Resilient Federated Learning Framework for 6G

Leonardo Almeida*, Pedro Rodrigues*, Mário Antunes*† and Rui L. Aguiar*†
*Instituto de Telecomunicações, Universidade de Aveiro, Aveiro, Portugal
E-mails: {leonardoalmeida,pedrofrodrigues,mario.antunes,ruilaa}@av.it.pt
†DETI, Universidade de Aveiro, Aveiro, Portugal
E-mails: {mario.antunes,ruilaa}@ua.pt

***Abstract*—Federated Learning enables collaborative model training while preserving data privacy, making it ideal for 5G, 6G, and Internet of Things environments. However, FL faces challenges such as communication costs, node failures, and scalability in dynamic networks. This paper proposes a Resilient Federated Learning Framework that enhances robustness by leveraging Zenoh for efficient communication. Results demonstrate improved training speed and reliability in heterogeneous networks, making FL more adaptable for real-world deployment.**

***Index Terms*—Federated Learning, Resilience, Edge Computing, IoT, Communication Protocols, Distributed Learning**

I. Introduction

Artificial Intelligence (AI) is becoming more prevalent in our daily lives, transforming industries ranging from healthcare and banking to transportation and education. However, the training of ML models, especially with sensitive or distributed data, poses significant challenges. Federated Learning (FL) has emerged as a promising approach that allow collaborative learning without the need for data sharing, addressing critical concerns related to data privacy and regulatory compliance, such as the General Data Protection Regulation (GDPR) and the AI Act.

The potential of FL heavily depends on its resilience and robustness, particularly in the context of 5G and 6G networks, where edge computing and Internet of Things (IoT) devices play a pivotal role. In this context, resilience refers to the ability of the system to maintain its training and performance, even in the presence of node failures, delays, dynamic network topologies and other adversities.

This paper proposes an framework to address these challenges, while highlighting the importance of resilience in FL and the critical role of communication protocols in achieving robust and scalable FL systems. Other challenges include non-iid data distribution, communication costs, and achieving a reliable model aggregation.

The remaining of this document is organized as follows: Section II provides background information and an overview of related work in the field of FL. Section III details the architecture and components of the proposed Resilient Federated Learning Framework. Section IV outlines the experimental setup and methodology used to evaluate the framework. Section V presents and discusses the results of the experiments. Finally, Section VI concludes the paper and highlights future directions for research.

II. Background and Related Work

FL is a paradigm of Distributed Machine Learning where the training process is performed by multiple clients, that under some circumstances, enables faster training and allow the use of larger models and datasets, that would not be possible to process in a single machine [1].

The FL training process can follow different techniques, in [2] the authors present a taxonomy with different approaches each with its own characteristics, advantages and disadvantages, that can be used to handle non-iid data distribution and achieve a reliable model aggregation.

Although FL solves some of the privacy concerns related to data sharing, it introduces new challenges, such as communication costs [3]. Some communication protocols used in distributed learning approaches may not be the best fit for FL [4], for instance, Message Passing Interface (MPI) requires Secure Shell (SSH) connections between the nodes, which can be a security concern, since the workers need to expose their port to the internet. It is also important to minimize the amount of data transferred because some user equipment may be connected via metered connections. However, the majority of state-of-the-art approaches [5], ignore network communication issues and concentrate on enhancing training algorithms to address the remaining drawbacks of FL, making them far from production-ready [6].

Existing works, such as [7], propose a resilient FL system, but lack modularity or do not have their code publicly available. This makes it difficult to adapt these solutions to different scenarios or to understand how they work.

Communication protocols have their features and limitations, in [4] the authors present a comparison between 8 different communication protocols, where Zenoh and Kafka are the most promising ones to ensure resilience and scalability in FL systems. Zenoh is a decentralized, low-latency communication middleware designed for efficient data sharing in dynamic and resource-constrained environments. It excels in minimal overhead, privacy, and fault tolerance, making it ideal for IoT and FL. Kafka is a distributed, broker-based platform for real-time data streaming. It supports scalable, fault-tolerant message storage and is optimized for high-throughput event-driven architectures.

The authors also compare Zenoh with MPI and show that these protocols have similar performance. This is important because Zenoh can be used for a resilient FL system, where

the workers can join and leave the system at any time, while MPI is limited in this regard. The results also showed that FL is reliable in heterogeneous networks and that asynchronous scheduling can benefit from these networks, showing faster convergence than synchronous scheduling.

Finally, preliminary results in [1] showed that FL can achieve a speedup in training time, compared to single training, by varying the number of workers $[2, 4, 8]$, while maintaining similar model accuracy. However, the experiments were done in an environment where the workers are homogeneous.

These findings are the basis for the proposed framework, which aims to improve the resilience and scalability of FL systems, while maintaining the same performance.

III. Proposed Solution

The proposed solution is a Resilient Federated Learning Framework, which aims to address the challenges of FL with a focus on modularity, filling the gaps of existing solutions, and leveraging the most promising communication protocols to ensure resilience and scalability. The framework is a continuation of the work presented in [1], where FL was done with a Decentralized optimization, Syncronous scheduling, and MPI for the communication layer. Figure 1 shows an example of this implementation.

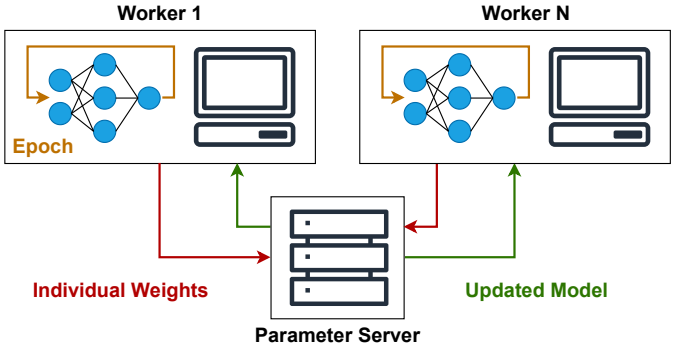

Fig. 1: Decentralized Optimization approach.

MPI is a widely used communication protocol in distributed systems, but as mentioned in Section II is not designed for dynamic and resource-constrained environments. The previous work did not ensure resilience, as it was not designed to handle node failures. The proposed framework aims to address these issues by using Zenoh as communication protocol, while extending the modularity of the system. The presented work is based on 5 modules: FL backend, ML backend, Dataset Manager, Neural Network Architecture and Communication Layer. To allow dynamic network topologies and optimize the communication costs, 2 new modules were added: Worker Manager and Message Layer, Figure 2 shows the framework's modules.

Each module has its own responsibilities and can be easily replaced by another module with the same interface. This allows the system to be easily adapted to different scenarios or use different strategies, such as aggregation methods,

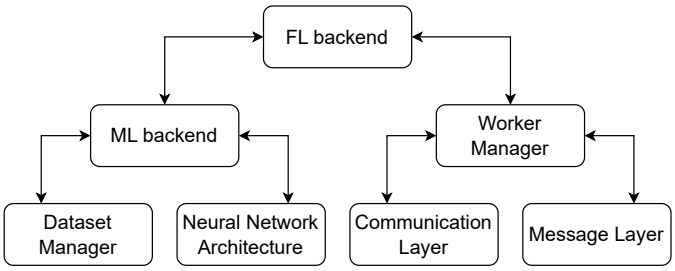

Fig. 2: Resilient Federated Learning Modules.

communication protocols, or scheduling strategies. The new modules will allow the system to choose workers based on their availability and performance, and to optimize the communication costs by compressing the messages.

To allow the FL algorithm to be resilient, a few changes were made to the training process. From all workers, only a subset of them will be selected to participate in each epoch, following round-robin scheduling, allowing the system to adapt to workers leaving or joining the system at any time. At the beginning of each epoch, the system waits for a certain amount of workers to be available and has a threshold of workers that needed to complete their work to finish the epoch. All of these parameters can be configured.

IV. Experimental Methodology

The experimental setup was deployed on an ARM-based server environment using 11 Virtual Machines (VMs) running Ubuntu 24.04 LTS. One VM was provisioned as the parameter server, with 8 CPU cores and 16 GB of RAM and without network bandwidth restrictions, enabling it to handle coordination tasks and higher computational loads without interference from network bottlenecks. The remaining 10 VMs were provisioned as workers, each with 2 CPU cores and 4 GB of RAM to simulate resource-constrained environments. To further emulate realistic distributed network conditions, each worker VM was limited to a bandwidth cap of 50 Mbps.

To accurately measure communication times between nodes, each node mesures timestamps both when sending and receiving messages. But precise clock synchronization across all nodes is crucial for accurate time measurements. To achieve this, the parameter server was configured as Network Time Protocol (NTP) server, and all worker nodes were synchronized with it. This setup minimized time drift between machines, enabling reliable calculation of communication latencies.

The experiments were conducted using the UNSW-NB15 dataset, this dataset was developed by the Australian Centre for Cyber Security (ACCS), combines real modern normal traffic with synthetically generated attack traffic. It contains over 2 million records with 49 features and supports both binary and multi-class classification tasks, with attacks categorized into nine types (e.g., DoS, Exploits, Reconnaissance). In this study we used a preprocessed version that uses NetFlow features, published in [8] by the University of Queensland, with binary

classification, where the task is to classify the traffic as normal or an attack. The dataset was splited into 70% for training, 7% for each worker, 15% for validation by the parameter server and 15% for future testing. Each node normalizes the data using their own statistics, ensuring that the data is not shared between nodes.

In [9], the authors evaluated three types of models (Artificial Neural Network (ANN), Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN)) on the UNSW-NB15 dataset, we selected the ANN model due to its simplicity and faster training time. Despite its simplicity, it performed comparably to the other models. The model consists of three hidden layers with 128, 96, and 64 neurons, followed by a dropout layer (rate = 0.25). ReLU is used as the activation function in all hidden layers, and training is performed using the Adam optimizer.

The experiments were conducted under two main scenarios: with and without node failures. For the failure-free case, six runs were performed, three using MPI and three using Zenoh as the communication protocol, to compare communication times. To evaluate fault tolerance, an additional Zenoh run was carried out where each worker had a 1% chance of failure per second. All experiments used a batch size of 1024, trained over 10 global epochs, with each worker performing 3 local epochs before communicating with the master node, finally, in each epoch the number of workers chosen was 7.

The code used to obtain the results presented is available on Github[1].

V. Results

The results presented in this section are based on the experiments mentioned in Section IV. Starting with the first scenario, where no node failures occurred, for each run, we recorded the average communication time per worker, average local computation time per worker, and the total duration of the run. The results are summarized in Table I.

TABLE I: Comparison of times for different communication protocols. Lower values are better.

Run	Average Comm Time (s)	Average Working Time (s)	Total Duration of the Run (s)
MPI 1	1.036	**162.37**	**261.15**
MPI 2	1.113	164.48	283.25
MPI 3	**1.026**	163.41	273.89
Zenoh 1	0.752	166.37	**261.36**
Zenoh 2	0.826	168.15	267.86
Zenoh 3	**0.669**	**164.07**	262.80

These results show that Zenoh runs consistently achieved lower average communication times compared to MPI, which differs from the results presented in [4], because the authors used a Zenoh wrapper to have the same interface as MPI, which added some overhead to the communication. While

[1]https://github.com/leoalmPT/FlexFL/tree/ICCT

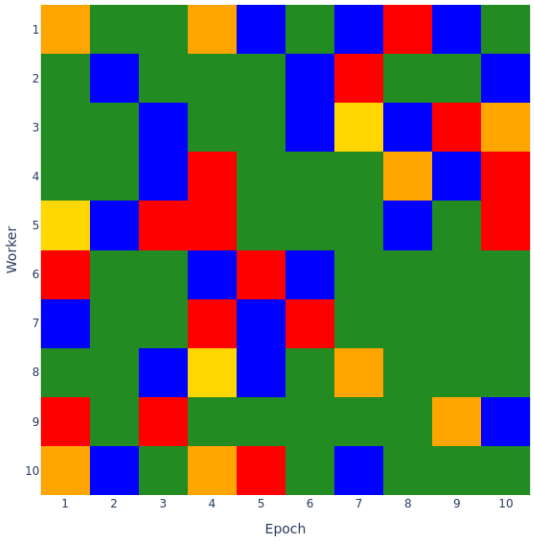

Fig. 3: Worker status in each epoch. Colors indicate status: Blue - i: Idle; Green - ii: Worked: Orange - iii: Failed while idle: Red - iv: Failed while working; Yellow - v: Failed after working.

MPI had slightly lower average working times in some cases, the overall run durations for Zenoh were generally shorter or comparable, with Zenoh 1 and MPI 1 both completing in approximately 261 seconds. These results suggest that Zenoh provides more efficient communication without significantly impacting local computation, making it a viable alternative to MPI in distributed training scenarios.

In the second scenario, where node failures were introduced, we also recorded the worker status in each epoch, which can be categorized into five types: (i) the node was idle, since it was not chosen to participate in the epoch, (ii) the node worked, (iii) the node failed while it was idle, (iv) the node failed while it was working, and (v) the node failed after successfully completing its work.

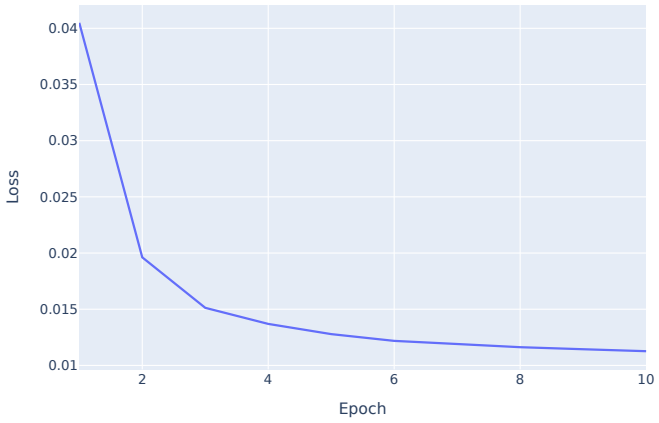

Fig. 4: Validation loss function over epochs.

The results in Figure 3, show that over 10 epochs there

was a total of 26 failures, with 15 of them impacting the training process, since the node was working when it failed. The remaining 11 failures occurred while the node was idle or after successfully completing its work. Despite these failures, the system was able to complete the training process and model converged as shown in Figure 4. The model achieved a validation Matthews Correlation Coefficient (MCC) of 0.9472, accuracy of 0.9958 and F1 score of 0.9958, which are similar to the results obtained in the first scenario. This demonstrates that the proposed framework is resilient to node failures and can maintain model performance even in the presence of such challenges.

VI. Conclusions

This paper introduced a modular and resilient Federated Learning framework tailored for dynamic and heterogeneous environments, such as those expected in 6G and large-scale IoT deployments. By integrating Zenoh as the communication protocol, the framework enhances scalability, fault tolerance, and efficiency, addressing key limitations found in traditional FL implementations.

The results confirm that Zenoh reduces communication overhead compared to MPI, while maintaining similar model performance. Furthermore, the framework successfully handled simulated node failures, maintaining high model accuracy and convergence, which highlights its robustness under real-world conditions.

The modular design allows for flexible experimentation with different communication protocols, aggregation methods, and training strategies, making it a strong foundation for future work in resilient FL systems. This positions the framework as a practical and scalable solution for edge-oriented AI workloads in next-generation networks.

Future directions include integrating more advanced aggregation algorithms and other communication protocols.

Acknowledgements

This study was funded by the PRR - Plano de Recuperação e Resiliência and by the NextGenerationEU funds at University of Aveiro, through the scope of the Agenda for Business Innovation "NEXUS: Pacto de Inovação - Transição Verde e Digital para Transportes, Logística e Mobilidade" (Project nº 53 with the application C645112083-00000059).

References

[1] L. Almeida, P. Rodrigues, R. Teixeira, M. Antunes, and R. L. Aguiar, "Privacy-preserving defense: Intrusion detection in iot using federated learning," in *2024 IEEE 22nd Mediterranean Electrotechnical Conference (MELECON)*, pp. 908–913, IEEE, 2024.

[2] M. Langer, Z. He, W. Rahayu, and Y. Xue, "Distributed training of deep learning models: A taxonomic perspective," *IEEE Transactions on Parallel and Distributed Systems*, vol. 31, no. 12, pp. 2802–2818, 2020.

[3] R. Teixeira, M. Antunes, D. Gomes, and R. L. Aguiar, "The learning costs of federated learning in constrained scenarios," in *2023 10th International Conference on Future Internet of Things and Cloud (FiCloud)*, pp. 18–25, IEEE, 2023.

[4] R. Teixeira, G. Baldoni, M. Antunes, D. Gomes, and R. L. Aguiar, "Leveraging decentralized communication for privacy-preserving federated learning in 6g networks," *Available at SSRN 4817067*, 2024.

[5] W. V. Solis, J. M. Parra-Ullauri, and A. Kertesz, "Exploring the synergy of fog computing, blockchain, and federated learning for iot applications: A systematic literature review," *IEEE Access*, 2024.

[6] 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*, vol. 10, no. 4, pp. 3642–3663, 2022.

[7] K. Jayaram, V. Muthusamy, G. Thomas, A. Verma, and M. Purcell, "Adaptive aggregation for federated learning," in *2022 IEEE International Conference on Big Data (Big Data)*, pp. 180–185, IEEE, 2022.

[8] M. Sarhan, S. Layeghy, and M. Portmann, "Towards a standard feature set for network intrusion detection system datasets," *Mobile Networks and Applications*, vol. 27, pp. 357–370, Nov. 2021.

[9] R. Abou Khamis and A. Matrawy, "Evaluation of adversarial training on different types of neural networks in deep learning-based idss," in *2020 international symposium on networks, computers and communications (ISNCC)*, pp. 1–6, IEEE, 2020.