



*A mi familia, mi madre y mis abuelos.*

*A mis compañeros, Ibon, José Luis, Daniel y Ame.*

*A mis tutores, Gregorio, Pedro Miguel y Enrique Tomás Martínez.*

# Índice

	Página
Declaración firmada sobre originalidad del trabajo	vii
Resumen	1
Extended Abstract	2
1. Introducción	8
2. Background	11
2.1. Aprendizaje federado . . . . .	11
2.1.1. Motivación del DFL . . . . .	11
2.1.2. Fundamentos del DFL . . . . .	12
2.2. Inteligencia artificial generativa . . . . .	13
2.2.1. Enfoques tradicionales . . . . .	13
2.2.2. Modelos de difusión . . . . .	14
3. Estado del Arte	16
3.1. Modelos de difusión . . . . .	16
3.2. Implementaciones de DFL . . . . .	16
3.3. Aplicación de GAI en FL . . . . .	17
3.4. Aplicación de DMs en CFL . . . . .	18
3.5. Motivación . . . . .	19
4. Objetivos y Metodología	23
4.1. Objetivos . . . . .	23
4.2. Metodología . . . . .	23
4.2.1. Estudio del estado del arte . . . . .	23
4.2.2. Estrategia de compartición de etiquetas . . . . .	24
4.2.3. Diseño del entorno . . . . .	24
4.2.4. Implementación modular del entorno . . . . .	24
4.2.5. Evaluación del entorno . . . . .	25
5. Diseño e Implementación de la Solución	26
5.1. Implementación de los nodos . . . . .	27
5.2. Algoritmo federado descentralizado . . . . .	29
5.3. Modularización del entorno . . . . .	31
6. Análisis de Resultados	33

6.1. Topología anillo . . . . .	34
6.2. Topología personalizada . . . . .	38
<b>7. Conclusiones y Vías Futuras</b>	<b>46</b>
<b>I. Muestras generadas en MNIST</b>	<b>48</b>
<b>II. Muestras generadas en FashionMNIST</b>	<b>56</b>
<b>IIIMuestras generadas en EMNIST Letters</b>	<b>65</b>
<b>Siglas</b>	<b>73</b>

## Índice de figuras

1.	Diagrama del funcionamiento a alto nivel de la aplicación. . . . .	26
2.	Arquitectura U-Net. Cada bloque representa a un mapa de características multi-canal. . . . .	28
3.	Estructura interna del entorno . . . . .	32
4.	Estructura interna del entorno - Evaluación . . . . .	33
5.	Topología anillo . . . . .	34
6.	Topología personalizada . . . . .	37

## Índice de tablas

1.	Estado del arte en aprendizaje federado y modelos generativos . . . . .	20
1.	Estado del arte en aprendizaje federado y modelos generativos . . . . .	21
1.	Estado del arte en aprendizaje federado y modelos generativos . . . . .	22
2.	Algoritmo federado descentralizado . . . . .	29
3.	Algoritmo de entrenamiento local . . . . .	30
4.	Resultados del caso de prueba MNIST sin estrategia de compartición en 50 rondas   Topología anillo . . . . .	34
5.	Resultados del caso de prueba MNIST con estrategia de compartición (10 %) en 50 rondas   Topología anillo . . . . .	35
6.	Resultados del caso de prueba FashionMNIST con estrategia de compartición (10 %) en 50 rondas   Topología anillo . . . . .	36
7.	Resultados del caso de prueba FashionMNIST con estrategia de compartición (10 %) en 100 rondas   Topología anillo . . . . .	36
8.	Resultados del caso de prueba FashionMNIST con estrategia de compartición (20 %) en 50 rondas   Topología anillo . . . . .	36
9.	Resultados del caso de prueba EMNIST Letters con estrategia de compartición (10 %) en 50 rondas   Topología anillo . . . . .	37
10.	Resultados del caso de prueba EMNIST Letters con estrategia de compartición (20 %) en 50 rondas   Topología anillo . . . . .	37
11.	Resultados del caso de prueba MNIST sin estrategia de compartición en 50 rondas   Topología personalizada . . . . .	38
12.	Resultados del caso de prueba MNIST sin estrategia de compartición en 50 rondas - 2   Topología personalizada . . . . .	38
13.	Resultados del caso de prueba MNIST sin estrategia de compartición en 100 rondas   Topología personalizada . . . . .	39
14.	Resultados del caso de prueba MNIST sin estrategia de compartición en 100 rondas - 2   Topología personalizada . . . . .	39
15.	Resultados del caso de prueba MNIST con estrategia de compartición (10 %) en 100 rondas   Topología personalizada . . . . .	40
16.	Resultados del caso de prueba MNIST con estrategia de compartición (10 %) en 100 rondas - 2   Topología personalizada . . . . .	40
17.	Muestras del caso de prueba FashionMNIST sin estrategia de compartición en 100 rondas   Topología personalizada . . . . .	41
18.	Muestras del caso de prueba FashionMNIST sin estrategia de compartición en 100 rondas - 2   Topología personalizada . . . . .	41
19.	Muestras del caso de prueba FashionMNIST con estrategia de compartición (10 %) en 100 rondas   Topología personalizada . . . . .	41

20.	Muestras del caso de prueba FashionMNIST con estrategia de compartición (10 %) en 100 rondas - 2   Topología personalizada . . . . .	42
21.	Muestras del caso de prueba FashionMNIST con estrategia de compartición (20 %) en 100 rondas   Topología personalizada . . . . .	42
22.	Muestras del caso de prueba FashionMNIST con estrategia de compartición (20 %) en 100 rondas - 2   Topología personalizada . . . . .	42
23.	Muestras del caso de prueba EMNIST Letters sin estrategia de compartición en 100 rondas   Topología personalizada . . . . .	43
24.	Muestras del caso de prueba EMNIST Letters sin estrategia de compartición en 100 rondas - 2   Topología personalizada . . . . .	43
25.	Muestras del caso de prueba EMNIST Letters con estrategia de compartición (10 %) en 100 rondas   Topología personalizada . . . . .	43
26.	Muestras del caso de prueba EMNIST Letters con estrategia de compartición (10 %) en 100 rondas - 2   Topología personalizada . . . . .	44
27.	Muestras del caso de prueba EMNIST Letters con estrategia de compartición (20 %) en 100 rondas   Topología personalizada . . . . .	44
28.	Muestras del caso de prueba EMNIST Letters con estrategia de compartición (20 %) en 100 rondas - 2   Topología personalizada . . . . .	44
29.	Comparación de resultados con la literatura . . . . .	45
30.	Muestras del caso de prueba MNIST sin estrategia de compartición en 50 rondas   Topología anillo . . . . .	48
31.	Muestras del caso de prueba MNIST con estrategia de compartición (10 %) en 50 rondas   Topología anillo . . . . .	49
32.	Muestras del caso de prueba MNIST sin estrategia de compartición en 50 rondas   Topología personalizada . . . . .	50
33.	Muestras del caso de prueba MNIST sin estrategia de compartición en 50 rondas - 2   Topología personalizada . . . . .	51
34.	Muestras del caso de prueba MNIST sin estrategia de compartición en 100 rondas   Topología personalizada . . . . .	52
35.	Muestras del caso de prueba MNIST sin estrategia de compartición en 100 rondas - 2   Topología personalizada . . . . .	53
36.	Muestras del caso de prueba MNIST con estrategia de compartición (10 %) en 100 rondas   Topología personalizada . . . . .	54
37.	Muestras del caso de prueba MNIST con estrategia de compartición (10 %) en 100 rondas - 2   Topología personalizada . . . . .	55
38.	Muestras del caso de prueba FashionMNIST con estrategia de compartición (10 %) en 50 rondas   Topología anillo . . . . .	56
39.	Muestras del caso de prueba FashionMNIST con estrategia de compartición (10 %) en 100 rondas   Topología anillo . . . . .	57

40.	Muestras del caso de prueba FashionMNIST con estrategia de compartición (20 %) en 50 rondas   Topología anillo . . . . .	58
41.	Muestras del caso de prueba FashionMNIST sin estrategia de compartición en 100 rondas   Topología personalizada . . . . .	59
42.	Muestras del caso de prueba FashionMNIST sin estrategia de compartición en 100 rondas - 2   Topología personalizada . . . . .	60
43.	Muestras del caso de prueba FashionMNIST con estrategia de compartición (10 %) en 100 rondas   Topología personalizada . . . . .	61
44.	Muestras del caso de prueba FashionMNIST con estrategia de compartición (10 %) en 100 rondas - 2   Topología personalizada . . . . .	62
45.	Muestras del caso de prueba FashionMNIST con estrategia de compartición (20 %) en 100 rondas   Topología personalizada . . . . .	63
46.	Muestras del caso de prueba FashionMNIST con estrategia de compartición (20 %) en 100 rondas - 2   Topología personalizada . . . . .	64
47.	Muestras del caso de prueba EMNIST Letters con estrategia de compartición (10 %) en 50 rondas   Topología anillo . . . . .	65
48.	Muestras del caso de prueba EMNIST Letters con estrategia de compartición (20 %) en 50 rondas   Topología anillo . . . . .	66
49.	Muestras del caso de prueba EMNIST Letters sin estrategia de compartición en 100 rondas   Topología personalizada . . . . .	67
50.	Muestras del caso de prueba EMNIST Letters sin estrategia de compartición en 100 rondas - 2   Topología personalizada . . . . .	68
51.	Muestras del caso de prueba EMNIST Letters con estrategia de compartición (10 %) en 100 rondas   Topología personalizada . . . . .	69
52.	Muestras del caso de prueba EMNIST Letters con estrategia de compartición (10 %) en 100 rondas - 2   Topología personalizada . . . . .	70
53.	Muestras del caso de prueba EMNIST Letters con estrategia de compartición (20 %) en 100 rondas   Topología personalizada . . . . .	71
54.	Muestras del caso de prueba EMNIST Letters con estrategia de compartición (20 %) en 100 rondas - 2   Topología personalizada . . . . .	72



# Declaración firmada sobre la originalidad del trabajo

D. **Maxim Utica Babyak**, con DNI **54801827A**, estudiante de la titulación de **Grado de Ingeniería Informática** de la Universidad de Murcia y autor del tfg titulado "**Análisis de la aplicación de modelos de difusión en aprendizaje federado descentralizado**"

De acuerdo con el Reglamento por el que se regulan los Trabajos Fin de Grado y de Fin de Máster en la Universidad de Murcia (aprobado C. de Gob. 30-04-2015, modificado 22-04-2016 y 28-09-2018), así como la normativa interna para la oferta, asignación, elaboración y defensa de los Trabajos Fin de Grado y Fin de Máster de las titulaciones impartidas en la Facultad de Informática de la Universidad de Murcia (aprobada en Junta de Facultad 27-11-2015)

DECLARO:

Que el Trabajo Fin de Grado presentado para su evaluación es original y de elaboración personal. Todas las fuentes utilizadas han sido debidamente citadas. Así mismo, declara que no incumple ningún contrato de confidencialidad, ni viola ningún derecho de propiedad intelectual e industrial

Murcia, a 23 de enero de 2025



Fdo.: **Maxim Utica Babyak**

*Autor del TFG*

## Resumen

En los últimos años, el Aprendizaje Federado Descentralizado (DFL) ha ganado relevancia por su capacidad de preservar la privacidad en sistemas distribuidos sin un servidor central. Sin embargo, enfrenta retos como la heterogeneidad de los datos y la convergencia de los modelos. El Modelo Probabilístico de Difusión para la Eliminación de Ruido (DDPM), modelo de difusión avanzado, ofrece soluciones innovadoras al manejar datos complejos y ruidosos, posicionándose como herramientas clave para superar estos desafíos y mejorar el rendimiento en entornos DFL.

El presente trabajo tiene como objetivo principal proponer una solución modular para simular diversas topologías de DFL, desde configuraciones completamente conectadas hasta estructuras en estrella, utilizando librerías que optimizan la gestión de modelos generativos en entornos distribuidos. Una innovación destacada es la estrategia de compartición de etiquetas, que enriquece los datos en los nodos, mitigando problemas de distribuciones no-IID y mejorando la convergencia y el rendimiento de los modelos locales. El algoritmo descentralizado sigue un flujo iterativo donde los nodos entrenan localmente, comparten parámetros con sus vecinos y agregan pesos para construir un modelo global robusto. Este enfoque descentralizado elimina la dependencia de un servidor central, garantizando escalabilidad, resiliencia y aplicabilidad en escenarios con recursos heterogéneos.

Los resultados experimentales obtenidos con datasets como MNIST, FashionMNIST y EMNIST Letters respaldan la eficacia del enfoque propuesto. Las simulaciones revelan mejoras significativas en precisión y representatividad, destacando especialmente en el manejo de clases no asignadas a nodos específicos. En particular, se observa una mejora del 124 % en la precisión de una topología más compleja al aplicar la estrategia de compartición de etiquetas. Estos resultados reflejan la capacidad del sistema para abordar escenarios distribuidos desafiantes, garantizando una convergencia equilibrada y un rendimiento robusto de los modelos entrenados.

Finalmente, el trabajo no solo destaca la relevancia de los modelos de difusión en el contexto del DFL, sino que también identifica áreas clave para futuras investigaciones. Entre estas se incluyen la optimización del uso de recursos, la incorporación de medidas de seguridad más robustas y la exploración de arquitecturas descentralizadas más complejas. Estas mejoras tienen el potencial de extender significativamente la aplicabilidad de los DDPMs en sistemas distribuidos, abriendo nuevas oportunidades para la investigación y el desarrollo en el campo del DFL. Con esta base sólida, el trabajo establece un marco que puede servir como punto de partida para futuras innovaciones en este ámbito.

## Extended Abstract

Artificial Intelligence (AI) has become a key driver of technological and scientific innovation over the past decade, redefining how we interact with the world and process information. Among its most notable advancements is Generative AI (GAI), a branch capable of creating original content such as text, images, and audio using advanced deep learning models. This capability has opened up a wide range of applications, from creative content generation to data synthesis across various domains.

On the other hand, Federated Learning (FL) has emerged as an innovative solution for training AI models while preserving data privacy. This approach allows multiple devices or entities to collaborate in training a common model without directly sharing sensitive data. However, the traditional centralized approach to FL relies on a central server to coordinate the training process, which introduces vulnerabilities in terms of security, scalability, and single points of failure. In response to these limitations, Decentralized Federated Learning (DFL) has emerged as an alternative paradigm, eliminating the dependence on a central server and leveraging techniques such as distributed graphs and peer-to-peer networks for coordination.

In this context, Diffusion Models (DMs), which initially revolutionized the field of content generation through the simulation of stochastic processes, are being explored as promising tools in FL. These models have an intrinsic ability to efficiently process distributed data, making them ideal candidates for addressing the inherent challenges of DFL. Among them, the innovation of Denoising Diffusion Probabilistic Models (DDPMs) stands out. This model represents an advanced approach in which complex data distributions are modeled by simulating gradual processes of degradation (noise introduction) and recovery (noise removal). In DDPMs, learning occurs by training a model to reverse this noise process, enabling the generation of highly realistic and precise data. Its ability to handle complex and noisy data makes it an ideal tool for improving the quality and robustness of models in distributed systems.

In recent years, there have been significant attempts to integrate the capabilities of GAI into FL. These initiatives aim to harness the power of advanced generative models, such as DMs, to synthesize data, enhance the robustness of trained models, and optimize communication among federated nodes. However, these implementations have been marked by technical challenges, such as high computational costs, protection of synthetic data, and interoperability within distributed networks.

In this context, a promising strategy for improving the convergence of local models in decentralized scenarios is the sharing of labels among nodes. This strategy enriches the datasets of nodes with samples of classes that were not originally assigned to them, selecting a subset limited by a predefined threshold. By allowing each node to have a basic reference for generating samples of unassigned classes, this technique enhances data

representation and facilitates convergence towards a more robust and balanced global model.

Despite these advancements, the need to decentralize FL remains a critical issue. The centralized approach offers clear advantages, such as simpler coordination and shorter convergence times. However, it also presents significant risks related to privacy, vulnerability to attacks, and dependency on centralized infrastructure. Consequently, various attempts to decentralize FL have demonstrated innovative approaches to distributing workloads and improving resilience while maintaining performance and security standards.

This work explores how DMs and the capabilities of GAI can be effectively integrated into the DFL framework, analyzing the challenge of convergence in a context of increasing demand for privacy and scalability in AI systems.

Motivated by the absence of a general-purpose library for DFL, the present work focuses on the design and evaluation of different techniques to simulate DFL scenarios and deeply analyze the application of DMs, specifically DDPMs. The application enables the simulation of various decentralized topologies, ranging from fully connected configurations to star topologies. However, regarding communication between nodes, the implementation does not delve into protective mechanisms such as cryptographic methods, as the primary focus is on evaluating the performance of DFL scenarios.

The implementation focuses on creating and simulating scenarios for DFL using DDPMs. The development relies on the *Diffusers* library for managing generative model pipelines and *PyTorch* for neural network training and data preprocessing.

Each node in the network operates autonomously, performing tasks such as local training of models and parameter aggregation from neighboring nodes. A key component of the training process is the U-Net architecture, a convolutional neural network originally developed for biomedical image segmentation. In this work, the U-Net has been adapted to effectively handle grayscale data, making it particularly suitable for tasks requiring both high-level contextual understanding and the preservation of fine-grained details. Its robustness in balancing these requirements makes it ideal for image synthesis in the DDPM.

The U-Net architecture implemented in this system comprises three encoding layers and three decoding layers. The encoding layers extract features from the input data while reducing its spatial resolution and increasing the depth of feature maps, enabling the capture of abstract and high-level information. These operations involve successive convolutions and downsampling steps, such as max-pooling. The first encoding layer expands the single-channel grayscale input to 64 feature maps, capturing low-level features, while subsequent layers increase the depth to 128 feature maps, representing more complex patterns.

The decoding layers aim to reconstruct the input's original spatial resolution while

preserving contextual information. This is achieved using upsampling operations, such as transposed convolutions, which progressively restore the spatial dimensions of the feature maps. Skip connections link each encoding layer to its corresponding decoding layer, ensuring that fine-grained details lost during downsampling are retained and integrated into the reconstruction process. These connections enable the model to combine local and global features effectively. The final decoding layer produces an output that matches the input’s spatial dimensions while embedding meaningful representations learned during training.

To further enhance the U-Net’s performance for diffusion-based tasks, attention mechanisms are incorporated into the deeper layers of the encoder and decoder. These mechanisms allow the model to focus on spatial regions of importance, prioritizing features critical for generating realistic and accurate samples. This adaptation is particularly valuable in decentralized learning scenarios, where data distributions can vary significantly across nodes. The integration of these design elements ensures that the U-Net achieves robust local training, enabling the generation of high-quality outputs while maintaining scalability and adaptability to diverse datasets and topologies.

The DDPM framework utilizes the U-Net architecture as its foundational network to facilitate the learning of the reverse diffusion process. DDPM models are generative systems that simulate a forward and reverse process to reconstruct data. In the forward diffusion process, noise is gradually added to clean data over a series of timesteps, progressively degrading it until the data becomes indistinguishable noise. This process creates a mapping from data to noise, which is learned by the model.

The U-Net, as the backbone of the DDPM, takes noisy images as input, along with their associated timesteps and class labels. Its role is to predict the noise residuals at each timestep, which is essential for reconstructing clean images during the reverse diffusion process. The skip connections in the U-Net preserve fine-grained details throughout the denoising process, while attention mechanisms help focus on the most relevant spatial features in the data. This combination allows the DDPM to learn how to denoise images effectively, generating high-quality samples that closely resemble the original input.

During training, the loss function calculates the difference between the predicted noise and the actual noise added to the data. By minimizing this loss, the DDPM learns to model the reverse process accurately, step-by-step, from noisy data back to clean data. The integration of the U-Net ensures that the system captures both global and local patterns, enabling high-quality generation even in complex datasets.

The overall algorithm is designed to operate in a decentralized federated learning framework, where nodes perform local computations and share information with their neighbors. The workflow begins with the initialization of datasets and models at each node, setting the stage for local training. Each node trains its model locally for a specified number of

epochs, iterating through the training data to extract meaningful patterns and learn the forward and reverse diffusion processes.

During training, the images in the dataset are augmented by adding noise according to the forward diffusion process. This augmentation simulates the degradation of data and is critical for teaching the model how to reverse the noise. The U-Net processes these noisy images, predicting the noise residuals for the reverse diffusion. The loss function compares the predicted noise with the actual noise added during augmentation. This loss is minimized by adjusting the model’s parameters through backpropagation, ensuring that the model learns to denoise effectively. Gradients are clipped to maintain numerical stability, and the optimizer updates the parameters, with the learning rate dynamically adjusted to improve convergence.

After completing local training for the current iteration, nodes exchange their model parameters with neighboring nodes based on the specified topology. This communication allows nodes to aggregate the insights learned by their neighbors, updating their models collaboratively. The aggregation process ensures that each node improves by incorporating shared knowledge while maintaining decentralization.

At periodic intervals or upon completing the training process, the models are saved along with additional test samples generated to evaluate performance. Each model is converted into a reusable pipeline, and key metrics such as training progress and step counts are logged for future analysis.

This process repeats over multiple epochs, combining local training and inter-node communication to achieve gradual convergence. By leveraging the U-Net within the DDPM framework and implementing decentralized aggregation, the algorithm enables nodes to collaboratively build a global model without relying on a central server. The combination of noise augmentation, iterative learning, and decentralized communication ensures the system is robust, scalable, and effective for generating high-quality outputs.

To address the challenges of convergence, particularly in scenarios with non-IID data distributions, a label-sharing strategy is implemented. Nodes enrich their local datasets with a controlled subset of unassigned classes based on a fixed threshold. This approach enables nodes to gain a basic representation of unassigned classes, improving generalization and facilitating convergence toward a more balanced global model.

Experiments were designed to evaluate the performance of the implementation in terms of precision and convergence. The datasets employed include MNIST, FashionMNIST, and EMNIST Letters, selected for their varying levels of complexity and applicability in generative and classification tasks.

Two topologies, a 4-node network and a 10-node network, were tested to analyze the impact of network complexity on model performance. Precision was chosen as the primary

evaluation metric, measuring the proportion of generated images that match their intended labels. This was validated by comparing the generated labels with those classified by an external classifier.

The label-sharing strategy significantly improved model convergence, particularly in non-IID data scenarios, by enabling nodes to better represent unassigned classes and enhancing the training process. In the 4-node topology, the precision for the MNIST dataset improved by 2 %, reaching 98.15 %. Although this improvement is modest, it is expected due to the simplicity of the dataset. For FashionMNIST, the model achieved a precision of 87.19 %, and for EMNIST Letters, it reached 90.53 %.

In the 10-node topology, the improvements were more substantial. For MNIST, the precision increased by 106 %, reaching 97.95 %. For FashionMNIST, the precision improved by 69 %, achieving 87.91 %. Finally, for EMNIST Letters, the precision saw a remarkable improvement of 124 %, reaching 87 %. These results underline the effectiveness and robustness of the label-sharing strategy in facilitating model convergence and delivering balanced and realistic data generation across varying datasets and topologies. The results align with or exceed benchmarks reported in previous works, such as Phoenix, highlighting the effectiveness of integrating DDPMs and label-sharing strategies in DFL scenarios.

Performance metrics, such as precision across iterations, revealed that fully connected topologies achieved faster convergence due to more frequent communication between nodes. The experiments also demonstrated that the label-sharing strategy contributed significantly to balancing class representation across nodes. This balanced representation was inferred from improved convergence rates and higher precision metrics across datasets and topologies.

This work successfully demonstrates the integration of DDPMs within DFL frameworks. The implementation achieves a modular and adaptable design that supports diverse topologies and datasets, effectively addressing challenges related to data distribution and model convergence. By introducing a label-sharing strategy, the system significantly enhances the convergence and precision of trained models, particularly in non-IID scenarios, while achieving competitive or superior performance compared to existing literature.

The decentralized nature of the approach eliminates the dependency on a central server, improving the system’s resilience and privacy. DDPMs have proven to be highly effective in generating realistic data in distributed environments, contributing to robust global models. However, the absence of cryptographic methods and security measures highlights a potential vulnerability in real-world applications, while highly distributed topologies could face challenges in achieving efficient convergence.

Future work should focus on exploring the inherent limitations of the proposed solution’s architecture to identify potential areas for improvement. Additionally, implementing fine-

tuning mechanisms would enhance the quality of training and allow the system to support a broader range of datasets. Another important avenue is the development of a role-assignment mechanism, enabling the differentiation of nodes into trainers, aggregators, and proxies, thereby optimizing the functionality and coordination of the network.

Efforts should also aim to reduce the frequency of inter-node communication to specific intervals, which would alleviate bandwidth stress and improve the efficiency of resource usage. Furthermore, the use of alternative evaluation metrics beyond precision could provide a more comprehensive assessment of the quality of generated images. Finally, privacy issues in DFL scenarios require further investigation to address concerns related to data security and ensure compliance with privacy standards. These advancements would collectively strengthen the robustness and applicability of decentralized federated learning systems.

In conclusion, the proposed framework provides a strong foundation for advancing decentralized federated learning systems by addressing key challenges in data representation and convergence. With further refinements, this approach has the potential to make significant contributions to distributed AI research and applications.



## 1. Introducción

La Inteligencia Artificial (IA) se ha convertido en un motor clave para la innovación tecnológica y científica en la última década, redefiniendo la forma en que interactuamos con el mundo y procesamos información. Entre sus avances más notables se encuentra la Inteligencia Artificial Generativa (GAI), una rama capaz de crear contenido original como texto, imágenes y audio, utilizando modelos avanzados de aprendizaje profundo. Esta capacidad ha abierto un abanico de aplicaciones, desde la generación de contenido creativo hasta la síntesis de datos en diversos dominios.

Por otro lado, el Aprendizaje Federado (FL) [1] ha emergido como una solución innovadora para entrenar modelos de IA preservando la privacidad de los datos. Este enfoque permite que múltiples dispositivos o entidades colaboren en el entrenamiento de un modelo común sin compartir directamente los datos sensibles. Sin embargo, el enfoque centralizado tradicional del FL depende de un servidor central para coordinar el entrenamiento, lo que introduce vulnerabilidades en términos de seguridad, escalabilidad y posibles puntos únicos de falla. En respuesta a estas limitaciones, el DFL ha surgido como un paradigma alternativo, eliminando la dependencia de un servidor central y utilizando técnicas como grafos distribuidos y redes peer-to-peer para lograr la coordinación.

Dentro de este contexto, los Modelo de Difusión (DM), que inicialmente revolucionaron el campo de la generación de contenido mediante la simulación de procesos estocásticos, están siendo explorados como herramientas prometedoras en el FL. Estos modelos poseen una capacidad intrínseca para procesar datos distribuidos de manera eficiente, lo que los convierte en candidatos ideales para abordar los retos inherentes del DFL. Entre estos, destaca la innovación del DDPM [2]. Este modelo representa un enfoque avanzado en el que las distribuciones complejas de datos se modelan al simular procesos graduales de degradación (introducción de ruido) y recuperación (eliminación del ruido). En DDPM, el aprendizaje ocurre al entrenar un modelo para revertir este proceso de ruido, permitiendo la generación de datos altamente realistas y precisos. Su capacidad para manejar datos complejos y ruidosos lo convierte en una herramienta ideal para mejorar la calidad y la robustez de los modelos en sistemas distribuidos.

En los últimos años, ha habido intentos significativos de integrar las capacidades de la GAI en el FL. Estas iniciativas buscan aprovechar la potencia de modelos generativos avanzados, como los DM, para sintetizar datos, mejorar la robustez de los modelos entrenados y optimizar la comunicación entre nodos federados. Sin embargo, estas implementaciones han estado marcadas por desafíos técnicos, como el alto costo computacional, la protección de datos sintéticos y la interoperabilidad en redes distribuidas.

En este contexto, una estrategia prometedora para mejorar la convergencia de los modelos locales en escenarios descentralizados es la compartición de etiquetas entre nodos. Esta estrategia consiste en enriquecer los conjuntos de datos de los nodos con muestras de

clases no asignadas originalmente, seleccionando un subconjunto limitado por un umbral predefinido. Al permitir que cada nodo tenga una referencia básica sobre cómo generar muestras de clases no asignadas, esta técnica mejora la representación de datos y facilita la convergencia hacia un modelo global más robusto y equilibrado.

A pesar de los avances, la necesidad de descentralizar el FL sigue siendo un tema crucial. El enfoque centralizado ofrece ventajas evidentes, como una coordinación más sencilla y un menor tiempo de convergencia. Sin embargo, también presenta riesgos significativos relacionados con la privacidad, la vulnerabilidad ante ataques y la dependencia en una infraestructura centralizada. Por ello, diversos intentos de descentralizar el FL han puesto de manifiesto enfoques innovadores para distribuir la carga de trabajo y mejorar la resiliencia, manteniendo al mismo tiempo los estándares de rendimiento y seguridad.

Este trabajo explora la integración de DMs y las capacidades de la GAI en el marco del DFL, con un enfoque en abordar los desafíos de convergencia en un contexto de creciente demanda de privacidad y escalabilidad en sistemas de IA. Ante la ausencia de una librería de propósito general para DFL, se diseña y evalúa una solución que simula diversas topologías descentralizadas, desde configuraciones completamente conectadas hasta estructuras en estrella. Aunque no se profundiza en la comunicación entre nodos ni en mecanismos criptográficos, el trabajo se centra en analizar la aplicación de DMs, especialmente los DDPMs, y su impacto en el rendimiento de los escenarios descentralizados.

La aplicación está diseñada para desplegar escenarios de DFL, utilizando DDPMs para el entrenamiento de los modelos. En este contexto, los conjuntos de datos utilizados se limitan a MNIST, FashionMNIST y EMNIST. Además, se implementó la estrategia de compartición de etiquetas previamente descrita, con el objetivo de mejorar la convergencia de los modelos entrenados en los nodos.

Para validar y evaluar los escenarios, se generan imágenes asociadas a las etiquetas entrenadas por los modelos en cada configuración. La precisión de los modelos se evalúa comparando las etiquetas generadas con las clasificadas mediante un clasificador, lo que permite medir el rendimiento de los nodos en la generación de muestras realistas.

Finalmente, se realizan experimentos utilizando dos topologías distintas y los conjuntos de datos mencionados. En particular, en una topología más distribuida, los experimentos con FashionMNIST y EMNIST Letters arrojan precisiones de 91.79 % y 90.49 %, respectivamente, destacando la efectividad de la implementación en estos contextos.

La estructura del trabajo se organiza de la siguiente manera: en la Sección 2 se presenta el contexto teórico del FL, abordando la motivación, los fundamentos del DFL y la GAI, con énfasis en los enfoques tradicionales y los DMs. En la Sección 3 se describe el estado del arte sobre los DMs, las implementaciones de DFL, la integración de GAI en FL y la aplicación de DMs en FL. La Sección 4 detalla los objetivos del trabajo y la metodología

seguida. En la Sección 5 se analizan las decisiones de diseño e implementación de la aplicación. En la Sección 6 se presentan los resultados obtenidos y se comparan con los hallazgos reportados en la literatura, evaluando el desempeño de los DMs en escenarios de DFL. Finalmente, en la Sección 7, se resumen las conclusiones extraídas del trabajo realizado y se discuten posibles líneas futuras de investigación.

## 2. Background

Esta sección introduce conceptos relacionados con la técnica de FL centrándose en el enfoque de DFL, pudiendo entender así sus objetivos y funcionamiento. Además, se incluyen definiciones de conceptos enlazados a la GAI, con el objetivo de introducir los modelos de difusión y sus ventajas.

### 2.1. Aprendizaje federado

El FL se define como una técnica novedosa en el campo del aprendizaje automático. Su implementación permite entrenar modelos de aprendizaje automático, ya sea en múltiples dispositivos o en servidores distribuidos, sin la necesidad de enviar datos de entrenamiento a un servidor central.

Respecto al modelo tradicional de entrenamiento, el proceso de entrenamiento toma lugar en un servidor central con acceso a todos los datos de entrenamiento, los cuales se recopilan y almacenan en un solo lugar. Este modelo plantea riesgos de privacidad, puesto que todos los datos son accesibles desde un único punto.

En comparación, el modelo del FL ofrece una mayor privacidad y seguridad al mantener los datos en los dispositivos locales. A su vez, teniendo los datos en un ámbito local, el proceso de entrenamiento tiene lugar en cada uno de los dispositivos, resultando en modelos locales que acaban siendo combinados para formar un modelo global.

#### 2.1.1. Motivación del DFL

Cuando se hizo la primera propuesta significativa de FL [1] se planteó un enfoque más centralizado en su diseño, el Aprendizaje Federado Centralizado (CFL). En este diseño, los dispositivos locales, tras terminar de entrenar, envían los parámetros actualizados a un servidor central para su agregación y consecuente actualización del modelo global.

Sin embargo, este enfoque sigue teniendo limitaciones, como la dependencia de un servidor central para funcionar. De esta manera, los principales desafíos se caracterizan por la comunicación entre los nodos y el servidor. En la transferencia de parámetros, el modelo global es susceptible a ataques que introduzcan sesgos o cualquier otro tipo de perturbación que afecte a su calidad e integridad, y a su vez, todos los parámetros transferidos pueden verse comprometidos, planteando serios riesgos de privacidad y seguridad. Otra limitación, dada la ingente comunicación entre el servidor y los nodos, son los costes en términos de ancho de banda y consumo de energía, pudiendo provocar serios problemas de congestión o indisponibilidad en el servidor.

En términos generales, la técnica de FL puede satisfacer las necesidades de ámbitos sensibles, desde la personalización de tratamientos o diagnósticos en el ámbito de la sanidad,

hasta la detección de actividades fraudulentas en el ámbito financiero, sin arriesgar la privacidad de los datos. Sin embargo, las limitaciones del enfoque centralizado no permiten un desempeño eficaz de la técnica. En todos estos ámbitos es necesario tener un modelo de aprendizaje seguro, eficiente y flexible. Con esta premisa, se motivó el desarrollo e implementación del enfoque descentralizado.

### **2.1.2. Fundamentos del DFL**

El enfoque descentralizado de la técnica de FL (DFL) permite el entrenamiento de modelos de aprendizaje automático en un escenario distribuido sin la necesidad de un servidor central. En su lugar, múltiples nodos colaboran directamente entre sí para entrenar un modelo global, reduciendo el riesgo de exposición de datos y asegurando así una comunicación eficiente y escalable.

Otra de las ventajas del enfoque descentralizado es la tolerancia a fallos, pues un escenario descentralizado se caracteriza por la capacidad de manejar la pérdida de dispositivos locales sin afectar la integridad del modelo global.

Respecto al proceso de actuación de los nodos participantes, estos siguen un esquema iterativo hasta converger en un modelo final:

1. Entrenamiento del modelo local.
2. Intercambio de parámetros con los nodos vecinos.
3. Agregación de los parámetros recibidos al modelo local.

Este esquema es seguido por los nodos con el rol de entrenador, pero requiere la participación de nodos con el rol de agregador. Los nodos agregadores recopilan los parámetros enviados por los nodos vecinos, los difunden a otros nodos y los integran en un modelo global. En topologías donde los nodos agregadores no son accesibles directamente, se introducen nodos proxy. Estos actúan como intermediarios, retransmitiendo los parámetros recibidos a otros nodos vecinos para mantener la conectividad de la red.

La comunicación entre nodos puede implementarse de tres formas principales:

1. Síncrona: Todos los nodos deben estar activos y disponibles simultáneamente para la comunicación en tiempo real. Este enfoque permite actualizaciones de modelos en momentos predefinidos, pero incrementa significativamente el costo de recursos debido a la necesidad de sincronización estricta.
2. Asíncrona: No requiere que los nodos estén activos simultáneamente. Los nodos actualizan sus modelos en su propio tiempo, proporcionando mayor flexibilidad y una mejor eficiencia en el uso de recursos.

3. Semisíncrona: Combina características de los enfoques síncrono y asíncrono. Permite actualizaciones periódicas de los modelos, ofreciendo un balance entre sincronización y flexibilidad.

Estos esquemas permiten adaptarse a diferentes topologías y requerimientos del sistema, optimizando el rendimiento y la eficiencia del DFL.

## 2.2. Inteligencia artificial generativa

La GAI es uno de los diversos campos dentro del amplio espectro de la IA. Este campo se centra en el desarrollo de sistemas capaces de generar contenido original, como imágenes, música, texto, audio y vídeos. A diferencia de enfoques tradicionales de la IA, que se limitaban a resolver tareas específicas y predefinidas, como el reconocimiento de imágenes, voz o la traducción de idiomas, la GAI adopta un enfoque más creativo. Su funcionamiento se basa en aprender patrones y características presentes en los conjuntos de datos, utilizando este conocimiento para generar nuevos datos que se ajusten a dichos patrones.

Este campo de la IA tiene una amplia gama de aplicaciones prácticas. Entre ellas, se encuentra la creación de arte digital, la composición de canciones en diversos estilos y géneros, la generación de historias escritas, e incluso la producción de videos con cierto nivel de coherencia narrativa. Estas capacidades destacan el potencial creativo y versátil de la GAI en distintos ámbitos.

### 2.2.1. Enfoques tradicionales

El campo de la GAI presenta distintos enfoques para generar contenido. Entre las principales técnicas se encuentran el Codificador Automático Variacional (VAE) [3], la Red Generativa Adversativa (GAN) [4], la Red Neuronal Recurrente (RNN) [5], el Modelo Autorregresivo [6] y el Modelo basado en Transformadores [7].

Los VAEs se entrenan para aprender la distribución de probabilidad subyacente de los datos y generar así muestras nuevas de la distribución aprendida. Este tipo de modelo generativo consta de un codificador que toma datos de entrada y los mapea en un espacio de variables latentes, y de un decodificador que toma muestras de dicho espacio y las reconstruye en el espacio original de los datos. Este sistema es eficaz en las tareas de clasificación y reconstrucción, aunque el proceso de entrenamiento puede ser complejo debido a la necesidad de optimizar tanto el codificador como el decodificador, resultando en dificultades para generar datos de altas dimensiones debido a la necesidad de modelar distribuciones complejas.

Las GANs son sistemas compuestos por dos redes neuronales: un generador y un discriminador, que trabajan en oposición. Durante el entrenamiento, el generador crea muestras

falsas intentando que sean indistinguibles de las reales en el conjunto de datos, mientras que el discriminador busca diferenciar las muestras reales de las generadas. Este proceso competitivo permite que el generador mejore progresivamente su capacidad para producir muestras realistas, logrando resultados de alta calidad. Por esta razón, las GAN se emplean comúnmente para generar imágenes, aunque también pueden aplicarse a la generación de texto o música. A pesar de su eficacia en la creación de datos de alta dimensionalidad y complejidad, las GANs presentan desafíos como la inestabilidad durante el entrenamiento y la incapacidad para evaluar la incertidumbre en las muestras generadas.

Las RNNs están diseñadas para modelar secuencias de datos, aprovechando su capacidad para procesar entradas de manera secuencial. En cada paso de tiempo, la salida se genera en función de la entrada actual y del historial de entradas previas, gracias a sus conexiones de retroalimentación, lo que les permite capturar dependencias a largo plazo en los datos. Estas características hacen que las RNNs sean ampliamente utilizadas en tareas como la predicción de series temporales, la traducción automática y la generación de texto. Sin embargo, su naturaleza secuencial presenta limitaciones, ya que puede resultar en una ineficiencia computacional que restringe su escalabilidad en secuencias muy largas. Además, en estas situaciones, es más probable que surjan problemas asociados al desvanecimiento o explosión del gradiente, complicando el proceso de entrenamiento.

Los modelos autorregresivos tienen un uso parecido a las RNNs, usándose también para la predicción de series temporales y generación de texto. Consisten en un modelo que genera una secuencia de salida paso a paso, donde cada elemento de la secuencia depende de los elementos anteriores. Estos modelos son conceptualmente simples y fáciles de implementar, con una gran capacidad para modelar distribuciones de probabilidad condicionales complejas. Aunque dichos modelos presentan problemas para capturar dependencias a largo plazo en secuencias, debido a su naturaleza autorregresiva, y como su generación es secuencial, los tiempos de inferencia pueden resultar lentos.

Los modelos basados en transformadores son útiles para el procesamiento del lenguaje natural debido a la capacidad inherente de capturar dependencias a largo plazo en secuencias de texto. Su arquitectura basada en una red neuronal difiere de las RNNs, puesto que los transformadores no tienen una estructura recurrente y son capaces de procesar todas las posiciones de la secuencia simultáneamente usando mecanismos de atención, presentando una mayor eficiencia computacional. No obstante, se requieren cantidades ingentes de datos y potencia computacional para entrenar modelos de alta calidad.

### **2.2.2. Modelos de difusión**

El DM es un modelo probabilístico formulado como una cadena de Márkov y entrenado mediante inferencia variacional. Este enfoque introduce ruido progresivamente en los datos para generar muestras. En particular, se utiliza el modelo probabilístico de difusión

diseñado para la eliminación de ruido. Cada transición en la cadena de Márkov aprende a revertir el proceso de difusión, añadiendo de manera gradual ruido gaussiano a los datos en sentido opuesto al muestreo, hasta descomponer completamente la señal original.

Los modelos de difusión tienen una gran capacidad de generar muestras de alta calidad, siendo que son fáciles de definir y eficientes de entrenar. La eficiencia del entrenamiento se define por el uso del descenso de gradiente estocástico, minimizando la función de pérdida.

Debido a sus mejorados resultados en la generación de imágenes y audio, estos modelos compiten directamente con las GANs, pudiendo igualar la calidad de la muestra, y al mismo tiempo, lograr una cobertura de modalidades mucho mejor [8].



### 3. Estado del Arte

Esta sección revisa los trabajos más relevantes sobre FL y GAI en la literatura, enfocándose en los objetivos propuestos y escenarios implementados.

#### 3.1. Modelos de difusión

El estudio de Zhang et al. [9] evalúa métodos de difusión en la generación de imágenes utilizando el conjunto de datos MS-COCO y la métrica Distancia de Inicio de Fréchet (FID), que mide la calidad y diversidad de las imágenes, donde valores más bajos indican mayor similitud con las reales. Los modelos destacados por su alta calidad y variabilidad son Imagen [10], eDiff-I [11] y ERNIE-ViLG2.0 [12].

Imagen es un modelo de difusión que implementa un Modelo de Lenguaje Grande (LLM) preentrenado como codificador de texto, mejorando su FID score al aumentar el tamaño del LLM.

El modelo eDiff-I, en cambio, es un conjunto de modelos de difusión especializados que se reparten en distintas etapas específicas del proceso de generación iterativo para entrenarse. El modelo empieza intentando producir contenido coherente con la entrada de texto, mientras que al final del proceso, acaba ignorándola para producir salidas de alta calidad.

ERNIE-ViLG2.0 funciona de manera similar a eDiff-I, incorporando distintos modelos especializados en distintas etapas del proceso, con la diferencia de que incorpora conocimiento adicional sobre la escena visual. En concreto, emplea un analizador de texto y un detector de objetos para extraer los elementos clave de la escena en el par texto-imagen de entrada, guiando a continuación para que preste más atención a su alineación en el proceso de aprendizaje, con la idea de que el modelo pueda manejar las relaciones entre varios objetos y atributos.

#### 3.2. Implementaciones de DFL

El trabajo de Lalitha et al. [13] cimienta las bases del DFL, implementando un algoritmo de aprendizaje distribuido que implica a todos los nodos de la red. Dicho algoritmo está adaptado para entrenar con una Red Neuronal Profunda (DNN) debido a que es intratable el cálculo exacto de las constantes de normalización. Si bien este trabajo presenta una sólida base para DFL, no ahonda en las restricciones inherentes del escenario.

El trabajo de Roy et al. [14] presenta *BrainTorrent*, una implementación práctica de DFL en el ámbito médico para tareas de segmentación *whole-brain*. Este enfoque descentralizado supera en rendimiento a los métodos centralizados, demostrando mejoras signi-

ficativas en privacidad, eficiencia y precisión, destacando su potencial para aplicaciones médicas sensibles.

En el ámbito de la red, el trabajo de Hu et al. [15] se centra en el uso eficiente de la capacidad de red entre nodos, mejorando el ancho de banda en las topologías descentralizadas. El planteamiento consta de dos partes. La primera parte trata de dividir el modelo en un conjunto de segmentaciones, que se actualizan agregando una segmentación local con la segmentación correspondiente de otros nodos. La segunda parte implementa un funcionamiento parecido al protocolo *gossip* para la transferencia del segmento de modelo en cada iteración de entrenamiento de un nodo a otro, cumpliendo con el esquema de federación descentralizada. Los resultados del trabajo muestran una reducción del tiempo de entrenamiento y del ancho de banda.

En el marco de la seguridad del aprendizaje federado, el equipo de Li et al. [16] propone un escenario de DFL basado en *blockchain*, *BFLC*. La arquitectura basada en la cadena de alianzas garantiza el control de permisos de los nodos FL. Para el acceso al último modelo, se diseña un patrón de almacenamiento en la cadena de modelos y actualizaciones, y respecto a la validación de actualizaciones, se opta por un mecanismo de consenso de bloques que logra estabilidad frente a ataques maliciosos.

### 3.3. Aplicación de GAI en FL

La propuesta de Cao et al. [17], *PerfFED*, introduce un enfoque de FL basado en GANs convolucionales que permite a los clientes entrenar sus modelos de manera independiente sin compartir arquitecturas ni parámetros. Este diseño protege la privacidad de los datos de entrenamiento, evitando posibles inferencias sensibles. *PerfFED* destaca como un avance significativo en privacidad, flexibilidad y autonomía dentro del aprendizaje federado.

La preocupación del equipo de Li et al. [18] por la consistencia de datos usados, les llevó a presentar una solución respecto al desafío de datos no-Independiente e Idénticamente Distribuido (IID) en FL, Aprendizaje Federado Asistido por Datos Sintéticos (SDA-FL). Esta propuesta resuelve el problema de los datos no-IID compartiendo datos sintetizados diferencialmente privados. Cada cliente preentrena una GAN convolucional local diferencialmente privada para generar datos sintéticos, evitando así compartir datos originales, para después subirlos a un servidor de parámetros que construye un conjunto de datos sintéticos global compartido.

El equipo de Li et al. [19] desarrolla Llenando lo que Falta (FIMI), un método de augmentación de datos adaptativo que mitiga la heterogeneidad en datos y recursos en FL. FIMI utiliza modelos generativos preentrenados para sintetizar datos personalizados por nodo, asegurando un entrenamiento eficiente al analizar la relación entre cantidad de datos y error de aprendizaje. Además, optimiza el consumo energético minimizándolo

mientras se cumplen los objetivos de rendimiento. Los resultados muestran una reducción de hasta un 50 % en el consumo energético sin comprometer la precisión, destacando a FIMI como un método escalable y sostenible para entornos federados.

El equipo de Nguyen et al. [20] presenta un esquema de DFL llamado *FedGAN* que genera imágenes de COVID-19 realistas para facilitar la detección de COVID-19 con mayor privacidad, mediante el uso de GANs. El escenario propuesto se sirve de una *blockchain* para sustituir la figura del servidor central y preservar la seguridad de la topología en un contexto tan importante como es el de la sanidad.

La propuesta de Wang et al. [21] introduce la integración de GAI en escenarios de DFL mediante el uso de GANs en nodos locales, evitando compartir datos sensibles. Los parámetros del modelo se intercambian utilizando Sistema de Archivos Interplanetario (IPFS), garantizando una comunicación segura y descentralizada. Este enfoque supera en precisión al algoritmo *FedAvg* [22], posicionándose como una solución eficiente y robusta. Además, el trabajo establece una base sólida para futuras investigaciones, optimizando el rendimiento y promoviendo el uso efectivo de GAI en sistemas distribuidos.

### 3.4. Aplicación de DMs en CFL

Los límites asociados a la comunicación en redes inalámbricas de FL motivaron al equipo de Huang et al. [23] a idear una solución que redujera los costes de comunicación y latencia de entrenamiento en escenarios de FL tratando con limitaciones en el preentrenamiento, ajuste e inferencia de los modelos de GAI, específicamente usando como caso de estudio un DM. Para solventar estas limitaciones, se decide una cantidad de datos adecuada para cada cliente, los métodos de ajuste se seleccionan y ajustan para reducir el uso de VRAM de la GPU y hacer más eficiente el cálculo basado en la GPU, y se entrena una pequeña parte de los parámetros del modelo mientras se mantiene el resto estático.

El equipo de Sattarov et al. [24] propone la aplicación de un DDPM para la generación de datos tabulares en FL. Su propuesta, *FedTabDiff*, genera datos tabulares de tipo mixto de alta fidelidad sin acceso centralizado a los conjuntos de datos tabulares originales. Los resultados demuestran que esta solución produce datos sintéticos que mantienen una alta fidelidad, utilidad, cobertura y privacidad.

Para facilitar la fusión segura de datos heterogéneos de clientes, el equipo de Li et al. [25] propone *FedDiff*, un escenario de FL con aplicación multimodal y multicliente de modelos de difusión. Esta propuesta establece una configuración de extracción de características de modelo de difusión de doble rama, en la que los datos modales se introducen en ramas separadas de la codificación. El punto de esto es que los modelos de difusión impulsados por diferentes modalidades son intrínsecamente complementarios en términos de pasos potenciales de eliminación de ruido sobre los que se pueden construir conexiones

bilaterales.

El trabajo de Yang et al. [26] presenta *FedDISC*, un método de coentrenamiento semisupervisado que aborda las limitaciones del FL, como los costos de comunicación, la heterogeneidad de datos y la presión del entrenamiento local. *FedDISC* elimina el entrenamiento local y reduce la comunicación a una sola ronda, utilizando un codificador *CLIP* preentrenado para extraer características y generar prototipos de categorías en el servidor. Los prototipos se envían a los clientes, donde se generan pseudoetiquetas y representaciones específicas del dominio, enriquecidas con ruido según el método de *Stable Diffusion*. Estas representaciones y centroides se envían de vuelta al servidor para generar imágenes finales mediante eliminación de ruido progresiva. Este enfoque optimiza la eficiencia del sistema federado, aliviando la carga computacional de los nodos y estableciendo un estándar para la integración de difusión avanzada en FL.

La propuesta de Jothiraj and Mashhadi [27], *Phoenix*, introduce un método innovador para entrenar DDPMs en topologías de FL, donde los nodos entrenan localmente sus modelos y comparten los pesos con un servidor central. Una estrategia clave es la compartición de un subconjunto del conjunto de datos global entre los nodos, lo que mejora la homogeneidad de las distribuciones y facilita la convergencia hacia un modelo global robusto. Los resultados muestran que *Phoenix* supera en rendimiento a los enfoques estándar de modelos de difusión en FL, destacándose por su uso eficiente de DDPMs y la estrategia de compartición de datos. Dada la relevancia de este trabajo, su enfoque de DDPMs en FL ha sido tomado como una referencia central para el desarrollo del presente estudio.

### 3.5. Motivación

Los trabajos discutidos hasta ahora sientan una base sólida para la implementación de DFL y el uso de DMs, además de explorar la integración de GAI y, específicamente, de DMs en entornos de FL. Algunos de estos estudios también abordan las limitaciones inherentes al FL, como la heterogeneidad de los datos y los costos de comunicación. Sin embargo, ninguno de ellos ha investigado la aplicación de un DM en una topología puramente descentralizada como las que ofrece el DFL.

Los escenarios de DFL presentan desafíos únicos que van más allá de los enfrentados en FL centralizado. Dos de las limitaciones más críticas son la carga computacional que los nodos pueden soportar, debido a recursos limitados, y la capacidad para lograr la convergencia de los modelos locales en ausencia de un servidor central. Estas restricciones resaltan la importancia de analizar cómo los DMs pueden influir en la eficacia y la escalabilidad de estas arquitecturas descentralizadas.

En este contexto, estudiar el impacto de los DMs en escenarios de DFL se vuelve

crucial para entender su potencial y optimizar su implementación. Evaluar su capacidad para manejar las limitaciones de los nodos y mejorar la convergencia de los modelos locales permitirá avanzar en la utilización de arquitecturas de aprendizaje verdaderamente descentralizadas y resilientes, adaptadas a las demandas de sistemas modernos distribuidos. La Tabla 1 resume los avances en la literatura del FL y los modelos generativos.

Tabla 1: Estado del arte en aprendizaje federado y modelos generativos

Paper	Año	Objetivo	FL	Arquitectura	Dataset	Métricas	Resultados
Sattarov et al. [24]	2024	Aplicación de modelos de difusión para la generación de datos tabulares en aprendizaje federado	CFL	DDPMs	Philadelphia City Payments Data	Fidelidad	0.59
						Utilidad	0.837
						Cobertura	0.944
						Privacidad	2.607
Jothiraj and Mashhadi [27]	2023	Aplicación de modelos de difusión en aprendizaje federado	CFL	DDPMs	CIFAR-10	IS	6.5758
						FID	0.0763
						Precisión	83.03 %
						Recall	70.06 %
Yang et al. [26]	2023	Reducción de costes de comunicación y eliminación del entrenamiento local en aprendizaje federado semi-supervisado	CFL	DMs preentrenados	DomainNet	Rendimiento	53.68
					OpenImage		59.42
					NICO++		C/U 84.17/91.03
Li et al. [25]	2023	Aplicación multimodal y multicliente de modelos de difusión en aprendizaje federado	CFL	CNNs de difusión	Trento	OA	99.38 %
						AA	98.79 %
						Kappa	99.18 %
Huang et al. [23]	2023	Reducción de costes de comunicación y latencia de entrenamiento en aprendizaje federado	CFL	Redes neuronales de difusión	Personalizado	Tiempo de convergencia	100 rondas
						Pérdida de entrenamiento	0.02
						Coste de comunicación	1320 MB/9.6 min

Tabla 1: Estado del arte en aprendizaje federado y modelos generativos

Paper	Año	Objetivo	FL	Arquitectura	Dataset	Métricas	Resultados
Li et al. [19]	2023	Mitigación de la heterogeneidad de los datos y optimización del uso de los recursos	CFL	Modelos generativos preentrenados	GTSRB	Energía	1319.82 J
						Latencia	0.63 horas
						Uplink	11.12 GB
						Precisión	93.8 %
Zhang et al. [9]	2023	Estudio sobre la aplicación de modelos de difusión	-	DMs	MS-COCO	FID	Imagen [10] 7.27
							eDiff-I [11] 6.95
							ERNIE-ViLG2.0 [12] 6.75
Li et al. [18]	2022	Aumento de la consistencia de los datos usados	CFL	GANs convolucionales	FashionMNIST	FID	217.81
						Precisión	83.76 %
					CIFAR-10	FID	129.25
						Precisión	84.56 %
Cao et al. [17]	2022	Aplicación de GANs en aprendizaje federado	CFL	GANs convolucionales	CIFAR-10	Precisión relativa	35.41 %
					CIFAR-100		49.4 %
					FEMNIST		42.63 %
Wang et al. [21]	2021	Aplicación de GANs en aprendizaje federado descentralizado con topología estrellada	DFL	GANs	CIFAR-10	Precisión	85.14 %
					CIFAR-100		55.28 %
Nguyen et al. [20]	2021	Generación de imágenes de COVID-19 protegidas	DFL	GANs convolucionales	COVID-19 x-ray	Precisión	99.3 %
						Sensibilidad	97.8 %
						F1	99.1 %
Li et al. [16]	2020	Aumento de la privacidad de los datos mediante la técnica de aprendizaje federado descentralizado en un escenario basado en blockchain	DFL	Aprendizaje basado en blockchain	FEMNIST	Precisión	90.02 %
Hu et al. [15]	2019	Reducción del ancho de banda	DFL	CNNs	CIFAR-10	Tiempo de convergencia	6-10 minutos
						Precisión	80 %

Tabla 1: Estado del arte en aprendizaje federado y modelos generativos

Paper	Año	Objetivo	FL	Arquitectura	Dataset	Métricas	Resultados
Roy et al. [14]	2019	Robustez ante interrupciones mediante la técnica de aprendizaje federado descentralizado	DFL	CNNs	MALC	Rendimiento	0.864
Lalitha et al. [13]	2018	Aplicación del aprendizaje federado descentralizado	DFL	DNNs	-	MSE	1.4
Konečný et al. [1]	2017	Mejora de la eficiencia de la comunicación	CFL	CNNs	CIFAR-10	Tiempo de convergencia	2000 rondas
						Precisión	90 %

## 4. Objetivos y Metodología

Esta sección detalla los objetivos que el trabajo busca alcanzar y la metodología seguida para llegar a tales objetivos.

### 4.1. Objetivos

El objetivo principal de este trabajo es el diseño, la implementación y la evaluación de los modelos de difusión en escenarios de DFL.

Para alcanzar dicho objetivo, se presenta la siguiente lista de subobjetivos:

- Estudiar el estado del arte para analizar los trabajos más relevantes para la implementación de un entorno que entrene modelos de difusión siguiendo las bases del DFL.
- Diseñar un entorno que gestione las distintas fases del entrenamiento de un modelo en una topología descentralizada de FL.
- Implementar el entorno modularmente, permitiendo la personalización de los escenarios.
- Evaluar el rendimiento y la eficacia del entorno ejecutando distintas pruebas basadas en la topología y conjunto de datos usado.

### 4.2. Metodología

A continuación, se describe la metodología seguida a lo largo del trabajo para alcanzar los objetivos previamente introducidos.

#### 4.2.1. Estudio del estado del arte

Para el desarrollo de este trabajo, se buscó una serie de trabajos relacionados con el uso de DMs, implementaciones de DFL, aplicaciones de GAI en FL, y aplicaciones de DMs en CFL. Era importante centrarse en aquellos trabajos que implementaran soluciones de DFL con GAI. Era por esto que el trabajo de Wang et al. [21] era fundamental en la implementación propuesta de este trabajo. Sin embargo, uno de los problemas recurrentes en topologías de DFL con datos no IID es la convergencia de los modelos. Para representar escenarios con datos no IID se escogió como sesgo la distribución de clases, esto es, cada nodo toma un conjunto de clases distinto al resto. Para una solución respecto a la convergencia de modelos en DFL, se optó por usar una estrategia de compartición de etiquetas similar a la compartición de datos propuesta por Jothiraj and Mashhadi [27]. Debido a esta similitud, se utilizarán los resultados obtenidos a partir de la solución propuesta en dicho trabajo como base para comparar nuestro enfoque con la literatura existente.



#### 4.2.2. Estrategia de compartición de etiquetas

Con el objetivo de mejorar la convergencia de los modelos locales en el escenario, se implementó una estrategia de compartición de etiquetas entre los nodos. Esta estrategia consiste en enriquecer el conjunto de datos de un nodo con muestras de clases o etiquetas que originalmente no le fueron asignadas.

El proceso se lleva a cabo seleccionando un subconjunto de datos de las clases no asignadas al nodo, siguiendo un umbral predefinido especificado al iniciar el escenario. Este umbral, que permanece constante para todos los nodos en la topología, limita la cantidad de datos añadidos de cada etiqueta no asignada. Así, se asegura de que la cantidad de datos adicionales no supere el valor establecido por el umbral, manteniendo un balance entre nodos.

Esta estrategia permite que cada nodo tenga una referencia básica sobre cómo deberían generarse muestras de las clases no asignadas, mejorando la representación general de los datos en el sistema y facilitando la convergencia hacia un modelo global más coherente y equilibrado.

#### 4.2.3. Diseño del entorno

El diseño del entorno comenzó con la implementación de un sistema básico que entrenaba un único DDPM utilizando el conjunto de datos MNIST, con el objetivo de sentar las bases para una posterior escalabilidad. Una vez establecido este entorno inicial, se amplió para incluir un escenario de CFL con varios nodos en la topología. En este escenario, cada nodo entrenaba su modelo de manera local y enviaba los parámetros al nodo central, que los agregaba para formar un modelo global y luego lo distribuía de vuelta a los nodos.

Finalmente, se descentralizó el entorno, eliminando la figura del nodo central. En esta nueva configuración, cada nodo se encargaba de agregar los parámetros de sus nodos vecinos directamente a su modelo local, lo que permitía una mayor autonomía en la red. Para implementar la noción de vecinos en este entorno descentralizado, fue necesario introducir una mecánica que definiera las relaciones entre nodos, asegurando una comunicación eficiente y estructurada dentro de la topología.

Esta evolución permitió una transición progresiva desde un entorno básico hacia uno completamente descentralizado, facilitando el estudio y la experimentación en escenarios más complejos.

#### 4.2.4. Implementación modular del entorno

Para garantizar la flexibilidad y adaptabilidad del entorno, fue necesario modularizar su diseño. Esta modularización permitió personalizar diversos hiperparámetros del escenario,

como el conjunto de datos utilizado, el número de nodos en la topología, las relaciones entre ellos y la distribución de clases en el conjunto de datos.

Gracias a este enfoque, el entorno se convirtió en una herramienta más versátil y adecuada para experimentar con diferentes configuraciones, facilitando la evaluación de estrategias y ajustes según las necesidades específicas de cada escenario.

#### **4.2.5. Evaluación del entorno**

El entorno debía evaluarse utilizando una métrica que reflejara su capacidad para generar muestras coherentes con las etiquetas correspondientes. Por ello, se optó por la métrica de precisión, que mide la proporción de imágenes generadas con una cualidad realista y ajustadas a las etiquetas esperadas.

Para la validación y evaluación de los escenarios, se implementó un mecanismo basado en *pipelines* de la librería *Diffusers* [28]. Este mecanismo genera imágenes asociadas a cada etiqueta entrenada por los modelos en el escenario y, posteriormente, evalúa la precisión mediante la comparación entre las etiquetas asignadas a las imágenes generadas y las obtenidas a través de un clasificador. Esta metodología garantiza un análisis sistemático del rendimiento del entorno.

Una vez definida la métrica y desarrollado el mecanismo de evaluación, se planificaron diversos casos de prueba. Estas pruebas se realizaron con tres conjuntos de datos distintos: MNIST, FashionMNIST y EMNIST, y emplearon dos configuraciones topológicas: una con 4 nodos y otra con 10 nodos. En ambos casos, todos los nodos asumieron el rol de entrenador, permitiendo explorar cómo el tamaño y la estructura de la topología afectan el rendimiento del entorno en la generación de muestras realistas.

Esta planificación proporcionó una base sólida para evaluar el entorno y explorar su rendimiento en diferentes escenarios.

## 5. Diseño e Implementación de la Solución

Esta sección detallará la solución propuesta, enfocándose primero en la implementación de los nodos en la topología, luego en el algoritmo de entrenamiento del escenario, y posteriormente en los módulos usados para la personalización del escenario.

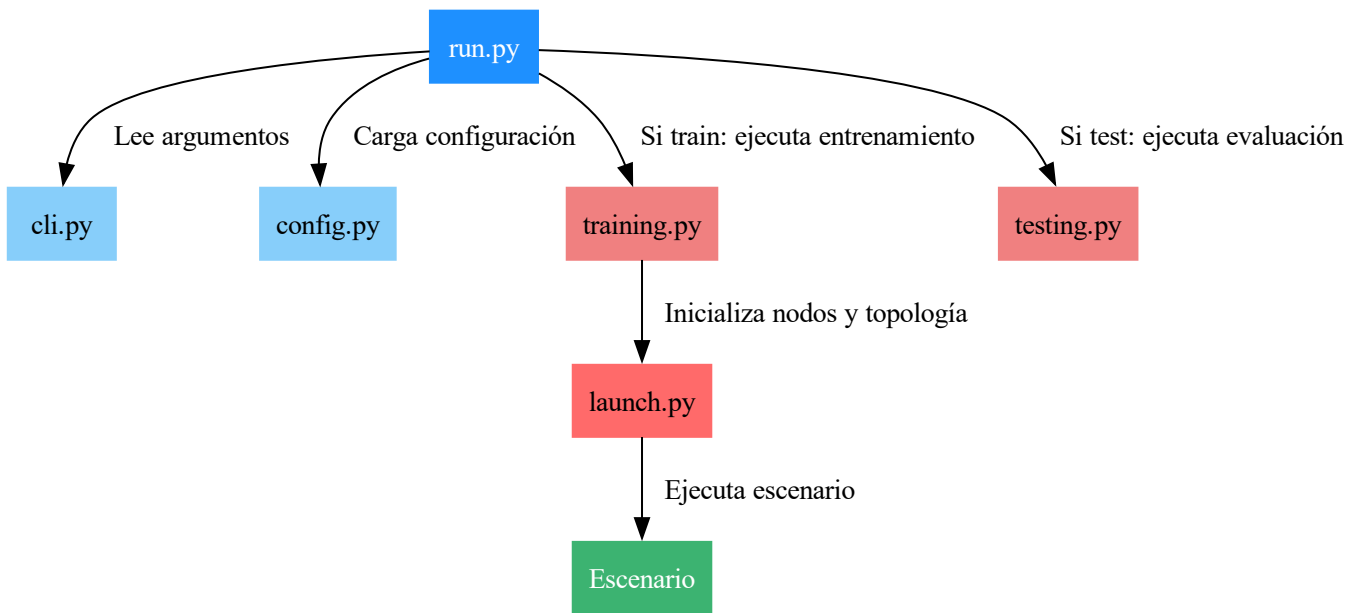


Figura 1: Diagrama del funcionamiento a alto nivel de la aplicación.

Antes de profundizar en los detalles técnicos, es importante explicar el funcionamiento general de la aplicación. Este proceso puede observarse de manera esquemática en la Figura 1.

El funcionamiento inicia con la ejecución del archivo principal, `run.py`, el cual requiere tres argumentos: el conjunto de datos a utilizar y la topología de la red. Estos argumentos se validan antes de proceder con el lanzamiento del escenario.

El lanzamiento del escenario implica dos etapas clave:

1. Inicialización de los nodos: Cada nodo se configura con su respectivo conjunto de datos según la topología seleccionada. Este paso es crucial para garantizar que los nodos cuenten con la información necesaria para entrenar sus modelos locales.

2. Ejecución del escenario: Una vez inicializados los nodos, se cargan los modelos correspondientes para cada uno de ellos y se inicia el proceso de entrenamiento. Durante esta etapa, los nodos interactúan según la topología definida, intercambiando parámetros y actualizando sus modelos.

Toda la implementación detallada, incluidas las configuraciones y el código fuente, está disponible en el repositorio donde se encuentra alojada la aplicación. Este repositorio permite explorar a fondo el diseño, el flujo de trabajo y los módulos empleados en el desarrollo de la solución [29].

### 5.1. Implementación de los nodos

Para la implementación de los nodos se ha utilizado la librería *Diffusers* [28] de *Huggingface*, que simplifica la inicialización y el manejo de modelos generativos. Esta librería permite no solo generar muestras mediante *pipelines*, sino también construir sistemas de difusión más complejos utilizando modelos preentrenados. Este enfoque facilita la integración de modelos avanzados en un entorno distribuido.

Adicionalmente, se emplea *PyTorch* [30] para el preprocesamiento de datos y el entrenamiento de las redes neuronales. Como una de las herramientas más utilizadas en aprendizaje profundo, PyTorch proporciona flexibilidad para implementar arquitecturas personalizadas y optimizar su rendimiento.

La arquitectura de red neuronal utilizada es una Red Neuronal Convolutiva (CNN) conocida como U-Net, ampliamente reconocida por su eficiencia en tareas de visión artificial. Desarrollada originalmente por el equipo de Ronneberger et al. [31], la U-Net se diseñó para la segmentación de imágenes biomédicas, destacándose por su capacidad para procesar datos con alta precisión. En este trabajo, la U-Net se adapta a las características del DDPM, sirviendo como base para procesar datos ruidosos y generar imágenes precisas.

La U-Net, tal y como se puede observar en la Figura 2, está compuesta por dos bloques principales: el codificador y el decodificador. El codificador se encarga de extraer características relevantes mediante varias capas convolucionales combinadas con operaciones de activación no lineales, como ReLU, y técnicas de *downsampling*, como max-pooling. Estas operaciones permiten reducir la resolución espacial mientras se incrementa la profundidad de las representaciones de características, capturando información contextual de alto nivel. Por su parte, el decodificador reconstruye la resolución original de la entrada mediante capas convolucionales acompañadas de operaciones de *upsampling*, como convoluciones transpuestas. Este proceso genera una salida con las mismas dimensiones que la entrada, preservando la información crítica gracias a las conexiones de salto (*skip connections*) entre las capas correspondientes del codificador y el decodificador.

En nuestra implementación, la U-Net se adapta para trabajar con conjuntos de datos

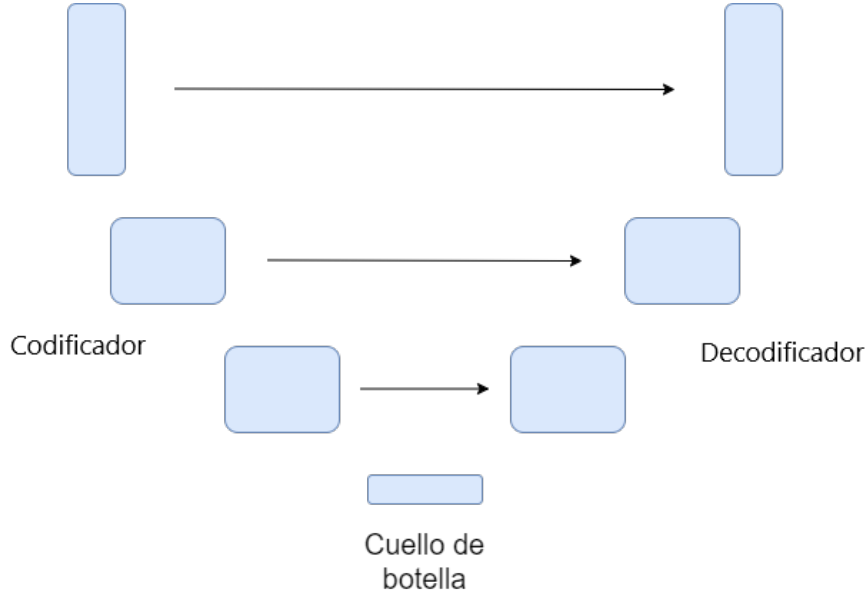


Figura 2: Arquitectura U-Net. Cada bloque representa a un mapa de características multi-canal.

en blanco y negro, configurándose con 3 capas de codificador y 3 capas de decodificador. En el codificador, la primera capa toma como entrada imágenes de un solo canal y las expande a 64 canales, lo que permite capturar características básicas. La segunda capa aumenta la profundidad a 128 canales mientras reduce la resolución espacial, permitiendo el modelado de patrones más complejos. Finalmente, la tercera capa incorpora mecanismos de atención espacial para enfocar las regiones más relevantes antes de reducir aún más la resolución.

El decodificador, por otro lado, comienza con una capa que aplica atención espacial para recuperar detalles críticos perdidos durante la codificación. Posteriormente, las siguientes capas expanden la resolución de la imagen, reconstruyendo progresivamente las características hasta alcanzar el formato original. La última capa restaura la salida a una representación coherente con las dimensiones iniciales, asegurando que las características aprendidas sean preservadas.

Los datos utilizados provienen de los conjuntos de datos MNIST, FashionMNIST y EMNIST Letters, obtenidos mediante la librería *Torchvision* [32]. Estos conjuntos se seleccionan en función del escenario de simulación y se adaptan a un formato uniforme de 32x32 píxeles. Este redimensionamiento garantiza que las operaciones de convolución y *pooling* sean óptimas en términos de memoria y eficiencia computacional. Además, cada nodo filtra las imágenes según las etiquetas asignadas, entrenando únicamente con las clases permitidas en su configuración. Esto asegura que la red neuronal sea capaz de

aceptar etiquetas y adaptarse a diferentes clases de entrada, permitiendo la generación condicionada de muestras acorde a las necesidades específicas de cada nodo.

Cabe destacar que las imágenes del conjunto se redimensionan a 32x32 píxeles, puesto que las operaciones de convolución y *pooling* están optimizadas para usar bloques de datos que son potencias de 2, tanto por una memoria y cálculo alineados, como por hardware optimizado.

Para mejorar la convergencia en escenarios donde los datos no están distribuidos uniformemente, se implementa una estrategia opcional de compartición de etiquetas. Esta estrategia permite a los nodos acceder a un porcentaje limitado de datos correspondientes a clases que no forman parte de su asignación inicial. Por defecto, cada nodo recibe un 10 % de los datos de las etiquetas no asignadas, lo que enriquece su capacidad para representar y generar muestras de estas clases.

En términos de comunicación, los nodos mantienen una lista de vecinos, que determina con quiénes pueden intercambiar parámetros de modelo. Esta lista se implementa como un vector booleano donde cada posición representa un nodo, indicando si es vecino o no. Este diseño simplifica la gestión de relaciones entre nodos y permite adaptarse fácilmente a diferentes topologías distribuidas.

## 5.2. Algoritmo federado descentralizado

```

N ← (set of nodes)
for each node n in N in parallel do
    initialize n

for each round t = 1, 2, ... do
    for each node n in N in parallel do
        wt ← TrainModel(n)
        send wt to neighbours
    until training done

    for each node n in N in parallel do
        aggregate received wt
    
```

Tabla 2: Algoritmo federado descentralizado

Tal como se muestra en la Tabla 2, el proceso comienza con la inicialización de cada nodo, donde se preparan la red neuronal, el conjunto de datos y la lista de vecinos.

Una vez que los nodos han sido inicializados, se da inicio a un período de rondas globales. Durante estas rondas, cada nodo entrena su modelo local utilizando su conjunto de datos

asignado. Al finalizar el entrenamiento, los pesos del modelo resultante se comparten con los nodos vecinos según la configuración de la topología de la red. Tras completar el intercambio de información, cada nodo procede a realizar la agregación de los pesos recibidos, combinándolos con los propios mediante el cálculo de la media. Este proceso de entrenamiento, intercambio y agregación de pesos se repite de manera iterativa hasta que se completan todas las rondas globales definidas.

```

TrainModel(n):
    B ← (set of batches)
    for batch b ∈ B do
        n ← (noise)
        x ← (image)
        n(x)
        repeat
            predict noise np (n(x))
            calculate loss l (np, n(x))
            propagate backwards (l)
            update parameters
        until converged
    return weights

```

Tabla 3: Algoritmo de entrenamiento local

El algoritmo que sigue cada nodo para entrenar se puede observar en la Tabla 3. El nodo utiliza un algoritmo de difusión para entrenar el modelo, basado en el funcionamiento del DDPM. El proceso comienza generando una muestra de ruido gaussiano blanco, que posteriormente se aplica a una serie de imágenes siguiendo el procedimiento del proceso de difusión. Este proceso añade ruido de manera progresiva a las imágenes, simulando el deterioro de los datos en varias etapas.

Durante el entrenamiento, el modelo predice los residuos de ruido en cada paso temporal utilizando las imágenes con ruido como entrada. La pérdida se calcula como el error cuadrático medio (MSE) entre los residuos de ruido predichos por el modelo y el ruido realmente añadido a las imágenes. Esta pérdida se retropropaga para actualizar los parámetros del modelo, calculando los gradientes necesarios. Los gradientes se recortan mediante técnicas de *gradient clipping* para evitar inestabilidades numéricas y explosión de gradientes durante el entrenamiento.

A continuación, los parámetros del modelo se actualizan utilizando el optimizador AdamW [33], una mejora del algoritmo Adam [34]. AdamW introduce un enfoque más robusto al aplicar decaimiento del peso de forma explícita, mejorando la generalización y mitigando los problemas asociados con una selección subóptima del factor de regulari-

zación. Este procedimiento se repite para todos los lotes (*batches*) del conjunto de datos asignado al nodo, completando así una iteración de entrenamiento local.

Para facilitar el entrenamiento distribuido en múltiples GPUs o TPUs, se emplea la librería *Accelerate* [35] de *Huggingface*. Esta herramienta simplifica la paralelización y el manejo de dispositivos, optimizando el rendimiento en escenarios de aprendizaje federado descentralizado. Una vez que el nodo ha completado el entrenamiento con todos los lotes, devuelve los pesos actualizados de su modelo, que posteriormente se compartirán con los nodos vecinos como parte del algoritmo descentralizado. Este enfoque asegura la robustez y la eficiencia del entrenamiento en escenarios distribuidos.

### 5.3. Modularización del entorno

La solución propuesta cuenta con un módulo *run* que acepta parámetros que determinan el tipo de escenario a ejecutar. Estos parámetros son el conjunto de datos que se quiere utilizar y el identificador de la topología que se quiera lanzar. En concreto, el módulo valida la entrada del usuario extrayendo así los parámetros. Al recibir el identificador de la topología, el módulo busca si existe dicha topología en el conjunto de topologías almacenadas, por lo que es necesario diseñar una topología antes de producir un escenario. Tras haber validado la entrada del usuario y haber extraído la información de la topología almacenada, esto es, un grafo que representa las relaciones de cada nodo de la topología y la distribución de etiquetas a lo largo de dicha topología, se envían al módulo *launch*.

El módulo *launch* fue diseñado para que recibiera la información del conjunto de datos a usar en el escenario, el grafo de relaciones entre nodos, y la distribución de etiquetas. Con esta información, se encarga de preparar el escenario con la correspondiente inicialización de los nodos. Dicha inicialización requiere asignarle a cada nodo una lista de nodos vecinos con los que puede comunicarse, y un conjunto de datos previamente filtrado con el que puede trabajar. Esta filtración se realiza según la lista de etiquetas asignada al nodo. Finalmente, tras inicializar los nodos del escenario, procede a ejecutar el escenario.

Con el objetivo de persistir los modelos entrenados y toda información relevante para su posterior reanudación, se produce una salida cada  $X$  rondas, siendo  $X$  un número que el usuario decida en la configuración de la ejecución. Esta salida consiste en los pesos de los modelos, en una cuadrícula de muestras, habiendo una muestra por clase, y un fichero *log* que contiene la última ronda trabajada y el número de pasos globales del escenario. De esta manera, si se procede a reanudar la ejecución del escenario posterior a su detención, se lee el fichero *log* para retomar la ejecución desde ese instante.

La estructura interna del entorno es tal y como aparece en la Figura 3. Como ya se dijo anteriormente, el módulo *main* extrae la información de la topología indicada en el directorio *Laboratory/Topologies/*, y posteriormente, tras la correcta inicialización y ejecución



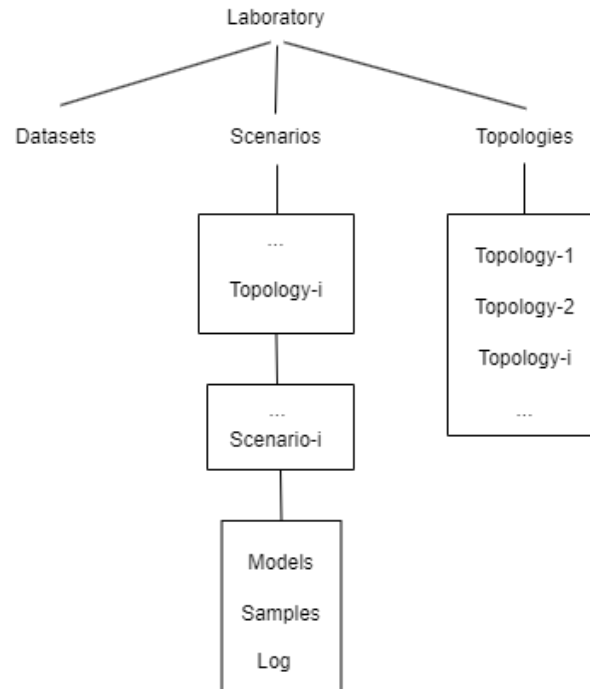


Figura 3: Estructura interna del entorno

del entorno, se guardan los modelos, muestras generadas por dichos modelos y el log con la información relevante en el directorio *Laboratory/Scenarios/Topology-i/Scenario-i/*

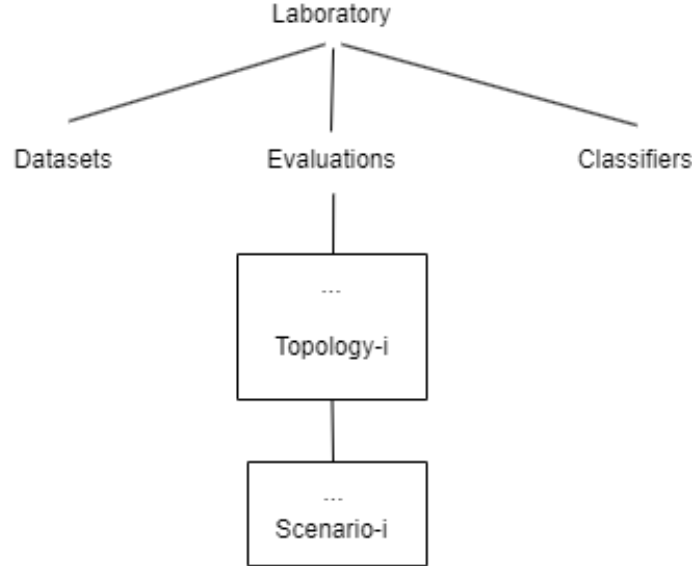


Figura 4: Estructura interna del entorno - Evaluación

## 6. Análisis de Resultados

Esta sección detalla los resultados obtenidos por 16 casos de estudio distintos. Para la evaluación de los distintos casos de estudio, se empleará la métrica de precisión. Con el objetivo de conseguir esta métrica, se han usado clasificadores propios de cada tipo de conjunto de datos, esto es, un clasificador MNIST, FashionMNIST y EMNIST Letters. Las muestras generadas se pueden consultar en el Apéndice I, el Apéndice II y el Apéndice III.

Los casos de estudio están divididos principalmente según la topología estudiada. Primero se estudiarán los casos de la Topología anillo, que está formada por 4 nodos en una topología circular, tal y como se puede observar en la Figura 5. Tras terminar de evaluar esta primera topología, se continuará con la topología personalizada, formada por 10 nodos de la manera representada en la Figura 6.

Para las evaluaciones se implementó un parámetro en la entrada del usuario que define si se quiere realizar una evaluación, manteniendo el resto de la entrada igual que en el caso de entrenamiento. Escoge el clasificador según el conjunto de datos introducido y empieza a generar imágenes de los modelos del escenario almacenados. Posteriormente, estas imágenes se pasan por el clasificador para que genere unas predicciones de las etiquetas que representan estas imágenes generadas. Junto a las predicciones generadas y a las etiquetas reales, se obtiene la métrica de precisión, que posteriormente se almacena en el directorio *Laboratory/Evaluations/Topology-i/Scenario-i*, tal y como se puede observar en la Figura 4.

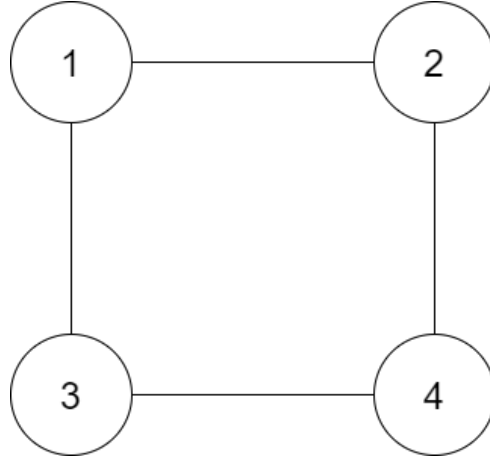


Figura 5: Topología anillo

### 6.1. Topología anillo

La topología utilizada en este escenario corresponde a una configuración de anillo, en la cual los 4 nodos están conectados de forma circular, como se ilustra en la Figura 5. En este tipo de topología, cada nodo está directamente conectado únicamente con sus dos vecinos inmediatos, lo que facilita un flujo ordenado de información y permite que los parámetros se propaguen eficientemente a lo largo del anillo. Esta estructura simplificada puede favorecer la convergencia de los modelos locales, ya que la comunicación limitada y directa entre nodos reduce la complejidad asociada con la agregación de parámetros en redes más densamente conectadas.

Se comenzará evaluando el caso de prueba del conjunto de datos MNIST. Para este caso, se ha ejecutado el escenario un total de 50 rondas. La distribución de las etiquetas es la siguiente:

- Nodo 1: 0, 1, 2
- Nodo 2: 3, 4, 5
- Nodo 3: 6, 7
- Nodo 4: 8, 9

Nodo 1	Nodo 2	Nodo 3	Nodo 4
Precisión: 95 %	Precisión: 97.59 %	Precisión: 95.8 %	Precisión: 96.2 %

Tabla 4: Resultados del caso de prueba MNIST sin estrategia de compartición en 50 rondas | Topología anillo

Al analizar la Tabla 4, se observa que la precisión alcanzada es considerablemente alta en cada nodo, llegando a una aproximación del 96 %. No obstante, existen ciertas muestras cuya calidad no es fácilmente perceptible para el ojo humano, como se muestra en la Tabla 30. Estas muestras corresponden a los dígitos 2 y 8 generados por el nodo 1; los dígitos 2, 6 del nodo 2; y el dígito 8 del nodo 4.

Una opción sería activar la compartición de etiquetas en la topología, de esa manera cada nodo tiene una idea mínima de cómo debería generarse una clase, permitiendo converger de manera óptima con los pesos de otros nodos.

Nodo 1	Nodo 2	Nodo 3	Nodo 4
Precisión: 98.8 %	Precisión: 98.2 %	Precisión: 97.8 %	Precisión: 97.8 %

Tabla 5: Resultados del caso de prueba MNIST con estrategia de compartición (10 %) en 50 rondas | Topología anillo

La Tabla 5 muestra los resultados obtenidos en el caso de prueba del conjunto de datos MNIST al aplicar una estrategia de compartición de etiquetas con un 10 %. Se observa un aumento del 2 % en la precisión y una mejora en la calidad perceptiva de las muestras generadas (Tabla 31), aunque persisten algunas excepciones. Estas excepciones incluyen el dígito 7 generado por el nodo 1, y el dígito 2 producido tanto por el nodo 2 como por el nodo 3. Esto podría atribuirse a la necesidad de realizar un mayor número de rondas para alcanzar un modelo óptimo, o bien a la posibilidad de que un aumento en el porcentaje de compartición de etiquetas mejore los resultados.

A continuación, se evalúa el caso de prueba del conjunto de datos FashionMNIST. Para este caso, se ha ejecutado el escenario un total de 50 rondas con compartición de etiquetas al 10 %. Como se entendió con el anterior experimento que sin aplicar una estrategia de compartición de etiquetas las muestras generadas no eran del todo perceptibles, se ha decidido empezar la experimentación del conjunto de datos FashionMNIST con la aplicación de la estrategia de compartición de etiquetas. La distribución de las etiquetas es la siguiente:

- Nodo 1: 0 T-shirt/top, 1 Trouser, 2 Pullover
- Nodo 2: 3 Dress, 4 Coat, 5 Sandal
- Nodo 3: 6 Shirt, 7 Sneaker
- Nodo 4: 8 Bag, 9 Ankle boot

Observando la Tabla 6, la precisión es alta, aunque no tan alta como en el caso del conjunto de datos MNIST, alcanzando una aproximación del 87 %. A pesar de esa diferencia

Nodo 1	Nodo 2	Nodo 3	Nodo 4
Precisión: 86.8 %	Precisión: 86 %	Precisión: 88.4 %	Precisión: 87.59 %

Tabla 6: Resultados del caso de prueba FashionMNIST con estrategia de compartición (10 %) en 50 rondas | Topología anillo

de precisión, respecto al ojo humano son bien diferenciables las muestras (Tabla 38). Es posible que haga falta entrenar más rondas para que la precisión mejore.

Nodo 1	Nodo 2	Nodo 3	Nodo 4
Precisión: 88.4 %	Precisión: 88.6 %	Precisión: 87.8 %	Precisión: 85.79 %

Tabla 7: Resultados del caso de prueba FashionMNIST con estrategia de compartición (10 %) en 100 rondas | Topología anillo

La Tabla 7 refleja las muestras generadas por el caso de prueba del conjunto de datos FashionMNIST tras entrenar 100 rondas. La precisión global no parece mejorar, y las muestras generadas siguen esta tendencia (Tabla 39), por lo que se opta por aumentar el % de compartición de etiquetas para comprobar si la precisión mejora acorde.

Nodo 1	Nodo 2	Nodo 3	Nodo 4
Precisión: 88.8 %	Precisión: 88.8 %	Precisión: 88.19 %	Precisión: 87.4 %

Tabla 8: Resultados del caso de prueba FashionMNIST con estrategia de compartición (20 %) en 50 rondas | Topología anillo

Por cómo se ve la Tabla 8, la precisión global ni mejora ni empeora. Si además se revisa la Tabla 40, que contiene las muestras generadas por los modelos de este caso de experimentación, la calidad de las muestras no mejora tampoco, por lo que se concluye que la limitación debe estar en la propia arquitectura.

Finalmente, se evalúa el caso de prueba del conjunto de datos EMNIST Letters. Para este caso, se ha ejecutado el escenario un total de 50 rondas con compartición de etiquetas al 10 %. La distribución de las etiquetas es la siguiente:

- Nodo 1: 0 Aa, 1 Bb, 2 Cc, 3 Dd, 4 Ee, 5 Ff, 6 Gg
- Nodo 2: 7 Hh, 8 Ii, 9 Jj, 10 Kk, 11 Ll, 12 Mm
- Nodo 3: 13 Nn, 14 Oo, 15 Pp, 16 Qq, 17 Rr, 18 Ss, 19 Tt

Nodo 1	Nodo 2	Nodo 3	Nodo 4
Precisión: 91.9 %	Precisión: 89.68 %	Precisión: 90.28 %	Precisión: 90.28 %

Tabla 9: Resultados del caso de prueba EMNIST Letters con estrategia de compartición (10 %) en 50 rondas | Topología anillo

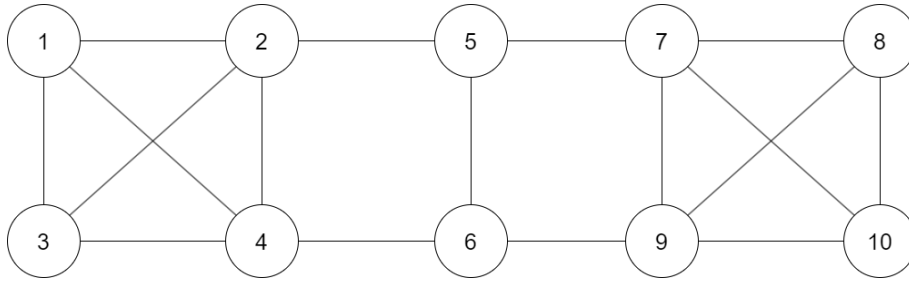


Figura 6: Topología personalizada

- Nodo 4: 20 Uu, 21 Vv, 22 Ww, 23 Xx, 24 Yy, 25 Zz

La Tabla 9 muestra que la precisión alcanzada es alta, alcanzando una aproximación del 90.5 %. Aunque la mayoría de las muestras son fácilmente diferenciables (Tabla 47), se identifican algunos casos en los que las letras generadas no son claramente distinguibles. Estas excepciones incluyen la letra J generada por el nodo 1, las letras G y Q producidas por el nodo 2, y las letras D y J generadas por el nodo 3. Esto sugiere que podría ser necesario incrementar el porcentaje de compartición de etiquetas para mejorar la diferenciación en estos casos específicos. Por ende, se aumenta el % de compartición de etiquetas al 20 %.

Nodo 1	Nodo 2	Nodo 3	Nodo 4
Precisión: 89.47 %	Precisión: 89.68 %	Precisión: 91.5 %	Precisión: 90.49 %

Tabla 10: Resultados del caso de prueba EMNIST Letters con estrategia de compartición (20 %) en 50 rondas | Topología anillo

Echando un vistazo a la Tabla 10, la precisión se mantiene igual de alta. Además, las muestras realmente son igual de perceptibles que antes (Tabla 48), por lo que la limitación se debería encontrar intrínseca a la arquitectura.

## 6.2. Topología personalizada

Esta topología se vuelve más compleja que la anterior. Presenta dos secciones de 4 nodos totalmente conectados, que se unen mediante dos nodos intermedios, tal y como se observa en la Figura 6.

Se comenzará evaluando el caso de prueba del conjunto de datos MNIST. Para este caso, se ha ejecutado el escenario un total de 50 rondas. La distribución de las etiquetas es la siguiente:

- Nodo 1: 0, 1
- Nodo 2: 1, 2
- Nodo 3: 2, 3
- Nodo 4: 3, 4
- Nodo 5: 4, 5
- Nodo 6: 5, 6
- Nodo 7: 6, 7
- Nodo 8: 7, 8
- Nodo 9: 8, 9
- Nodo 10: 9, 0

Nodo 1	Nodo 2	Nodo 3	Nodo 4	Nodo 5
Precisión: 51.4 %	Precisión: 51.4 %	Precisión: 50 %	Precisión: 53 %	Precisión: 37.59 %

Tabla 11: Resultados del caso de prueba MNIST sin estrategia de compartición en 50 rondas | Topología personalizada

Nodo 6	Nodo 7	Nodo 8	Nodo 9	Nodo 10
Precisión: 37.79 %	Precisión: 47.6 %	Precisión: 50.8 %	Precisión: 43.79 %	Precisión: 49.2 %

Tabla 12: Resultados del caso de prueba MNIST sin estrategia de compartición en 50 rondas - 2 | Topología personalizada

La Tabla 11 y la Tabla 12 evidencian una precisión significativamente baja, en contraste con los resultados obtenidos en el mismo caso de prueba dentro de la topología anillo. Además, las muestras generadas son, en muchos casos, difíciles de distinguir (Tabla 32,

Tabla 33). Los únicos casos claramente distinguibles incluyen los dígitos del 0 al 4 generados por los nodos 1, 2, 3 y 4; el dígito 6 generado por el nodo 5; los dígitos 4 y 5 producidos por el nodo 6; los dígitos del 6 al 9 generados por los nodos 7, 8 y 10; y los dígitos del 7 al 9 generados por el nodo 9. Cabe destacar que los nodos con dificultades para generar imágenes de ciertas etiquetas suelen estar ubicados en áreas de la topología donde no se entrenan dichas etiquetas, lo que sugiere una relación directa entre la ubicación topológica y la capacidad de representación de etiquetas.

Dentro de estos pésimos resultados, se puede observar en la precisión sacada de los nodos 5 y 6, que son nodos intermedios de las dos secciones totalmente conectadas de la topología, y que aquí es donde se nota la debilidad del aprendizaje federado descentralizado, pues al no formar parte de una sección totalmente conectada, tienen menos influencia de los pesos de los modelos que entrenan con el resto de etiquetas.

Es importante anotar que una topología con nodos totalmente conectados asegura sacar buenos resultados con cualquier etiqueta en cualquier nodo, funcionando de manera similar a una topología centralizada que tiene a un servidor agregando parámetros para obtener un modelo global.

Para el siguiente caso de prueba, se aumenta el número de rondas de ejecución del escenario a 100 rondas, puesto que la topología ha crecido en tamaño y hará falta más tiempo para converger en un modelo óptimo.

Nodo 1	Nodo 2	Nodo 3	Nodo 4	Nodo 5
Precisión: 48 %	Precisión: 50.2 %	Precisión: 48.59 %	Precisión: 51.2 %	Precisión: 45.4 %

Tabla 13: Resultados del caso de prueba MNIST sin estrategia de compartición en 100 rondas | Topología personalizada

Nodo 6	Nodo 7	Nodo 8	Nodo 9	Nodo 10
Precisión: 43.2 %	Precisión: 47.6 %	Precisión: 47.2 %	Precisión: 47.79 %	Precisión: 46.4 %

Tabla 14: Resultados del caso de prueba MNIST sin estrategia de compartición en 100 rondas - 2 | Topología personalizada

Como es notorio en la Tabla 13 y la Tabla 14, las precisiones de los nodos de las secciones totalmente conectadas de la topología se mantienen iguales, a diferencia de los nodos intermedios (5 y 6), cuyas precisiones han aumentado aproximadamente un 17.5 %, y cuyas muestras generadas de las etiquetas que no entrenan, pero sí sus nodos vecinos más próximos, se diferencian muy poco (Tabla 34, Tabla 35). Esta mejora en los nodos



intermedios se debe a que los modelos han tenido suficientes rondas para converger en un resultado, aunque el resultado no sea bueno.

Para este siguiente caso de prueba, se decide activar la compartición de etiquetas en la topología al 10 %, con la esperanza de que mejoren los resultados.

Nodo 1	Nodo 2	Nodo 3	Nodo 4	Nodo 5
Precisión: 98.19 %	Precisión: 97.8 %	Precisión: 98.4 %	Precisión: 97.4 %	Precisión: 98.19 %

Tabla 15: Resultados del caso de prueba MNIST con estrategia de compartición (10 %) en 100 rondas | Topología personalizada

Nodo 6	Nodo 7	Nodo 8	Nodo 9	Nodo 10
Precisión: 98.6 %	Precisión: 97.8 %	Precisión: 97.19 %	Precisión: 98.2 %	Precisión: 97.6 %

Tabla 16: Resultados del caso de prueba MNIST con estrategia de compartición (10 %) en 100 rondas - 2 | Topología personalizada

La Tabla 15 y la Tabla 16 reflejan las muestras generadas por el caso de prueba del conjunto de datos MNIST aplicando la compartición de etiquetas en un 10 %. La precisión aumenta enormemente, alcanzando una mejora del 106 % aproximadamente, y las muestras son totalmente más perceptibles que antes (Tabla 36, Tabla 37).

Es evidente que una topología más descentralizada se beneficia significativamente de la estrategia de compartición de etiquetas, al menos en el caso del conjunto de datos MNIST. Este beneficio radica en que los nodos tienen la posibilidad de entrenar con etiquetas no asignadas originalmente, lo que mejora su conocimiento sobre estas clases y facilita la convergencia hacia una solución más equilibrada y precisa.

Se continúa evaluando los casos de prueba del conjunto de datos FashionMNIST. Para el primer caso, se ha ejecutado el escenario un total de 100 rondas sin compartición de etiquetas, para comprobar que se cumple la misma situación respecto al conjunto de datos MNIST. La distribución de las etiquetas es la siguiente:

- Nodo 1: 0 T-shirt/top, 1 Trouser
- Nodo 2: 1 Trouser, 2 Pullover
- Nodo 3: 2 Pullover, 3 Dress
- Nodo 4: 3 Dress, 4 Coat
- Nodo 5: 4 Coat, 5 Sandal

- Nodo 6: 5 Sandal, 6 Shirt
- Nodo 7: 6 Shirt, 7 Sneaker
- Nodo 8: 7 Sneaker, 8 Bag
- Nodo 9: 8 Bag, 9 Ankle boot
- Nodo 10: 9 Ankle boot, 0 T-shirt/top

Nodo 1	Nodo 2	Nodo 3	Nodo 4	Nodo 5
Precisión: 50.8 %	Precisión: 48.59 %	Precisión: 52.39 %	Precisión: 50.59 %	Precisión: 52.79 %

Tabla 17: Muestras del caso de prueba FashionMNIST sin estrategia de compartición en 100 rondas | Topología personalizada

Nodo 6	Nodo 7	Nodo 8	Nodo 9	Nodo 10
Precisión: 55.4 %	Precisión: 50.6 %	Precisión: 53.19 %	Precisión: 55.4 %	Precisión: 51 %

Tabla 18: Muestras del caso de prueba FashionMNIST sin estrategia de compartición en 100 rondas - 2 | Topología personalizada

Tal y como se observa en la Tabla 17 y la Tabla 18, los resultados obtenidos son igualmente deficientes que los del caso de prueba con el conjunto de datos MNIST. Además, se aprecia que los nodos intermedios (5 y 6) presentan precisiones similares a las del resto de nodos, probablemente debido al número de rondas de entrenamiento realizadas. En las muestras generadas, también se evidencia una considerable confusión respecto a las prendas que debían producir (Tabla 41, Tabla 42). Entre las muestras confusas se encuentran zapatillas, bolsos y botas cortas generadas por los nodos 1 al 6, así como camisetas, pantalones, suéteres, vestidos y abrigos producidos por los nodos 7 al 10.

Se avanza con el siguiente caso de prueba, aplicando una estrategia de compartición de etiquetas al 10 %.

Nodo 1	Nodo 2	Nodo 3	Nodo 4	Nodo 5
Precisión: 91.79 %	Precisión: 89.6 %	Precisión: 89 %	Precisión: 89.19 %	Precisión: 86.4 %

Tabla 19: Muestras del caso de prueba FashionMNIST con estrategia de compartición (10 %) en 100 rondas | Topología personalizada

Como se observa en la Tabla 19 y la Tabla 20, la precisión ha mejorado aproximadamente un 69 %, un resultado directamente atribuido a la aplicación de la estrategia de

Nodo 6	Nodo 7	Nodo 8	Nodo 9	Nodo 10
Precisión: 86 %	Precisión: 85.59 %	Precisión: 88 %	Precisión: 87 %	Precisión: 86.6 %

Tabla 20: Muestras del caso de prueba FashionMNIST con estrategia de compartición (10 %) en 100 rondas - 2 | Topología personalizada

compartición de etiquetas. Además, las muestras generadas son claramente distinguibles para el ojo humano y se corresponden adecuadamente con las etiquetas objetivo, lo que evidencia que los modelos cumplen eficazmente con su propósito (Tabla 43, Tabla 44).

Como último caso de prueba con el conjunto de datos FashionMNIST, se opta por aumentar el % de la compartición de etiquetas al 20 % con el objetivo de comprobar si es posible mejorar la precisión.

Nodo 1	Nodo 2	Nodo 3	Nodo 4	Nodo 5
Precisión: 88.4 %	Precisión: 92 %	Precisión: 90 %	Precisión: 90.4 %	Precisión: 89.2 %

Tabla 21: Muestras del caso de prueba FashionMNIST con estrategia de compartición (20 %) en 100 rondas | Topología personalizada

Nodo 6	Nodo 7	Nodo 8	Nodo 9	Nodo 10
Precisión: 86.6 %	Precisión: 82 %	Precisión: 86.4 %	Precisión: 84 %	Precisión: 85 %

Tabla 22: Muestras del caso de prueba FashionMNIST con estrategia de compartición (20 %) en 100 rondas - 2 | Topología personalizada

Viendo la Tabla 21 y la Tabla 22, la precisión no varía, y además las muestras generadas se mantienen con la misma calidad (Tabla 45, Tabla 46), por lo que se concluye que no hay mejora en aumentar el % de compartición de etiquetas.

Finalmente, se evalúan los casos de prueba del conjunto de datos EMNIST Letters. Para el primer caso, se ha ejecutado el escenario un total de 100 rondas sin estrategia de compartición de etiquetas, nuevamente para validar que se cumple el mismo caso que con los conjuntos de datos MNIST y FashionMNIST. La distribución de las etiquetas es la siguiente:

- Nodo 1: 0 Aa, 1 Bb, 2 Cc
- Nodo 2: 3 Dd, 4 Ee, 5 Ff
- Nodo 3: 6 Gg, 7 Hh, 8 Ii

- Nodo 4: 9 Jj, 10 Kk, 11 Ll
- Nodo 5: 12 Mm, 13 Nn, 14 Oo
- Nodo 6: 15 Pp, 16 Qq, 17 Rr
- Nodo 7: 18 Ss, 19 Tt, 20 Uu
- Nodo 8: 21 Vv, 22 Ww, 23 Xx
- Nodo 9: 24 Yy, 25 Zz
- Nodo 10: 0 Aa, 1 Bb

Nodo 1	Nodo 2	Nodo 3	Nodo 4	Nodo 5
Precisión: 43.93 %	Precisión: 42.92 %	Precisión: 44.33 %	Precisión: 44.13 %	Precisión: 37.45 %

Tabla 23: Muestras del caso de prueba EMNIST Letters sin estrategia de compartición en 100 rondas | Topología personalizada

Nodo 6	Nodo 7	Nodo 8	Nodo 9	Nodo 10
Precisión: 34.21 %	Precisión: 34.62 %	Precisión: 37.65 %	Precisión: 33.8 %	Precisión: 35.02 %

Tabla 24: Muestras del caso de prueba EMNIST Letters sin estrategia de compartición en 100 rondas - 2 | Topología personalizada

Observando la Tabla 23 y la Tabla 24, la precisión es tan baja como en el mismo caso de prueba con los conjuntos de datos MNIST y FashionMNIST, por lo que se concluye que con una topología descentralizada como esta, es vital aplicar la estrategia de compartición de etiquetas. En adición, las muestras generadas no son entendibles, teniendo en cuenta que deben ser las letras del alfabeto latino (Tabla 49 , Tabla 50).

Se procede a aplicar la estrategia de compartición de etiquetas al 10 %.

Nodo 1	Nodo 2	Nodo 3	Nodo 4	Nodo 5
Precisión: 86.64 %	Precisión: 85.02 %	Precisión: 86.64 %	Precisión: 84.01 %	Precisión: 84.41 %

Tabla 25: Muestras del caso de prueba EMNIST Letters con estrategia de compartición (10 %) en 100 rondas | Topología personalizada

Como era de esperar, al igual que en el mismo caso de prueba con otros conjuntos de datos, se obtienen precisiones muy altas, alcanzando el 87 % aproximadamente (Tabla 25,

Nodo 6	Nodo 7	Nodo 8	Nodo 9	Nodo 10
Precisión: 87.65 %	Precisión: 88.87 %	Precisión: 87.25 %	Precisión: 89.88 %	Precisión: 90.49 %

Tabla 26: Muestras del caso de prueba EMNIST Letters con estrategia de compartición (10 %) en 100 rondas - 2 | Topología personalizada

Tabla 26), mejorando la precisión en un 124 %. Las muestras generadas también son perceptibles y comprensibles, aunque se identifican ciertos problemas al generar algunas letras correctamente escritas (Tabla 51, Tabla 52). Estas imprecisiones corresponden a las letras A, D, G, H, T y Z generadas por el nodo 1; las letras G, N, Q y R producidas por el nodo 2; las letras Q y R por el nodo 3; las letras G, Q, R y S por el nodo 4; las letras E y G por el nodo 5; las letras A, B y F por el nodo 6; las letras G, H y J por el nodo 7; las letras F, K y S por el nodo 8; las letras A y F por el nodo 9; y las letras F, G, Q y R por el nodo 10.

Se pretende aumentar el % de compartición de etiquetas al 20 % para verificar si la calidad de las muestras generadas mejora.

Nodo 1	Nodo 2	Nodo 3	Nodo 4	Nodo 5
Precisión: 87.85 %	Precisión: 84.21 %	Precisión: 89.07 %	Precisión: 85.43 %	Precisión: 86.64 %

Tabla 27: Muestras del caso de prueba EMNIST Letters con estrategia de compartición (20 %) en 100 rondas | Topología personalizada

Nodo 6	Nodo 7	Nodo 8	Nodo 9	Nodo 10
Precisión: 87.45 %	Precisión: 90.28 %	Precisión: 89.27 %	Precisión: 89.47 %	Precisión: 91.3 %

Tabla 28: Muestras del caso de prueba EMNIST Letters con estrategia de compartición (20 %) en 100 rondas - 2 | Topología personalizada

Viendo la Tabla 27 y la Tabla 28, las precisiones no han variado, cumpliendo así con el mismo caso con los demás conjuntos de datos. Aún así, revisando las muestras generadas (Tabla 53, Tabla 54), la calidad de algunas letras complicadas de generar ha mejorado. Se comprende que se necesitan métricas que reflejen mejor los resultados obtenidos, pues la métrica de la precisión se queda corta si queremos evaluar estos casos de prueba.

Para la comparación, se seleccionaron los resultados de los casos de prueba correspondientes a los modelos que implementaron la estrategia de compartición de etiquetas, utilizando un 10 % como límite superior. Este porcentaje se determinó como óptimo, ya

Tabla 29: Comparación de resultados con la literatura

Phoenix CIFAR-10 10-k	Topología anillo MNIST 10 %	Topología anillo Fashion- MNIST 10 %	Topología anillo EMNIST Letters 10 %	Topología personalizada MNIST 10 %	Topología personalizada Fashion- MNIST 10 %	Topología personalizada EMNIST Letters 10 %
Precisión: 83.03 %	Precisión: 98.8 %	Precisión: 88.6 %	Precisión: 91.9 %	Precisión: 98.6 %	Precisión: 91.79 %	Precisión: 90.49 %

que, según las observaciones realizadas, incrementar la compartición de etiquetas más allá del 10 % no generaba mejoras adicionales en las precisiones, dado el diseño de la arquitectura.

Al analizar la comparación de resultados entre nuestros casos de prueba y los reportados en la literatura, como se muestra en la Tabla 29, se observa que los resultados obtenidos son superiores. Es importante destacar que, aunque la precisión alcanzada en el caso de prueba con el conjunto de datos MNIST es notablemente alta, esto se debe en gran medida a la simplicidad inherente de dicho conjunto. Por esta razón, se emplearon otros conjuntos de datos más complejos para validar la propuesta. Los resultados obtenidos confirman que el rendimiento del entorno es consistentemente elevado.

## 7. Conclusiones y Vías Futuras

La aplicación de modelos generativos de difusión en DFL tiene muchas ventajas de cara al mundo real. Cuando se combinan, la mezcla de modelos generativos de difusión con el aprendizaje federado descentralizado proporciona una potente sinergia para mejorar la privacidad, la eficiencia y la solidez durante el entrenamiento de modelos distribuidos. Esta combinación encuentra su mayor utilidad en las aplicaciones que requieren privacidad de datos como la sanidad, la banca y otros sectores sensibles a los datos.

En este trabajo se ha creado un entorno que permite el entrenamiento federado de una topología descentralizada siguiendo las bases de DFL usando DMs, más específicamente DDPMs. Se ha realizado una serie de casos de prueba, en 2 topologías distintas con 3 conjuntos de datos distintos, MNIST, FashionMNIST y EMNIST Letters; y tras analizar los resultados de los modelos entrenados de cada caso de prueba, se concluye:

- La estrategia de compartición de etiquetas mostró una mejora significativa en la convergencia de los modelos. En particular, la precisión aumentó en un 124 % en la topología más compleja al implementar esta técnica en comparación con escenarios sin compartición de etiquetas.
- La precisión no es suficiente para evaluar completamente la calidad de las imágenes generadas, ya que, a pesar de obtener valores elevados, las imágenes generadas pueden ser diferenciadas por el ojo humano. Se recomienda considerar métricas adicionales como FID para una evaluación más precisa.
- El número de rondas necesarias para alcanzar la convergencia es proporcional a la complejidad tanto de la topología como del conjunto de datos utilizado. Por ejemplo, los escenarios con EMNIST Letters requerían más de rondas para converger en comparación con MNIST, debido a su mayor número de clases y diversidad de datos.
- Una topología de nodos completamente conectados mostró un comportamiento similar al de una topología CFL en términos de convergencia hacia un modelo global, lo que indica que la comunicación directa entre nodos puede igualar el desempeño de un servidor central en escenarios menos complejos.

Con esta propuesta, se ha logrado mantener la privacidad al no compartir datos entre los nodos, aunque siguen existiendo otros problemas de privacidad que en este trabajo no se han abordado, y deberían abordarse en futuros trabajos. Teniendo como objetivo mejorar este entorno, se consideran las siguientes vías futuras para este trabajo:

- Investigar las limitaciones inherentes a la arquitectura de la solución.
- Implementar fine-tuning en la propuesta para mejorar la calidad de entrenamiento pudiendo usar más conjuntos de datos.

- Implementar un mecanismo de asignación de roles, pudiendo diferenciarse: entrenadores, agregadores y proxies.
- Reducir la proporción de comunicación entre nodos a cada cierto número de rondas, disminuyendo así el estrés sobre el ancho de banda.
- Emplear una métrica distinta a la precisión para evaluar la calidad de las imágenes generadas.
- Investigar los problemas de privacidad que afectan a los escenarios de DFL.



## I. Muestras generadas en MNIST

Nodo 1	Nodo 2	Nodo 3	Nodo 4
0	0	0	0
1	1	1	1
2	2	2	2
3	3	3	3
4	4	4	4
5	5	5	5
6	6	6	6
7	7	7	7
8	8	8	8
9	9	9	9

Tabla 30: Muestras del caso de prueba MNIST sin estrategia de compartición en 50 rondas  
| Topología anillo

Nodo 1	Nodo 2	Nodo 3	Nodo 4
6 1 2 3 4 5 6 7 8 9	0 1 2 3 4 5 6 7 8 9	0 1 2 3 4 5 6 7 8 9	0 1 2 3 4 5 6 7 8 9

Tabla 31: Muestras del caso de prueba MNIST con estrategia de compartición (10%) en 50 rondas | Topología anillo

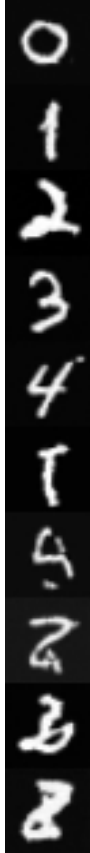
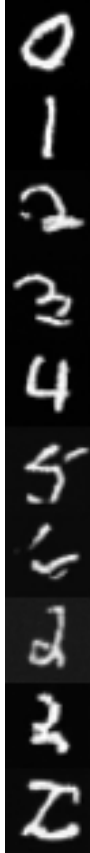

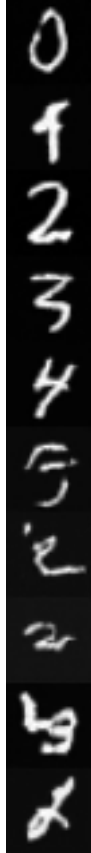

Nodo 1	Nodo 2	Nodo 3	Nodo 4	Nodo 5
				

Tabla 32: Muestras del caso de prueba MNIST sin estrategia de compartición en 50 rondas  
| Topología personalizada



















































Nodo 6	Nodo 7	Nodo 8	Nodo 9	Nodo 10
				
				
				
				
				
				
				
				
				
				

Tabla 33: Muestras del caso de prueba MNIST sin estrategia de compartición en 50 rondas  
- 2 | Topología personalizada

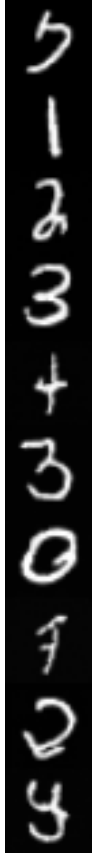




Nodo 1	Nodo 2	Nodo 3	Nodo 4	Nodo 5
				

Tabla 34: Muestras del caso de prueba MNIST sin estrategia de compartición en 100 rondas | Topología personalizada



















































Nodo 6	Nodo 7	Nodo 8	Nodo 9	Nodo 10
				
				
				
				
				
				
				
				
				
				

Tabla 35: Muestras del caso de prueba MNIST sin estrategia de compartición en 100 rondas - 2 | Topología personalizada

Nodo 1	Nodo 2	Nodo 3	Nodo 4	Nodo 5
0	0	0	0	0
1	1	1	1	1
2	2	2	2	2
3	3	3	3	3
4	4	4	4	4
5	5	5	5	5
6	6	6	6	6
7	7	7	7	7
8	8	8	8	8
9	9	9	9	9

Tabla 36: Muestras del caso de prueba MNIST con estrategia de compartición (10 %) en 100 rondas | Topología personalizada



















































Nodo 6	Nodo 7	Nodo 8	Nodo 9	Nodo 10
				
				
				
				
				
				
				
				
				
				

Tabla 37: Muestras del caso de prueba MNIST con estrategia de compartición (10 %) en 100 rondas - 2 | Topología personalizada



## II. Muestras generadas en FashionMNIST



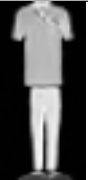

































Nodo 1	Nodo 2	Nodo 3	Nodo 4
			
			
			
			
			
			
			
			
			

Tabla 38: Muestras del caso de prueba FashionMNIST con estrategia de compartición (10 %) en 50 rondas | Topología anillo





Nodo 1	Nodo 2	Nodo 3	Nodo 4
			

Tabla 39: Muestras del caso de prueba FashionMNIST con estrategia de compartición (10 %) en 100 rondas | Topología anillo





Nodo 1	Nodo 2	Nodo 3	Nodo 4
			

Tabla 40: Muestras del caso de prueba FashionMNIST con estrategia de compartición (20 %) en 50 rondas | Topología anillo






Nodo 1	Nodo 2	Nodo 3	Nodo 4	Nodo 5
				

Tabla 41: Muestras del caso de prueba FashionMNIST sin estrategia de compartición en 100 rondas | Topología personalizada






Nodo 6	Nodo 7	Nodo 8	Nodo 9	Nodo 10
				

Tabla 42: Muestras del caso de prueba FashionMNIST sin estrategia de compartición en 100 rondas - 2 | Topología personalizada






Nodo 1	Nodo 2	Nodo 3	Nodo 4	Nodo 5
				

Tabla 43: Muestras del caso de prueba FashionMNIST con estrategia de compartición (10 %) en 100 rondas | Topología personalizada






Nodo 6	Nodo 7	Nodo 8	Nodo 9	Nodo 10
				

Tabla 44: Muestras del caso de prueba FashionMNIST con estrategia de compartición (10 %) en 100 rondas - 2 | Topología personalizada






Nodo 1	Nodo 2	Nodo 3	Nodo 4	Nodo 5
				

Tabla 45: Muestras del caso de prueba FashionMNIST con estrategia de compartición (20 %) en 100 rondas | Topología personalizada








Nodo 6	Nodo 7	Nodo 8	Nodo 9	Nodo 10
				

Tabla 46: Muestras del caso de prueba FashionMNIST con estrategia de compartición (20 %) en 100 rondas - 2 | Topología personalizada

### III. Muestras generadas en EMNIST Letters

Tabla 47: Muestras del caso de prueba EMNIST Letters con estrategia de compartición (10 %) en 50 rondas | Topología anillo





Nodo 1	Nodo 2	Nodo 3	Nodo 4
			

Tabla 48: Muestras del caso de prueba EMNIST Letters con estrategia de compartición (20 %) en 50 rondas | Topología anillo





Nodo 1	Nodo 2	Nodo 3	Nodo 4
			

Tabla 49: Muestras del caso de prueba EMNIST Letters sin estrategia de compartición en 100 rondas | Topología personalizada






Nodo 1	Nodo 2	Nodo 3	Nodo 4	Nodo 5
				

Tabla 50: Muestras del caso de prueba EMNIST Letters sin estrategia de compartición en 100 rondas - 2 | Topología personalizada

Nodo 6	Nodo 7	Nodo 8	Nodo 9	Nodo 10

Tabla 51: Muestras del caso de prueba EMNIST Letters con estrategia de compartición (10 %) en 100 rondas | Topología personalizada






Nodo 1	Nodo 2	Nodo 3	Nodo 4	Nodo 5
				

Tabla 52: Muestras del caso de prueba EMNIST Letters con estrategia de compartición (10 %) en 100 rondas - 2 | Topología personalizada






Nodo 6	Nodo 7	Nodo 8	Nodo 9	Nodo 10
				

Tabla 53: Muestras del caso de prueba EMNIST Letters con estrategia de compartición (20 %) en 100 rondas | Topología personalizada











Nodo 1	Nodo 2	Nodo 3	Nodo 4	Nodo 5
				



Tabla 54: Muestras del caso de prueba EMNIST Letters con estrategia de compartición (20 %) en 100 rondas - 2 | Topología personalizada

Nodo 6	Nodo 7	Nodo 8	Nodo 9	Nodo 10
				

## Siglas

CFL	Aprendizaje Federado Centralizado
CNN	Red Neuronal Convolutiva
DDPM	Modelo Probabilístico de Difusión para la Eliminación de Ruido
DFL	Aprendizaje Federado Descentralizado
DM	Modelo de Difusión
DNN	Red Neuronal Profunda
FID	Distancia de Inicio de Fréchet
FIMI	Llenando lo que Falta
FL	Aprendizaje Federado
GAI	Inteligencia Artificial Generativa
GAN	Red Generativa Adversativa
IA	Inteligencia Artificial
IID	Independiente e Idénticamente Distribuido
IPFS	Sistema de Archivos Interplanetario
LLM	Modelo de Lenguaje Grande
RNN	Red Neuronal Recurrente
SDA-FL	Aprendizaje Federado Asistido por Datos Sintéticos
VAE	Codificador Automático Variacional

## Referencias

- [1] Jakub Konečný, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 8, 2016.
- [2] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.
- [3] Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2022.
- [4] I Goodfellow, J Pouget-Abadie, M Mehdi, B Xu, D Warde-Farley, S Ozair, A Courville, and Y Bengio. Proceedings of the international conference on neural information processing systems nips 2014. 2014.
- [5] Samuel Dupond. A thorough review on the current advance of neural network structures. *Annual Reviews in Control*, 14(14):200–230, 2019.
- [6] George EP Box, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.
- [7] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.
- [8] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International conference on machine learning*, pages 8162–8171. PMLR, 2021.
- [9] Chenshuang Zhang, Chaoning Zhang, Mengchun Zhang, and In So Kweon. Text-to-image diffusion model in generative ai: A survey. *arXiv preprint arXiv:2303.07909*, 2023.
- [10] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in neural information processing systems*, 35:36479–36494, 2022.
- [11] Yogesh Balaji, Seungjun Nah, Xun Huang, A Vahdat, J Song, K Kreis, and MY Liu. ediffi: Text-to-image diffusion models with an ensemble of expert denoisers. *arxiv 2022. arXiv preprint arXiv:2211.01324*, 2022.

- [12] Zhida Feng, Zhenyu Zhang, Xintong Yu, Yewei Fang, Lanxin Li, Xuyi Chen, Yuxiang Lu, Jiayang Liu, Weichong Yin, Shikun Feng, et al. Ernie-vilg 2.0: Improving text-to-image diffusion model with knowledge-enhanced mixture-of-denoising-experts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10135–10145, 2023.
- [13] Anusha Lalitha, Shubhanshu Shekhar, Tara Javidi, and Farinaz Koushanfar. Fully decentralized federated learning. In *Third workshop on bayesian deep learning (NeurIPS)*, volume 2, 2018.
- [14] Abhijit Guha Roy, Shayan Siddiqui, Sebastian Pölsterl, Nassir Navab, and Christian Wachinger. Braintorrent: A peer-to-peer environment for decentralized federated learning. *arXiv preprint arXiv:1905.06731*, 2019.
- [15] Chenghao Hu, Jingyan Jiang, and Zhi Wang. Decentralized federated learning: A segmented gossip approach. *arXiv preprint arXiv:1908.07782*, 2019.
- [16] Yuzheng Li, Chuan Chen, Nan Liu, Huawei Huang, Zibin Zheng, and Qiang Yan. A blockchain-based decentralized federated learning framework with committee consensus. *IEEE Network*, 35(1):234–241, 2020.
- [17] Xingjian Cao, Gang Sun, Hongfang Yu, and Mohsen Guizani. Perfed-gan: Personalized federated learning via generative adversarial networks. *IEEE Internet of Things Journal*, 10(5):3749–3762, 2022.
- [18] Zijian Li, Jiawei Shao, Yuyi Mao, Jessie Hui Wang, and Jun Zhang. Federated learning with gan-based data synthesis for non-iid clients. In *International Workshop on Trustworthy Federated Learning*, pages 17–32. Springer, 2022.
- [19] Peichun Li, Hanwen Zhang, Yuan Wu, Liping Qian, Rong Yu, Dusit Niyato, and Xue-min Shen. Filling the missing: Exploring generative ai for enhanced federated learning over heterogeneous mobile edge devices. *IEEE Transactions on Mobile Computing*, 2024.
- [20] Dinh C Nguyen, Ming Ding, Pubudu N Pathirana, Aruna Seneviratne, and Albert Y Zomaya. Federated learning for covid-19 detection with generative adversarial networks in edge cloud computing. *IEEE Internet of Things Journal*, 9(12):10257–10271, 2021.
- [21] Zhao Wang, Yifan Hu, Jun Xiao, and Chao Wu. Efficient ring-topology decentralized federated learning with deep generative models for industrial artificial intelligent. *arXiv preprint arXiv:2104.08100*, 2021.

- [22] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.
- [23] Xumin Huang, Peichun Li, Hongyang Du, Jiawen Kang, Dusit Niyato, Dong In Kim, and Yuan Wu. Federated learning-empowered ai-generated content in wireless networks. *IEEE Network*, 2024.
- [24] Timur Sattarov, Marco Schreyer, and Damian Borth. Fedtabdiff: Federated learning of diffusion probabilistic models for synthetic mixed-type tabular data generation. *arXiv preprint arXiv:2401.06263*, 2024.
- [25] Daixun Li, Weiyang Xie, Zixuan Wang, Yibing Lu, Yunsong Li, and Leyuan Fang. Feddiff: Diffusion model driven federated learning for multi-modal and multi-clients. *IEEE Transactions on Circuits and Systems for Video Technology*, 2024.
- [26] Mingzhao Yang, Shangchao Su, Bin Li, and Xiangyang Xue. Exploring one-shot semi-supervised federated learning with pre-trained diffusion models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 16325–16333, 2024.
- [27] Fiona Victoria Stanley Jothiraj and Afra Mashhadi. Phoenix: A federated generative diffusion model, 2023.
- [28] Patrick von Platen, Suraj Patil, Anton Lozhkov, Pedro Cuenca, Nathan Lambert, Kashif Rasul, Mishig Davaadorj, Dhruv Nair, Sayak Paul, Steven Liu, William Berman, Yiyi Xu, and Thomas Wolf. Diffusers: State-of-the-art diffusion models. URL <https://github.com/huggingface/diffusers>.
- [29] Maxim Utica Babyak. Diffusion DFL Implementation, January 2025. URL <https://github.com/Shiroi-Max/diffusion-dfl-implementation>.
- [30] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.
- [31] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation, 2015. URL <https://arxiv.org/abs/1505.04597>.
- [32] TorchVision maintainers and contributors. TorchVision: PyTorch’s Computer Vision library, November 2016. URL <https://github.com/pytorch/vision>.

- [33] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization, 2019. URL <https://arxiv.org/abs/1711.05101>.
- [34] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017. URL <https://arxiv.org/abs/1412.6980>.
- [35] Sylvain Gugger, Lysandre Debut, Thomas Wolf, Philipp Schmid, Zachary Mueller, Sourab Mangrulkar, Marc Sun, and Benjamin Bossan. Accelerate: Training and inference at scale made simple, efficient and adaptable. <https://github.com/huggingface/accelerate>, 2022.