utilised does not surpass the physical limitations of network components. Furthermore, our framework presumes that both devices and network infrastructures are equipped to execute multiple algorithms concurrently. Consequently, while optimising our approach, we may not fully address the complexities and synchronisation challenges associated with multipath aggregation. This simplification is intended to focus on the primary benefits of MPTCP, recognising that practical deployments may require additional considerations for managing these technical intricacies.

5.3 System model

In this study, our aim is to train personalised models on edge nodes (i.e., clients) utilising heterogeneous data from the participants' clients while considering critical communication constraints.

To achieve our goal within an edge computing environment that includes a total of K clients, represented by the set $\varphi = \{1, \ldots, K\}$, we focus on minimising the empirical loss function $F(.)$ of DFL model. This set includes both edge-connected clients (k) and a designated primary client. The primary client is the client that is assigned to play a critical role by aggregating local updates directly from neighbouring clients to update the global model during each specific iteration of DFL. Our objective is to learn an optimal weight vector $\mathbf{W}_k$ for each client by minimising the loss function f, which is defined as follows:

$$\arg \min_{f(\mathbf{W}_k)} \frac{1}{|\mathcal{D}_k|} \sum_{i=1}^{|\mathcal{D}_k|} F(\mathbf{W}_k; (x_i, y_i)) \tag{5.3}$$

where x_i represents the prior client data, y_i is the target value, and $F(\mathbf{W}_k; (x_i, y_i))$ signifies the loss relative to the model $\mathbf{W}_k$. $\mathcal{D}_k = \{(x_i, y_i)\}_{i=1}^{|\mathcal{D}_k|}$ denotes the training set of edge client k following a distinct distribution.

Given that each client has its optimisation objective, we simplify the problem by defining the objective function as minimising the sum of all clients' loss functions. This can be represented as:

$$\arg \min_{W} \sum_{i=1}^{K} f(\mathbf{W}_i), \tag{5.4}$$

where $\mathbf{W} = \{\mathbf{W}_1, \mathbf{W}_2, ..., \mathbf{W}_K\}$ denotes a set of weight vectors containing all clients' models. In addition to minimising the objective function, we aim to increase communication reliability and reduce the total communication rounds (CR) and latency in the decentralised learning framework.

To address these limitations, our system model proposes a DFL approach leveraging MPTCP to optimise model update exchange among clients as illustrated in Fig. 5.3. The design encompasses three distinct scenarios aimed at maximising throughput, bandwidth efficiency, and adaptability to varying network conditions. The scenarios involve simultaneous exchange through available subflows, prioritising a master subflow based on

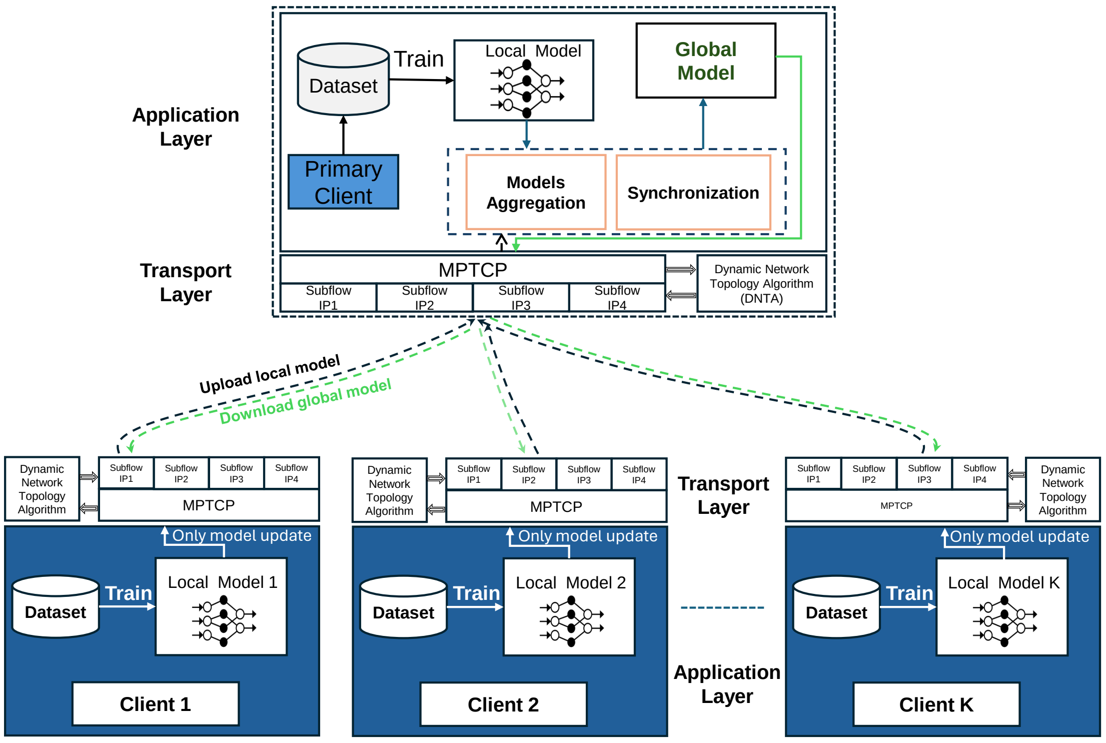


Figure 5.3: Proposed architecture of the DFL framework leveraging MPTCP

connection quality, and assigning subflows for sharing local and global model updates, as explained below:

5.3.1 Network scenarios for DFL through MPTCP

In DFL networks over MPTCP, clients have multiple avenues for peer-to-peer communication. Thus, this work introduces innovative topologies designated to satisfy the DFL requirement, such as sharing the local and global model updates efficiently, as shown in Fig. 5.3. The topologies are proposed to enhance the efficiency of subflow use within our DFL network operating over the proposed MPTCP architecture. The proposed topologies are adaptable to various distinct scenarios, denoted as s, offering robust solutions for diverse network conditions as follows:

1. **Simultaneous Subflows Utilisation Scenario** :

   In this scenario, clients engage in concurrent model update exchanges with their neighbours through multiple available subflows. The objective is to increase throughput and optimise bandwidth utilisation by leveraging the parallel transmission capabilities of MPTCP. This approach promotes efficient communication among neighbouring clients, fostering collaborative learning while minimising the impact of network delays.

2. **Subflow Allocation for Model Sharing Scenario**:

   In the second scenario, clients strategically allocate subflows for different purposes. Particular subflows are dedicated to sharing the local model's updates with other clients, while others are designated for global model updates. This fine-grained allocation optimises the utilisation of network resources, facilitating efficient collaboration and knowledge exchange within the DFL framework.

3. **Master Subflow Priority Scenario**:

   The third scenario introduces a master subflow that serves as the primary communication channel for model updates unless the connection quality through this master subflow deteriorates. In the event of poor master subflow conditions, clients dynamically switch to utilising available subflows, ensuring a seamless transition to maintain communication reliability. This adaptive approach aims to balance the benefits of a consistent communication channel with the flexibility to address suboptimal network conditions.

Through these scenarios, we explore a variety of communication strategies within the DFL network over MPTCP, providing a preliminary glimpse into how network topologies can be optimised to enhance collaborative learning in distributed environments.

Given this context, a thorough understanding of network topology is crucial to fully grasp the dynamics of these interactions. Our analysis delves into the effectiveness of MPTCP within the DFL framework, highlighting how the network's structural design influences the efficiency of collaborative learning processes.

Figure 5.3 illustrates our proposed model architecture. The network features a primary client responsible for aggregating updates from neighbours to update the global model during each epoch. Simultaneously, other clients collaborate to enhance overall model performance. Each client processes its local data, trains the model locally, and uploads the updated parameters to the primary client. This primary client then integrates these contributions into the global model and redistributes it (download) to all clients for the subsequent training iteration. These exchanges leverage MPTCP, with the communication topology dynamically optimised using the network topology algorithm outlined in Algorithm 5.2.

Building on this foundation, our research specifically aims to identify and address potential network challenges in typical DFL setups. We evaluate whether adopting different MPTCP topologies can mitigate these challenges, thereby improving the network's resilience and performance.

5.3.2 Optimise Transmission Reliability in DFL Networks over MPTCP

The transport layer primarily employs two protocols: UDP (User Datagram Protocol) and TCP (Transmission Control Protocol). UDP is connectionless, offering no guarantees on packet delivery or order and lacks rate adaptation based on available bandwidth. In contrast, TCP is connection-oriented, ensuring packets are delivered successfully and in order to their destination. Additionally, TCP includes congestion control mechanisms to limit the volume of data in the network by adjusting the congestion window (CWND) [22].

In this work, we focused on the transport layer for our proposed DFL using MPTCP; we examined how MPTCP enhances the success rate of transmitting updates for DFL models. Employing system reliability theory [23], we investigated the time needed to transmit model updates and multiple interfaces' impact. Various failures may occur during the transmission of model update packets, including unintended packet losses or delays in their arrival. As shown in Fig. 5.3, the proposed architecture contains several clients; one of these clients is assigned as a primary client in this round, and this primary client is responsible for performing the aggregating process of the local models' parameters from others to update the global model. The local model parameters result from training the local ML algorithm using the local data. In our architecture, each client can use multiple subflows to share the model parameters. Furthermore, we optimise the way that we can utilise the available subflows by implementing our network topology optimisation algorithm called Dynamic Network Topology Algorithm (further details will be introduced in Sec. 5.3.3 and in Algorithm 5.2).

In this work, we examine a diverse wireless setting that integrates a number of access network subflows (N) among communicating terminals. Wireless access networks serve as communication routes in MPTCP, encompassing both wired and wireless domains. Each communication subflow $p \in N$ functions as an autonomous transport link, possessing the following attributes.

Property 1 (Round trip time, RTT_p): The time taken for a packet to reach its destination, along with the delay in receiving an acknowledgement of that packet. Thus, RTT_p comprises the packet transmission time and the path propagation delay.

Property 2 (Packet loss rate): The average percentage of data packets that fail to reach the destination while traversing the communication paths. These packet losses may stem from network congestion, wireless channel fading, external interference, etc.

Property 3 (Available bandwidth, BW_p): The maximum transmission rate offered by the end-to-end communication path for the data flow. In HTTP/TCP data streaming systems, the available bandwidth can be estimated based on the observed TCP throughput for each subflow.

A significant attribute of MPTCP, in general, involves its congestion control mechanism, which is essential to maintain network stability during periods of increased load. Congestion control schemes play pivotal roles in achieving optimal bandwidth utilisation [24]. Congestion Avoidance Algorithms are introduced within the framework of optimising DFL through MPTCP, representing a valuable strategy for enhancing communication efficiency in decentralised systems. The study harnesses MPTCP's capability to regulate congestion windows using various algorithms. Similar to individual TCP flows, each MPTCP subflow maintains its congestion window and retransmission strategy throughout data transfer, commencing with a slow-start phase and transitioning to the congestion avoidance phase per Round-Trip Time (RTT) [25]. In the framework of DFL utilising MPTCP, various model-based congestion control algorithms like CUBIC, BBR, and OLIA stand as viable options. Each offers distinct mechanisms for handling congestion and optimising throughput across multiple network paths.

CUBIC [26], a predominant Linux algorithm, utilises packet losses as a congestion indicator, transforming window growth into a cubic function for enhanced adaptability. Decoupling window growth from RTT ensures consistent bandwidth distribution, making CUBIC flexible and stable in scenarios with non-negligible network bandwidth and latency.

BBR (Bottleneck Bandwidth and Round-trip propagation time) [27], introduced by Google, excels in networks with elevated latency and packet loss ratios. BBR periodically evaluates the available bandwidth and RTT, relying on an estimate of the bottleneck bandwidth to calculate the congestion window.

Based on empirical findings from experiments [28], BBR's performance varies depending on the combination of RTT, bandwidths, and buffer size values. This enables the identification of network conditions where BBR exhibits higher throughput compared to TCP congestion control algorithms like CUBIC. In some scenarios, such as a highway driving scenario [29], CUBIC and BBR generally demonstrate similar throughput, but BBR displays significantly reduced self-inflicted delays compared to CUBIC. These observations align with reports in [30], affirming that, in TCP simulations on highways, BBR outperforms in scenarios involving packet loss on communication links with low Signal to Interference plus Noise Ratio (SINR).

OLIA (the Opportunistic Linked-Increases Algorithm) developed by Khalili [31], addresses MPTCP's non-Pareto-optimal behaviour, especially its aggressiveness toward

```
Algorithm 5.1 DFL using MPTCP Rewards Algorithm
Evaluation on DFL Network Scenarios ($s$):
Rewards are evaluated based on path performance over any of these distinct scenarios ($s$) of DFL network using MPTCP, including:
    1. Simultaneous Subflow Exchange Scenario
    2. Subflow Allocation for Model Sharing Scenario
    3. Master Subflow Priority Scenario
Parameters:
$x_i^s \in [0, 1]$: current reward of action $i$ in the DFL Network scenarios $s$
$\overline{x}_i^s \in [0, 1]$: average reward of action $i$ in the DFL Network scenarios $s$
$n_i^s$: the number of times the action $i$ is chosen in the DFL Network scenarios $s$
$t^s$: the overall number of trials in DFL Network scenarios $s$

Function $Initialization()$:
  1: Take each action once and set the parameters.
end
Function based on UCB1 $MainLoop()$:
  1: Take action $i$ that maximises $\overline{x}_i^s + \sqrt{\frac{2\log(t^s)}{n_i^s}}$
  2: Get reward $x_i^s$.
  3: Update the parameters.
end
```

TCP users. Crafted as a Pareto-optimal algorithm, OLIA is responsive and nonflappy, ensuring optimal traffic distribution across different paths. In the context of DFL optimisation, OLIA contributes to enhancing the efficiency of MPTCP Congestion Avoidance Algorithms. Congestion avoidance control is a decentralised algorithm that adjusts parameters such as congestion window, round trip time, and loss in a feedback loop. For simplicity, we omit the time variable in the expressions.

In this work, the proposal DFL over MPTCP design incorporates a reward monitor Algorithm 5.1 and reinforcement learning (RL) control mechanisms in order to optimise the network performance over the selected subflow during the learning process, and thus further technique details are described in Sec. 5.3.3. Therefore, we improve the network performance by integrating this reward monitor Algorithm 5.1 and the RL-based MPTCP design, which functions to continuously monitor the dynamics of the network and traffic and regularly work on finding the optimum topology for the DFL network. Moreover, calculate the effective congestion window $\hat{\omega}_i$ and schedule for all paths, periodically enforcing them to ensure the target delay while accommodating transient network variations. The inner optimisation then adopts the enforced $\hat{\omega}_i$ (i.e., $\omega_i \leftarrow \hat{\omega}_i$) and fine-tunes it using Eq. (5.7) until the next reinforcement from the outer optimisation (see Sec. 5.3.3 for more details). This yields a reinforced total rate from client k over all paths, denoted as ($\hat{\theta}_k$) and then utilised for enforcing the new schedule ($h_{\hat{k}}$) and the round trip time τ_i for the path i, and these can be calculated as follows:

$$\hat{\theta}_k = \sum_i \hat{\omega}_{i,k}/\tau_i, \tag{5.5}$$

$$h_{\hat{k}} = \hat{\theta}_{i,k}/\hat{\theta}_k. \tag{5.6}$$

This revised formulation for the scheduling strategy is crucial in determining the necessary time-frame to assess network performance, thereby enabling our model to make informed decisions about whether to initiate a handover to a different network topology or to continue with the existing one

Furthermore, this work involves a model-based MPTCP algorithm (e.g., OLIA), serving as an inner control optimisation stage. A further discussion on this is available in Sec. 5.3.3. Upon detecting packet loss indication or observing round trip time on path i, it triggers a multiplicative decrease, halving the congestion window ω_i. In contrast, upon receiving an acknowledgement from path i, it employs a continuous increase in ω_i calculated as $\omega_i' \leftarrow \omega_i' + \min\{1/\omega_i', \alpha(\omega_i')\}$, where:

$$\alpha(\omega_i') = \frac{\max_j\{\omega_j'/\tau_j^2\}}{\left(\sum_j \omega_j'/\tau_j\right)^2}. \tag{5.7}$$

This approach ensures that the CNWD adapts to the network conditions, balancing between efficient throughput and network stability.

5.3.3 Topological Analysis in DFL

The learning process in the proposed DFL framework over MPTCP can manifest in one of three distinct scenarios, as described in Sec. 5.3.1.

In the first scenario, simultaneous utilisation of all subflows, clients simultaneously exchange model updates with their neighbours via multiple available subflows. The main aim in this scenario is to optimise bandwidth utilisation and increase throughput by using MPTCP's parallel transmission capabilities. By allowing clients to communicate simultaneously through different subflows, this approach promotes efficient communication between clients within our network, improving DFL learning while minimising the impact of network latency.

The total number of clients participating in the DFL process is denoted as K. In this context, each client communicates with its neighbours through N available subflows. Therefore, the overall throughput T achieved in the network can be represented as:

$$T = K \times N \times \text{Throughput per subflow} \tag{5.8}$$

This illustrates how the overall throughput increases linearly with the number of clients and the number of available subflows, demonstrating the potential improvement in network efficiency by leveraging MPTCP's parallel transmission capabilities. In the context of the DFL framework, this scenario involves clients engaging in concurrent model update exchanges through multiple available subflows.

In the second scenario, clients have opted to strategically allocate subflows for different purposes, aiming to optimise the dissemination of model updates and foster collaboration within the DFL framework. Each client dynamically allocates subflows based on its role in the learning process and the specific communication requirements. In this context, particular subflows are dedicated to transmitting and receiving local model updates within neighbouring clients. These dedicated subflows ensure an efficient and timely exchange of information, allowing clients to update their models based on the latest insights from nearby peers. In contrast, other subflows are assigned for global model updates, which involve broadcasting the latest global model update across the network. In a stable network, sharing model updates on specific subflows can optimise network resources and streamline communication for higher efficiency.

Lastly, the Master Subflow Priority Scenario involves assigning communication resources based on a predefined priority list. In this scenario, clients select a certain communication channel, known as the master subflow, for sharing model updates (both local model and also global model updates in case the client is the primary client in this iteration). The master subflow acts as the main path unless its connection strength weakens and becomes below the accepted threshold, prompting a transition to backup subflows, which is the second option in the priority list. The reason behind this approach is to maintain less complexity and a consistent communication channel whenever possible, optimising bandwidth utilisation and minimising network instability. Moreover, the system can adjust to network situations smoothly switching to alternative subflows when necessary. This flexibility guarantees communication and mitigates the impact of potential changes in network conditions on the learning process.

By balancing the benefits of a stable communication channel with the flexibility to adapt to changing network dynamics, the master subflow priority scenario enhances the efficiency and robustness of DFL over MPTCP. This approach optimises resource utilisation and minimises complexity while prioritising communication reliability, thereby facilitating collaborative learning in dynamic network environments.

To improve how we use subflows in the learning process within the DFL network, we suggest a method called the Dynamic Network Topology Algorithm (DNTA) for our DFL framework. This strategy breaks down system optimisation into two stages. The layout of these stages and our proposed DFL framework is shown in Fig. 5.4, and can be described as follows:

1. **Inner Stage (Subflows use optimisation)**: The main goal in this stage is to improve the selection of the subflows for communication between clients in DFL

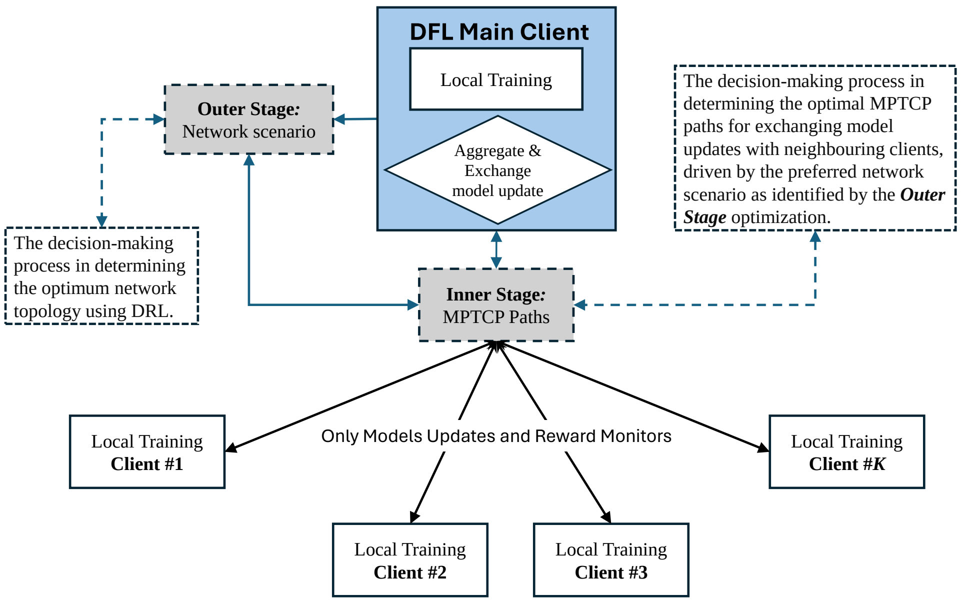

DFL Main Client
Local Training
Aggregate & Exchange model update
Outer Stage: Network scenario
The decision-making process in determining the optimum network topology using DRL.
The decision-making process in determining the optimal MPTCP paths for exchanging model updates with neighbouring clients, driven by the preferred network scenario as identified by the Outer Stage optimization.
Inner Stage: MPTCP Paths
Only Models Updates and Reward Monitors
Local Training Client #1
Local Training Client #2
Local Training Client #3
Local Training Client #K

Figure 5.4: The stages and framework layout of our proposed DFL over MPTCP

network during the learning process. This phase uses methods to guarantee that the effective subflows are located and used, thereby enhancing the performance and efficiency of the DFL network.

2. **Outer Stage (Network topology optimisation)**: This stage concentrates on dynamically and regularly adopting the most suitable network topology for the DFL operations over MPTCP. This involves enhancing network reliability and learning performance.

Subflows use optimisation: This stage uses the concept of the model-based MPTCP algorithm, combined with our reward-based criteria detailed in Algorithm 5.1. This method, inspired by the Upper Confidence Bound (UCB1) algorithm from [32], aims to maximise the efficiency of using multiple subflows in the DFL network. It tackles packet scheduling, a method commonly employed to address the well-known multi-armed bandit problem, aiming to optimise path selection to reduce delays and improve data throughput.

In our approach, time is partitioned into scheduling intervals (SI), with each SI comprising multiple epochs. Within a complete cycle of DFL, multiple SIs may occur. During each epoch, every client representing a client in the collaborative network probabilistically selects one of N potential actions. Here, each action in the DFL over MPTCP corresponds to the selection of subflows for model updates. These actions, denoted by integers i where $1 \leq i \leq N$, signify the available subflows for a particular network topology, with the total number of actions N equivalent to the total number of available subflows. After performing an action, each client receives a reward assignment, which forms an infinite sequence denoted as $\{x(1), x(2), \ldots\}$, representing the rewards associated with each action over time. We assume that each client is only aware of the rewards obtained from previously selected actions within its vicinity. The rewards for each action are independently and identically distributed according to an unknown distribution with an unknown expected value μ_i.

Each client employs an algorithm, denoted A, can determine the next action based on its past actions and the rewards received from its neighbours. Let $T_i(n)$ denote the number of times the action i has been selected by the client A after n tests. A common metric for evaluating the performance of multi-armed bandit algorithms is a regret metric, which measures the potential loss in reward due to not always choosing the optimal action. For a given client A, the regret after n trials is quantified by the difference between the total expected reward of always selecting the optimal action and the accumulated expected rewards from the actions taken, as follows:

$$R(n) = \mu^* \cdot n - \sum_{j=1}^{N} \mu_j \cdot E[T_j(n)], \tag{5.9}$$

where:

- $R(n)$ is the total regret after n trials.
- $\mu^* = \max\limits_{1 \leq i \leq N} \mu_i$ represents the maximum expected reward among all actions, indicating the reward of the optimal action.
- N is the total number of available actions or arms (i.e., number of available subflows).
- μ_j is the expected reward of action j.

- $E[T_j(n)]$ denotes the expected number of times action j has been selected up to trial n.

Network topology optimisation: The external phase incorporates Reinforcement Learning (RL) techniques to continuously evaluate and adapt to the prevailing network conditions. This RL-based approach constructs an optimal network scenario policy tailored for DFL users, empowering the system to dynamically respond to fluctuations and uphold superior learning and communication efficacy across diverse network conditions. In our research, we delineate three specific scenarios' options, as elaborated in Sec. 5.3.1.

The foundation of our network performance optimisation strategy lies in the application of the Q-Learning algorithm (i.e., a common RL model), as encapsulated by Eq. (5.10):

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha[R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t)], \tag{5.10}$$

where $Q(S_t, A_t)$ represents the value of taking action A_t in state S_t, α is the learning rate determining the weight of new information over past knowledge, R_{t+1} is the reward received after taking action A_t, γ is the discount factor that balances the importance of immediate versus future rewards, and $\max_a Q(S_{t+1}, a)$ denotes the maximum reward that can be achieved from the next state S_{t+1}, considering all possible actions a.

Therefore, our model proposes the utilisation of a combination of our reward monitors algorithm and experience replay [33] for optimising decision-making algorithms and providing smart learning at the outer optimisation stage by allowing the algorithm to not only learn from its current actions (and their immediate outcomes) but also from a broader set of past experiences. This can lead to a more informed and nuanced approach to minimising regret. Experience replay is a strategy utilised in reinforcement learning and a fundamental component of Q-Learning. It revolves around storing experiences, like sequences of state-action-reward-state, encountered during interactions with an environment within a designated replay buffer. This combination facilitates efficient and fair bandwidth allocation of DFL, steering clear of instability and oscillations. The proposed approach effectively integrates successful features of RL in continuous multipath band-

Algorithm 5.2 Dynamic Network Topology Algorithm for our DFL using MPTCP (DNTA)

```
Require: Incorporate reward monitor Algorithm 5.1 and reinforcement learning (RL)
    mechanisms.
 1: Initialise:
        • Set the scheduling intervals (SI) time.
        • Set the network architecture scenarios.
        • Set the potential actions for each client.
 2: Subflows' Selection Stages (Inner Stage):
 3: for each epoch in the DFL learning process do
        - Model-based MPTCP algorithm (e.g., OLIA) as inner control optimisation stage.
        - Upon packet loss or $RTT$ observation on path $i$, trigger multiplicative decrease.
        - Upon receiving acknowledgment from path $i$, employ continuous increase in $\omega_i$.
 4: end for
 5: Network Scenario Stage (Outer Stage):
 6: Initialise the network topology.
 7: Set up the initial state with all MPTCP connections.
 8: Dynamically switch among the following scenarios:
        • Simultaneous Subflows Utilisation Topology.
        • Strategic Subflow Allocation Topology.
        • Master Subflow Priority Topology.
 9: for each scheduling interval do
        - Initialise: The Q-table for the Q-learning algorithm.
        - Explore Actions: For the current state $S_t$, select a topology scenario action.
        - Transition to Next State $S_{t+1}$: Implement action and transition to new state
          $S_{t+1}$.
        - Select Action with Highest Q-value: From state $S_{t+1}$, choose the action
          with the highest Q-value.
        - Update the Q-table using Eq. (5.10).
        - Update the Network State: Set $S_{t+1}$ as the new current state.
10: end for
11: Return: Optimal network scenario from step 8 and then repeat step 2.
12: Repeat the Process: Continue until the optimisation goal is achieved.
```

width allocation and intelligent scheduling while benefiting from fairness and stability offered by model-based MPTCP mechanisms.

Consequently, upon detecting discrepancies between network conditions and learned policies, our model swiftly abandons outdated strategies in favour of new ones. Thus, this algorithm is proposed as it not only incorporates the optimal path chosen by the client but also assesses its efficiency across three distinct scenarios elucidated at the outset of Sec. 5.3.1. By evaluating the efficacy of the selected path within these scenarios, we gain invaluable information on the performance of DFL under diverse network conditions.

5.3.4 Learning Criteria and Framework Architecture of DFL

This section delineates an overview of DFL architecture, emphasising the model weight parameter updating and transmission process across networked edge clients' devices. To further augment the efficiency of the DFL learning setup, our approach integrates the use of optimised MPTCP, a significant enhancement over traditional TCP, allowing multiple parallel data streams to increase the reliability and throughput of communication between devices. This is achieved by adeptly managing the available subflows, ensuring an optimised use of network resources, which in turn facilitates a more efficient and reliable model training and data exchange process within the DFL framework.

Our architecture utilises a collection of K edge devices, each denoted as a client within a set $\varphi = \{1, \ldots, K\}$, strategically dispersed across a vast network in accordance with a stationary Poisson Point Process (PPP) [34], characterised by a density parameter (λ). These clients are situated within disk cells, uniformly spread out to ensure comprehensive coverage. Each client $i \in \varphi$ is equipped with a dataset $\mathfrak{D}_{(i)}$, which comprises m pairs of instance-label samples (X_i^n, Y_i^n). Here, X_i^m and Y_i^m signify the input and output for the i^{th} device, sequentially, with $m = 0, \ldots, M$. These samples originate from datasets that are either homogeneous or heterogeneous in nature, adhering to an undisclosed probability distribution $p(x, y)$, and might exhibit interactions among distinct sets (e.g., $\mathfrak{D}_{(i)}$ and $\mathfrak{D}_{(j)}$, for $i \neq j$). Each instance X_i^n falls within X_i, a subset of the global instance space X, which collectively forms the basis for our supervised learning models, accommodating both binary classification and regression tasks.

To enhance communication reliability and throughput among the clients within the DFL network, we introduce the implementation of an optimised MPTCP. This innovative approach leverages multiple available network paths (subflows) to facilitate efficient and reliable data exchange across the network, ensuring seamless and robust interactions among the clients.

Within this system, all clients are synchronised to operate on a consistent machine-learning model, such as a Convolutional Neural Network (CNN) [35], which incorporates a shared matrix of weight parameters $\mathbf{W}$. The primary ambition of our system design is to diminish the cross-entropy loss function's variance between anticipated and actual outputs as follows:

$$\arg\min_{\mathbf{W}} F(\mathbf{W}) \equiv \frac{1}{A_i} \sum_{i \in \varphi} f_i(\mathbf{W}), \tag{5.11}$$

where $F(\mathbf{W})$ represents the overarching loss function and $f_i(\mathbf{W})$ the localised loss per device i, with a total number of devices achieving successful transmission to the i^{th} participant across each training iteration can be denoted as (A_i).

The local loss function for device i is derived from the cross-entropy loss pertinent to its dataset $\mathfrak{D}_{(i)}$, calculated as:

$$f_i(\mathbf{W}) = \frac{1}{M} \sum_{m=0}^{M} l(h_{\mathbf{W}}(X_i^m), y_i^m), \tag{5.12}$$

$$m = 0, 1, 2, ..., M$$

where m represents a subset of the overall number of local datasets, and M equals the size of the dataset for participant i normalised by the batch size $(\mathbf{B})$, that is $(M = \frac{|\mathfrak{D}_{(i)}|}{B_i})$. The function $l(h_{\mathbf{W}}(X_i^{(m)}), y_i^{(m)})$ signifies the cost function associated with the weights matrix $\mathbf{W}$ when evaluated against a hypothesis $h_{\mathbf{W}}(X_i^m)$ using data samples X_i^m, exemplified in the case of simple linear regression as $h_{\mathbf{W}}(X) = \mathbf{W}_0 + \mathbf{W}_1 X$.

During the t^{th} iteration of the DFL process, each device $i \in \varphi$ updates its local parameter weights matrix $\mathbf{W}_i^{(t)}$ to enhance model accuracy and pinpoint the most suitable solution for the problem at hand.

The DFL methodology incorporates a dual-level optimisation strategy: at the local model level, each device employs a conventional machine learning optimiser, such as the Stochastic Gradient Descent (SGD) algorithm, to adjust local parameters according to its dataset. At the global model level, a global optimiser aggregates parameters from neighbouring devices using the Federated Averaging algorithm (FedAvg) or any other suitable optimiser techniques to refine and update the global model.

Each participating device i undertakes M training iterations over its local dataset $\mathfrak{D}_{(i)}$ to contemporaneously modify its local model weights via SGD, employing a learning rate η and batch size B_i, demonstrated as follows:

$$\nabla f_i(\mathbf{W}_i^t) = \frac{1}{M}\sum_{m=0}^{M} \nabla l(h_{(\mathbf{W}_i^t)}(X_i^m, y_i^m)), \tag{5.13}$$

$$\mathbf{W}_i^t := \mathbf{W}_i^t - \eta_i \nabla f_i(\mathbf{W}_i^t), \tag{5.14}$$

$$\forall\ m = 0, \ldots, M \text{ and } i \in \varphi.$$

Here, $\nabla f_i(\mathbf{W}_i^t)$ indicates the gradient matrix derived from the device i's batch B_i after M training iterations on local data at iteration t, expressing the rate of f_i's change relative to $\mathbf{W}_i$.

After completing a predefined number of iterations, each device having trained its model with adequate hyperparameters shares its model parameters with neighbouring devices via one-hop communication, following a mesh network topology. During the learning process, a single client assumes the role of the main client, which is responsible for aggregating the models from others and updating the global model accordingly. This collaborative approach ensures that each participant not only updates its local parameters based on direct observations but also integrates insights gathered from neighbours, thereby collectively enhancing the global model accuracy and reliability through the efficient utilisation of MPTCP for improved communication efficacy.

In subsequent phases, participants aggregate these updated parameters $\mathbf{W}_i^t$ from their neighbours, adhering to wireless communication standards, to perform FedAvg, thereby generating a new, refined global model at each iteration. This continuous exchange and aggregation foster a dynamic, server-free learning environment, incrementally improving model performance towards a predetermined convergence threshold ε_k, as depicted in the following condition:

$$\arg\min_{\mathbf{W}} \nabla F_i(\mathbf{W}) \triangleq \left[\left(\frac{1}{(A_i+1)} (\hat{\nabla} f_i(\hat{\mathbf{W}}_i^t) + \sum_{s_i=1}^{A_i} \hat{\nabla} f_{s_i}(\hat{\mathbf{W}}_{s_i}^t)) \right| A_i \geq 1 \right] \leq \varepsilon_k. \tag{5.15}$$

where A_i is the expected number of devices achieving successful transmission to the main client at each epoch, and the convergence threshold ε_k can be determined based on the target outcomes of the model.

In any case, clients are assumed to proceed to share these parameters and loss function metrics over a wireless network, employing a peer-to-peer communication scheme, thereby collectively advancing towards achieving the system's defined objectives.

Our proposal stands to significantly enhance the DFL process through the integration of MPTCP and the strategic use of available network subflows. This optimisation not only fosters a more efficient data exchange but also reinforces the communication backbone of our distributed learning architecture, thereby elevating the network's reliability and throughput.

A synopsis of the algorithm deployed is presented in Algorithm 5.3.

5.3.5 Performance Metrics

To evaluate the effectiveness of our DFL for solving the proposed classification problem, we employ two key metrics commonly utilised in the realm of Machine Learning (ML):

1. **Categorical Cross-Entropy Loss**: This metric quantifies the divergence between the actual probability distribution of target categories and the model's predicted probability distribution. It is a crucial measure for evaluating the performance of classification models. The calculation of this loss is as follows:

$$\text{Loss} = -\sum_{i=1}^{D} \sum_{j=1}^{C} y_{ij} \log(\hat{y}_{ij}) \tag{5.16}$$

Algorithm 5.3 Local Update Algorithm and FL Training (Procedure in each client k)

```
Require: Set subflows selection policy based on Algorithm5.2
Require: Local training set $D_k$, Test set $D_{\text{test},k}$, threshold $\beta$
Ensure: Local weight vector $\mathbf{W}_k$
 1: Initialise $t \leftarrow 0$, $\mathbf{W}_k^{(0)} \leftarrow 0$
 2: Initialise $\beta$ as a constant
 3: repeat
 4:     Server aggregates model updates $\mathbf{W}(t)$ from clients
 5:     Compute global model
 6:     Share global model to update local model
 7:     for each client $k$, each local epoch do
 8:         for samples $D_k$ train the model locally do
 9:
              $\mathbf{W}_k^{(t+1)} = \mathbf{W}_k^{(t)} - \eta\frac{1}{B}\sum_{(x_i,y_i)\in b}\nabla F(\mathbf{W}_k^{(t)};(x_i,y_i))$
10:             $t = t + 1$
11:             $accuracy = \text{Test}(\mathbf{W}_k^{(t)}, D_{\text{test},k})$
12:             if Accuracy $< \beta$ then
13:                 Send $\mathbf{W}_k^{(t)}$ to server
14:             end if
15:         end for
16:     end for
17: until Accuracy $\geq \beta$ return $\mathbf{W}_k$
```

where D represents the total number of observations in the dataset, C is the count of distinct classes, y_{ij} denotes the actual probability that the i^{th} observation belongs to the j^{th} class, and $\hat{y}_{ij}$ is the corresponding predicted probability. Lower scores of loss are indicative of superior classification performance by the model.

2. **Accuracy**: It measures the proportion of correct predictions made by the model, which includes both true positive and true negative predictions, relative to all predictions made on the dataset. The formula for calculating accuracy is given by:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{5.17}$$

where TP denotes the number of true positives, TN denotes the number of true negatives, FP represents the number of false positives, and FN signifies the number of false negatives.

Additionally, we consider model complexity and latency as integral metrics for assessing our DFL model. This encompasses an examination of the model's structure, the learning dynamics, and the inter-device communication mechanisms through MPTCP. The latency refers to the time it takes for model updates to be communicated between participating clients in the network. Due to the decentralised nature, there is no central server, and clients often communicate directly with each other, potentially over multiple network paths simultaneously. On the other hand, the complexity of the model's architecture, notably the quantity of parameters in the global model, significantly influences overall complexity. This effect is particularly pronounced in larger models, where an increased parameter count typically indicates greater complexity. The communication process is a further complexity factor, influenced by the number of devices contributing to the learning process and the requisite communication iterations for model training, is another critical factor. Execution duration further contributes to an understanding of model complexity. Hence, it is imperative to achieve an optimal balance between model complexity and achievable accuracy during the design phase of the DFL model.

In our model, we acknowledge that the latency in TCP connections is primarily affected by two factors: the Round Trip Time (RTT) and the congestion control mechanism. Furthermore, we extend our analysis to include the impact of the model size and dataset size on latency. The foundational equation to represent the latency encountered by a data packet is:

$$L_{\text{TCP}} = \text{RTT} + \frac{C + \delta(M, D)}{CW}, \tag{5.18}$$

where:

- L_{TCP} represents the latency experienced in TCP protocols.
- RTT denotes the round trip time.
- C symbolises a coefficient reflecting the effects of congestion control techniques, such as slow start and congestion avoidance phases.
- CW indicates the size of the congestion window.

- $\delta(M, D)$ is an additional delay factor introduced by the size of the ML model (M) and the dataset (D), accounting for data preprocessing and model loading times.

On the other hand, MPTCP uses multiple subflows to reduce total latency, enhancing load balancing and redundancy. Thus, the determination of latency in MPTCP requires a careful approach because it considers numerous paths in its workings and also involves large ML models and dataset sizes. The estimated latency for MPTCP is given as:

$$L_{\text{MPTCP}} = \min(\text{RTT}_1, \ldots, \text{RTT}_n) + \frac{C + \delta(M, D)}{CW_1 + \ldots + CW_n}, \tag{5.19}$$

- L_{MPTCP} indicates the latency in MPTCP configurations.
- RTT_i is the round trip time for the i-th path.
- CW_i denotes the congestion window size for the i-th path.
- n is the number of paths.
- $\delta(M, D)$, as before, represents the delay factor due to the ML model and dataset size.

Consequently, the primary objective is to boost network performance and decrease latency through the deployment of our DFL model leveraging based-model MPTCP. This approach aims to enhance the accuracy and minimise the loss of DFL models, thereby optimising overall system efficiency.

5.4 Simulation results and discussion

In our study, we propose the implementation of the based-model MPTCP within a DFL framework to augment throughput, reduce operational costs, and enhance system flexibility, all the while achieving competitive and often superior results compared to other benchmarked models. This advancement is particularly significant given that, although benchmark systems relying on a central server may exhibit fluctuating accuracy levels that often do not meet our model's performance, the central server architecture is inherently vulnerable to being a single point of failure. This vulnerability is critical to recognise, as system failures or communication blockages in centralised models could lead to significant downtime.

In our model, the main advantage lies in effectively managing simultaneous, reliable communication across diverse network subflows. As leveraging the parallel transmission capabilities of MPTCP can potentially increase throughput and optimise bandwidth utilisation. Therefore, we utilise the MPTCP approach, which envisages engaging numerous participants scattered across a two-dimensional confined space. For ease of representation, a sizable circular zone with a radius $R > 0$ is suggested; however, a hexagonal pattern is also viable and would result in marginally differing outcomes by an insignificant constant. The participants' distribution follows a Poisson pattern with network density λ over an area A, meaning the count within A is given as a Poisson random variable with an average λA, where $\lambda > 0$ is the network density, and A denotes the circular network area $A = 2\pi R^2$.

The system's central client, situated at the core of this two-dimensional boundary, serves as the nexus for the DFL model, with participants randomly dispersed in the designated zone. The efficiency of participation and transmission in the learning processing is conditioned by wireless communication constraints, such as channel fading and interference, to closely resemble real-world scenarios.

In DFL, each client i identifies its neighbours via a one-hop communication link, confined within a signal coverage area marked by a radius $r_i < R$.

Participants are independently and arbitrarily positioned within the circular network zone, ensuring each one's distance to the main client is $d_{k0} \leq R$. The simulations quantify the probability of a successful transmission $P(\text{SINR} \geq T_k)$ by assessing the Signal to Interference plus Noise Ratio (SINR) against the network's threshold (T_k). A participant is deemed successfully connected if the SINR exceed the predefined threshold T_k, considering a network space radius $R = 1000$ meters and an intensity $\lambda = 0.1$.

The participants are assumed to be equipped with 4 subflows and varying transmission powers per subflow, and thus, only a finite number can concurrently share and update their parameters, influenced by the network's topology. Diminishing the SINR threshold invites more participants into the learning process but escalates the bandwidth requirement and system latency, necessitating a balance between the success probability and the SINR threshold for optimal throughput, capacity, and acceptable latency.

In this study, we benchmark our model against widely recognised FL frameworks, including the FedAvg model [36], which is introduced by Google and other state-of-the-art models such as FLAIR [37] and Greedy FL [38]. These models have been assumed to use the traditional single path protocol (i.e., TCP) and a central server for exchanging the model updates during the learning process. Our evaluation criteria encompass both the model performance and the system efficiency, comparing our approach with these FL architectures. The simulation setup for these benchmarks FL models involves positioning a central server at the core of a designated circular area with radius R, serving as a hub for a comprehensive network of edge devices and clients.

The network designs are similarly simulated for both our proposal model and the benchmark models (i.e., FedAvg, FLAIR and Greedy FL models), where the clients are represented as a Poisson Point Process (PPP) characterised by a density λ and a total of N participants. The simulation for the benchmark models adopts a simplified approach where all clients, each within a distance $r_i < R$ from the server, have identical transmission power and are randomly arrayed around the centre within the network area. The system leverages Python and Pytorch APIs [39] to train a classification algorithm on the widely recognised CIFAR-10 dataset comprising 60,000 colour images categorically distributed across 10 distinct image classifications.

By aggregating participants' parameters and utilising diverse methodologies (i.e., centralised and decentralised FL), the proposed algorithms aim to allow the clients to train the model locally their datasets and share only the model parameters with either a central server in the traditional FL approach or with the neighbours directly in the DFL approach, and these occur iteratively to update the global model, primarily focusing on

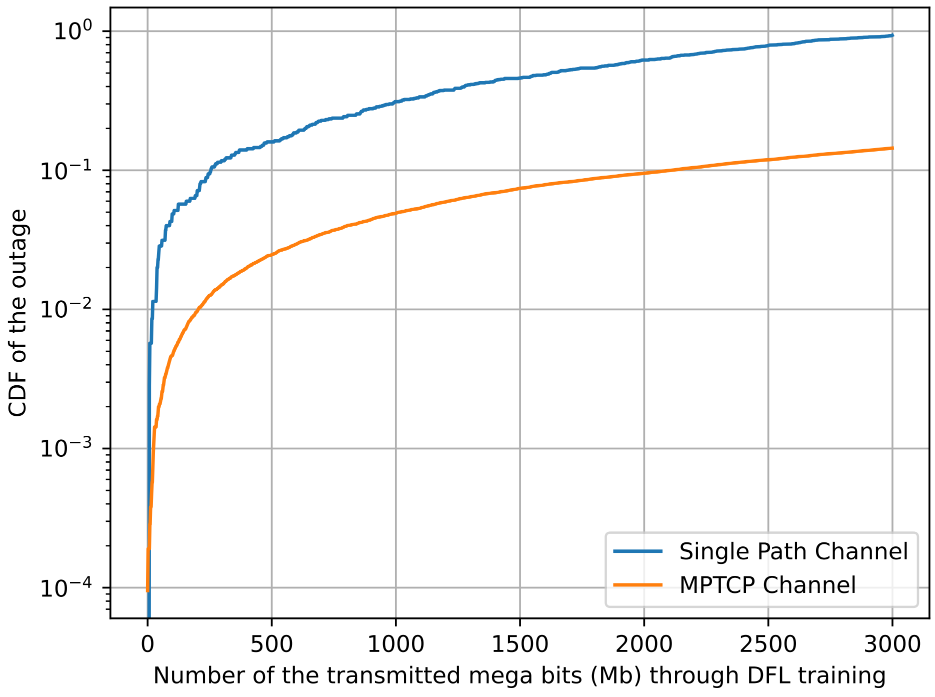


Figure 5.5: Comparison of outage probabilities for exchanging model parameters during DFL.

optimising both the model's accuracy and its loss.

In the optimisation phase of our network, specifically during network topology selection (i.e., *Outage stage* as outlined in Algorithm 5.2), we assess the connectivity performance within the chosen network topology. This evaluation employs a rewards-based approach regularly at every scheduling interval, leveraging the Q-learning algorithm to incorporate communication reliability and network parameters, including device density and participant distribution, and thus dynamically assign the optimum network topology based on a rewards-penalty mechanism that uses the Q-learning algorithm as shown in Eq. (5.10). Based on the assigned topology, the proposed model-based MPTCP in our algorithm, as described in Algorithm 5.2, utilising efficiently the available subflows for exchanging the models' updates with the neighbours to update the global model. Thus, the results of our proposed model for dynamic selecting MPTCP network topology, as illustrated in Fig. 5.5, confirm its efficacy in maintaining lower outage rates by achieving a higher success probability relative to the SINR threshold. This ensures optimal network capacity and robust client connectivity throughout the learning process.

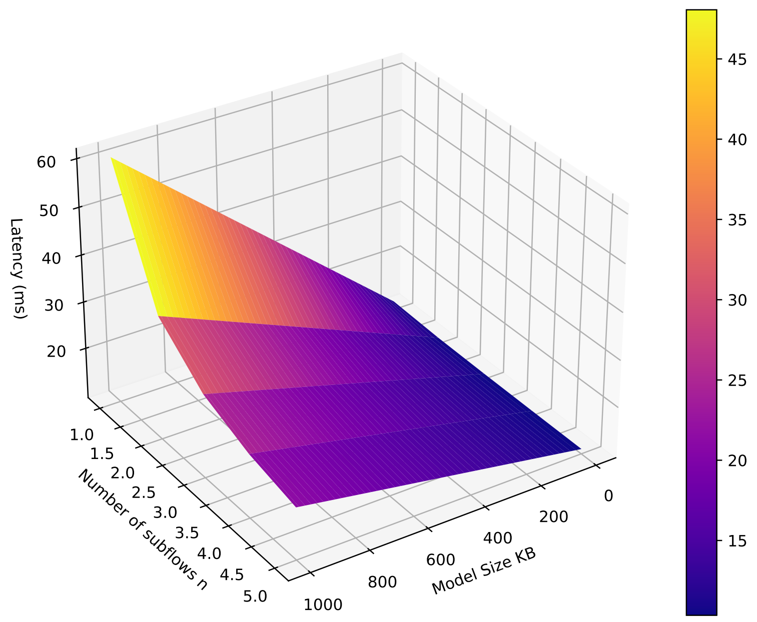


Figure 5.6: Comparative analysis of average latency in DFL models employing TCP and MPTCP protocols for various models' sizes.

Latency per client varies slightly, primarily depending on the iterations needed to reach system convergence. Our findings reveal an average client latency below 15 ms, as shown in Fig. 5.6, assuming uniform computation and broadcast times across clients. Despite leveraging multiple subflows for parallel data transmission and network performance optimisation, the system's flash memory requirements and power consumption determined by GPU usage during the training process remain comparable to other models. It is considered relatively low, particularly since our training can be completed in a few minutes, as illustrated in Fig. 5.7.

Given the criteria for successful transmissions and network capacity, each device within the network autonomously forms a one-hop neighbour group every round to facilitate parameter exchange and collaboratively refine the global model, aiming to enhance model performance. To illustrate system behaviour, results from randomly selected participants in the DFL simulation serve as examples.

Our network design ensures significantly high connectivity success in the DFL model, with every participant capable of collaborating with at least one neighbour. This contrasts with centralised FL models, where the ability of clients to meet communication require-

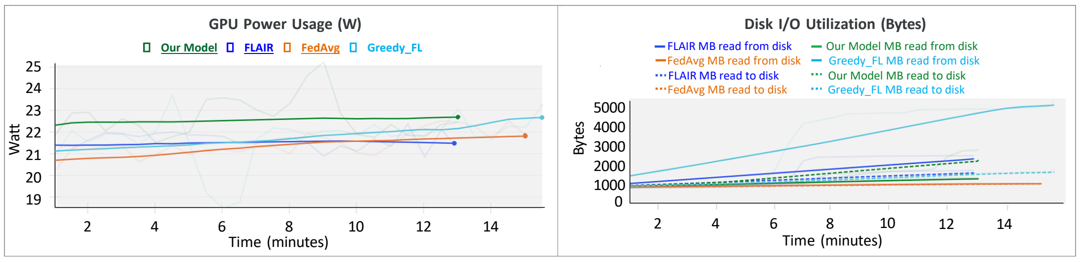


Figure 5.7: Comparative analysis of our model's GPU power consumption and data storage utilisation against other relevant models.

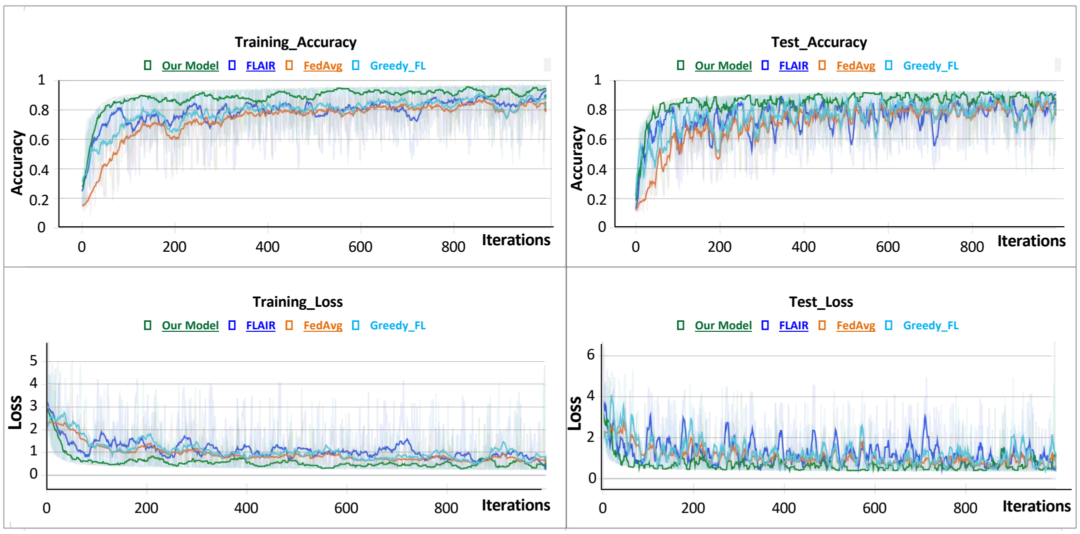


Figure 5.8: Comparative performance of our model's accuracy and loss against other relevant models.

ments and connect with the server for parameter exchange varies. The DFL simulation demonstrates a classification accuracy exceeding 95% within the initial 200 iterations, displaying minimal variance among the participating devices and maintaining latency under 15 ms. Conversely, alternative models report marginally slightly lower accuracy, increased variability between the participants, and heightened latency, attributed to their reliance on a singular communication path during learning.

Results and statistical analyses for the average network performance are depicted in Fig. 5.8 and Table 5.1. Without a central server, the DFL approach records high accuracy and minimal cross-entropy loss. Integrating the MPTCP-based networking model with the DFL framework boosts communication reliability and the count of successfully participating clients.

Model	Client Algo.	Dataset	Accuracy	Loss	Standard Deviation	Latency (ms)	Power Consum. (W)	Flash memory size (MB)
Our Model	CNN	CIFAR-10	**96.50**	**0.038**	**0.115**	**15 ↓**	22.80	1.20
FLAIR	CNN	CIFAR-10	94.10	0.042	0.32	100 ↑	22.80	1.90
FedAvg	CNN	CIFAR-10	85.30	0.085	0.40	100 ↑	**21.50**	**1.02**
Greedy FL	CNN	CIFAR-10	90.20	0.093	0.29	100 ↑	23	4.90

Table 5.1: Comparison of our DFL model using MPTCP against other relevant models' performance and system metrics.

Figure 5.8 highlights the proposed model's achievement of high accuracy and low loss through MPTCP integration. Client records using available subflows indicate parallel learning capabilities, demonstrating consistent progress towards convergence in accuracy and loss. The DFL framework's advantage lies in data retention on participant devices, bolstering privacy. Compared to centralised benchmarks, our decentralised model delivers competitive outcomes without a central server, achieving 95% accuracy, a cross-entropy loss of 0.088, and 15 ms latency over 500 iterations. In contrast, benchmark FL models such as FedAvg, FLAIR, and Greedy FL demonstrate slightly lower accuracy, higher loss, and substantially increased outage probability and latency exceeding 100 ms as depicted in Fig. s 5.5 and 5.6. These issues occur from the reliance on single-path models, which incur higher communication costs and greater instability compared to our optimised multipath approach.

Consequently, integrating DFL with MPTCP minimises latency and loss while accelerating convergence, outperforming traditional FL methods. These findings underscore the DFL model's substantial advancements in classification prediction accuracy using adequate datasets, and thus further enhanced by MPTCP to facilitate more reliable client communication during the learning and collaboration processes.

5.5 Conclusions

Our decentralised learning approach not only mitigates the risks associated with connection instability prevalent in traditional FL approach by eliminating the central point of failure but also demonstrates the potential to exceed the performance of conventional models through enhanced data throughput, cost efficiency, and system flexibility. In effect, our model is capable of achieving competitive outcomes and, in many cases, surpassing the accuracy of other benchmark models in this study, thereby underscoring the efficacy of DFL in addressing the complexities of real-world communication challenges.

To sum up, the DFL model introduced in this study, which utilises MPTCP, capitalises on the available subflows within the assigned network topology to facilitate efficient data transmission and higher throughput. As a consequence, the system secures enhanced reliability in connectivity throughout the learning process, and this leads to rapid convergence, higher accuracy and lower latency, markedly overcoming conventional baselines that struggle to address the challenges of connection instability in DFL networks inherent in real-world communication environments. Although the study employs a relatively straightforward classification problem for illustrative purposes, it carefully examines various efficient models to accurately assess performance outcomes. The improvement achieved by our proposed model is particularly crucial in real-time applications, where both latency and accuracy are essential metrics.

Integrating the DFL methodology with the MPTCP protocol has shown the potential to rival traditional FL frameworks. Simulation results indicate that our developed DFL architecture attains improved latency, enhanced system adaptability, and marginally superior accuracy. All this is accomplished without sharing personal data to preserve the client's privacy, as only the model updates are shared, and without the need for a central server to manage the learning within the network.

References

[1] Y. Liu, T. Yu, Q. Meng, and Q. Liu, "Flow optimization strategies in data center networks: A survey," *Journal of Network and Computer Applications*, p. 103 883, 2024.

[2] S. Ferlin, S. Kucera, H. Claussen, and Ö. Alay, "Mptcp meets fec: Supporting latency-sensitive applications over heterogeneous networks," *IEEE/ACM Transactions on Networking*, vol. 26, no. 5, pp. 2005–2018, 2018.

[3] A. Salama, A. Stergioulis, A. M. Hayajneh, S. A. R. Zaidi, D. McLernon, and I. Robertson, "Decentralized federated learning over slotted aloha wireless mesh networking," *IEEE Access*, vol. 11, pp. 18 326–18 342, 2023.

[4] C. Raiciu, M. Handley, and D. Wischik, "Coupled congestion control for multipath transport protocols," Tech. Rep., 2011.

[5] V. S. Hapanchak, A. Costa, J. Pereira, and M. J. Nicolau, "An intelligent path management in heterogeneous vehicular networks," *Vehicular Communications*, vol. 45, p. 100 690, 2024.

[6] L. Chao, C. Wu, T. Yoshinaga, W. Bao, and Y. Ji, "A brief review of multipath tcp for vehicular networks," *Sensors*, vol. 21, no. 8, p. 2793, 2021.

[7] K. D. Singh and P. Singh, "Fog cloud computing and iot integration for ai enabled autonomous systems in robotics," *EAI Endorsed Transactions on AI and Robotics*, vol. 3, 2024.

[8] A. Badie-Modiri, C. Boldrini, L. Valerio, J. Kertész, and M. Karsai, "Initialisation and topology effects in decentralised federated learning," *arXiv preprint arXiv:2403.15855*, 2024.

[9] H. Su, C. Xiang, and B. Ramesh, "Towards confidential chatbot conversations: A decentralised federated learning framework," *The Journal of The British Blockchain Association*, 2024.

[10] A. Salama, S. A. Zaidi, D. McLernon, and M. M. Qazzaz, "Flcc: Efficient distributed federated learning on iomt over csma/ca," *arXiv preprint arXiv:2304.13549*, 2023.

[11] L. Palmieri, C. Boldrini, L. Valerio, A. Passarella, and M. Conti, "Impact of network topology on the performance of decentralized federated learning," *arXiv preprint arXiv:2402.18606*, 2024.

[12] T. Tsuru, M. Hasegawa, Y. Shoji, K. Nguyen, and H. Sekiya, "An implementation and evaluation of mptcp-based iot router," *Multimedia Tools and Applications*, vol. 82, no. 18, pp. 28 389–28 404, 2023.

[13] Q. De Coninck and O. Bonaventure, "Multipath quic: Design and evaluation," in *Proceedings of the 13th international conference on emerging networking experiments and technologies*, 2017, pp. 160–166.

[14] J. R. Iyengar, P. D. Amer, and R. Stewart, "Concurrent multipath transfer using sctp multihoming over independent end-to-end paths," *IEEE/ACM Transactions on networking*, vol. 14, no. 5, pp. 951–964, 2006.

[15] A. Ford, C. Raiciu, M. Handley, and O. Bonaventure, "Tcp extensions for multipath operation with multiple addresses," *Signal Processing*, 2013.

[16] O. Bonaventure, *Apple music on ios13 uses multipath tcp through load-balancers*, 2019.

[17] T. Use Multipath, "Use multipath tcp to create backup connections for ios," *HT201373*, 2017.

[18] O. Bonaventure and S. Seo, "Multipath tcp deployments," *IETF Journal*, vol. 12, no. 2, pp. 24–27, 2016.

[19] W. Lee, J. Y. Lee, H. Joo, and H. Kim, “An mptcp-based transmission scheme for improving the control stability of unmanned aerial vehicles,” *Sensors*, vol. 21, no. 8, p. 2791, 2021.

[20] A. Gentili, S. Horsmanheimo, L. Tuomimäki, P. Hyvärinen, H. Kokkoniemi-Tarkkanen, and I. Ahmad, “Improving 5g performance in critical environments through mptcp,” in *2023 IEEE 98th Vehicular Technology Conference (VTC2023-Fall)*, IEEE, 2023, pp. 1–5.

[21] P. Hurtig, K.-J. Grinnemo, A. Brunstrom, S. Ferlin, Ö. Alay, and N. Kuhn, “Low-latency scheduling in mptcp,” *IEEE/ACM transactions on networking*, vol. 27, no. 1, pp. 302–315, 2018.

[22] J. Postel, “Transmission control protocol,” Tech. Rep., 1981.

[23] A. Nota, “Implementation and evaluation of multipath tcp (mptcp) schedulers for reliable and low-latency car-to-cloud communication,” Ph.D. dissertation, Politecnico di Torino, 2020.

[24] V. Jacobson, R. Braden, and D. Borman, “Tcp extensions for high performance,” *Signal Processing*, 1992.

[25] Y.-C. Chen, Y.-s. Lim, R. J. Gibbens, E. M. Nahum, R. Khalili, and D. Towsley, “A measurement-based study of multipath tcp performance over wireless networks,” in *Proceedings of the 2013 conference on Internet measurement conference*, 2013, pp. 455–468.

[26] S. Ha, I. Rhee, and L. Xu, “Cubic: A new tcp-friendly high-speed tcp variant,” *ACM SIGOPS operating systems review*, vol. 42, no. 5, pp. 64–74, 2008.

[27] D. Scholz, B. Jaeger, L. Schwaighofer, D. Raumer, F. Geyer, and G. Carle, “Towards a deeper understanding of tcp bbr congestion control,” in *2018 IFIP networking conference (IFIP networking) and workshops*, IEEE, 2018, pp. 1–9.

[28] Y. Cao, A. Jain, K. Sharma, A. Balasubramanian, and A. Gandhi, “When to use and when not to use bbr: An empirical analysis and evaluation study,” in *Proceedings of the Internet Measurement Conference*, 2019, pp. 130–136.

[29] F. Li, J. W. Chung, X. Jiang, and M. Claypool, “Tcp cubic versus bbr on the highway,” in *Passive and Active Measurement: 19th International Conference, PAM 2018, Berlin, Germany, March 26–27, 2018, Proceedings 19*, Springer, 2018, pp. 269–280.

[30] J. Pillmann, D. Behnke, B. Sliwa, M. Priebe, and C. Wietfeld, “Efficient and reliable car-to-cloud data transfer empowered by bbr-enabled network coding,” in *2018 IEEE 88th Vehicular Technology Conference (VTC-Fall)*, IEEE, 2018, pp. 1–5.

[31] R. Khalili, N. Gast, M. Popovic, and J.-Y. Le Boudec, “Mptcp is not pareto-optimal: Performance issues and a possible solution,” *IEEE/ACM Transactions On Networking*, vol. 21, no. 5, pp. 1651–1665, 2013.

[32] P. Auer, N. Cesa-Bianchi, and P. Fischer, “Finite-time analysis of the multiarmed bandit problem,” *Machine learning*, vol. 47, pp. 235–256, 2002.

[33] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, “Prioritized experience replay,” *arXiv preprint arXiv:1511.05952*, 2015.

[34] M. Haenggi, J. G. Andrews, F. Baccelli, O. Dousse, and M. Franceschetti, “Stochastic geometry and random graphs for the analysis and design of wireless networks,” *IEEE journal on selected areas in communications*, vol. 27, no. 7, pp. 1029–1046, 2009.

[35] K. O’shea and R. Nash, “An introduction to convolutional neural networks,” *arXiv preprint arXiv:1511.08458*, 2015.

[36] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, “Communication-efficient learning of deep networks from decentralized data,” in *Artificial intelligence and statistics*, PMLR, 2017, pp. 1273–1282.

[37] A. Sharma, W. Chen, J. Zhao, Q. Qiu, S. Bagchi, and S. Chaterji, “Flair: Defense against model poisoning attack in federated learning,” *ACM ASIA Conference on Computer and Communications Security*, 2023.

[38] M. Mehta and C. Shao, “A greedy agglomerative framework for clustered federated learning,” *IEEE Transactions on Industrial Informatics*, 2023.

[39] A. Paszke, S. Gross, F. Massa, *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, vol. 32, 2019.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

In the four years since I commenced my research, the project has evolved significantly from its initial focus on addressing specific limitations to enhancing the development of DFL through geometric models, advanced communication protocols, and robust various aggregation methods. My journey started with the exploration of existing DFL models, which notably lacked simulation capabilities for dynamically modelling device distribution and had no substantial actuation systems for reliable performance. From these early challenges, I adopted stochastic geometry to precisely model these dynamics, enabling a quantifiable performance assessment of the DFL scheme. The primary goal was to enhance classification accuracy without compromising privacy and to ensure adaptability to network dynamics. This research has led to significant publications, including works that detail how network topologies influence FL performance in practical deployments [1], as well as innovations in aggregation methods tailored for edge computing within DFL frameworks [2].

This study has also tackled the challenges of Byzantine device scenarios within the DFL network, aiming to establish a robust model over ad hoc networks. I developed a Byzantine-resilient DFL model that employs a novel reputation scoring system to detect and eliminate untrustworthy devices while using advanced aggregation and spatial analysis techniques to maintain network throughput. This work is currently under review for publication in IEEE Transactions on Machine Learning in Communications and Networking.

Furthermore, I have explored the integration of Multipath TCP (MPTCP) with DFL to enhance communication reliability during the learning phase. This integration seeks to use MPTCP's capability to manage multiple data paths, thereby reducing latency and improving throughput for collaborative learning applications. I proposed three distinct DFL network topologies, Simultaneous Subflows Utilisation, Strategic Subflow Allocation, and Master Subflow Priority, as part of an innovative strategy that uses Reinforcement Learning algorithms to select the most effective network topology dynamically. This research has recently been submitted to the IEEE Open Journal of the Communications Society and is currently under review.

I have personally written the five articles included in this thesis and collaborated on three additional publications, with more in development. Each solution or finding in this project tends to reveal further questions, marking the ongoing nature of research. In the next sections, I will outline the major proposals in the DFL area and discuss any ongoing or planned work to address these. While not all proposals can be conclusively implemented, and new ones may undoubtedly arise, this summary should provide a clear view of the future proposals of the DFL framework.

6.2 Future Work

The research presented in this thesis lays a strong foundation for numerous exciting avenues of future investigation. The DFL models, as intensively explored throughout this study, show significant potential but also open the door to several improvements and expansions that could further enhance the utility, efficiency, and applicability of DFL frameworks. Below are the critical areas of future work proposed based on the findings and experiences gained during this PhD research:

1. **DFL Optimisation for High-Mobility Environments**: The need for robust DFL systems capable of functioning efficiently in high-mobility environments is paramount, especially in scenarios such as wireless mesh networks used in Driverless Transport Systems. Future research should focus on designing DFL models that cater specifically to the unique challenges posed by high-speed mobility, including rapid changes in network topology and frequent interruptions. This involves not only improving the robustness and scalability of the systems but also enhancing their flexibility to dynamically adapt to the swiftly changing conditions without compromising on performance.

2. **Tailored Networking Protocols for DFL**: Further investigation is required to identify and develop networking protocols that are specifically tailored for DFL im-

plementations over wireless mesh networks. Protocols such as Thread [3], which are designed for low-power, low-data rate transmissions, could be particularly advantageous for IoT systems. Future studies should aim to benchmark and optimise these protocols within DFL contexts, enhancing the overall efficiency and effectiveness of data communications within such networks.

3. **Integration of Green Energy and Blockchain Technology**: As the world moves towards more sustainable technologies, integrating green energy solutions [4] with blockchain technology [5] within DFL architectures offers a promising direction. Blockchain technology enhances security, privacy, and trust within decentralised networks by providing a cryptography framework that ensures data integrity and transparency of transactions. However, it is important to note that while blockchain offers significant advantages, such as immutability and decentralised verification [6], it also introduces challenges. These include scalability issues, energy consumption in proof-of-work systems, and potential privacy concerns related to the transparency of public blockchains [7]. Therefore, while blockchain can substantially improve certain aspects of security and trust in such DFL scenarios, it is not without its trade-offs and limitations, which need to be carefully considered in its implementation to develop greener and more robust DFL systems.

4. **Advanced Network Segmentation Techniques**: Integrating advanced network segmentation techniques [8] into DFL systems could significantly improve learning performance across the network. This approach involves dividing the network into segments or clusters to manage communications and model updates more efficiently. Such segmentation could lead to enhanced performance by reducing latency, managing bandwidth more effectively, and aligning model training processes with the network structure and dynamics more closely.

Each of these areas not only extends the current research but also aligns with the ongoing advancements in technology and network management. By addressing these topics, future work can lead to more sophisticated, adaptable, and efficient DFL systems, ultimately pushing the boundaries of what decentralised learning environments can achieve, especially in dynamic and demanding applications.

References

[1] A. Salama, A. Stergioulis, A. M. Hayajneh, S. A. R. Zaidi, D. McLernon, and I. Robertson, “Decentralized federated learning over slotted aloha wireless mesh networking,” *IEEE Access*, vol. 11, pp. 18 326–18 342, 2023.

[2] A. Salama, A. Stergioulis, S. A. R. Zaidi, and D. McLernon, “Decentralized federated learning on the edge over wireless mesh networks,” *IEEE Access*, vol. 11, pp. 124 709–124 724, 2023.

[3] *Openthread protocol*, Google, 2022. [Online]. Available: `[Online]%20https://openthread.io/`.

[4] X. Liu and N. Ansari, “Toward green iot: Energy solutions and key challenges,” *IEEE Communications Magazine*, vol. 57, no. 3, pp. 104–110, 2019.

[5] Z. Zheng, S. Xie, H. Dai, X. Chen, and H. Wang, “An overview of blockchain technology: Architecture, consensus, and future trends,” in *2017 IEEE international congress on big data (BigData congress)*, Ieee, 2017, pp. 557–564.

[6] G. Tripathi, M. A. Ahad, and G. Casalino, “A comprehensive review of blockchain technology: Underlying principles and historical background with future challenges,” *Decision Analytics Journal*, p. 100 344, 2023.

[7] Z. Wenhua, F. Qamar, T.-A. N. Abdali, R. Hassan, S. T. A. Jafri, and Q. N. Nguyen, “Blockchain technology: Security issues, healthcare applications, challenges and future trends,” *Electronics*, vol. 12, no. 3, p. 546, 2023.

[8] C. Han, M. Dianati, Y. Cao, F. Mccullough, and A. Mouzakitis, “Adaptive network segmentation and channel allocation in large-scale v2x communication networks,” *IEEE Transactions on Communications*, vol. 67, no. 1, pp. 405–416, 2018.