FedGP Resilience: A Comparative Study with Standard Federated Aggregation Methods under Adversarial Scenarios

Elia Pacioni[1,2][0000−0002−1557−4870], Célien Muller[1][0009−0002−5377−0415], Francisco Fernández De Vega[2][0000−0002−1086−1483], and Davide Calvaresi[1][0000−0001−9816−7439]

[1] HES-SO Valais-Wallis, 3960 Sierre, Switzerland
{elia.pacioni, davide.calvaresi}@hevs.ch, celien.muller@students.hevs.ch
[2] University of Extremadura, 06800 Mérida, Spain
fcofdez@unex.es

Abstract. Federated Learning (FL) enables decentralized model training while preserving data privacy. However, classical aggregation strategies such as FedAVG, FedPROX, and FedNOVA show significant performance degradation when client data distributions are highly non-IID. They are also vulnerable to adversarial perturbations that corrupt client updates. To overcome these limitations, FedGP has been introduced in literature, a Genetic Programming (GP)–based aggregation approach that evolves symbolic aggregation functions instead of relying on fixed rules.This study investigates the resilience of FedGP in challenging FL scenarios. Thanks to its adaptive mechanism, the server can dynamically combine, reweight, or discard client contributions according to their reliability, improving robustness against noisy or malicious updates. We conduct a comparative evaluation on PathMNIST and FashionMNIST under standard and adversarial conditions, including Gaussian noise, label-flipping, and sign-flipping attacks. All methods are tested under controlled non-IID settings with an imbalance rate of 0.8 and identical training hyperparameters to ensure fairness. Experimental results show that FedGP consistently outperforms traditional aggregation methods in terms of robustness and accuracy, particularly in the presence of corrupted clients.

Keywords: FedGP · Federated Learning · Genetic Programming · Aggregation method resilience

1 Introduction

Federated Learning (FL) has emerged as a central paradigm in distributed machine learning, allowing models to be trained on decentralized data without transferring sensitive information to a central server [6, 14]. This approach has facilitated the adoption of distributed learning-based applications in privacy-sensitive domains, such as healthcare [23] and finance [12], as well as large-scale

mobile ecosystems (e.g., next-word prediction on smartphones) [5], where data locality and communication efficiency are crucial.

At the core of FL lies the model aggregation step: client devices train local models on private data and periodically send their updates to a coordinating server, which fuses them into a new global model [14]. FedAVG is the *de facto* standard strategy. It average the weights of the client models and propagates the resulting model. Widely appreciated for its simplicity and scalability, this approach is known to suffer if client data are non-IID, unbalanced, or generated by heterogeneous devices, often leading to slower convergence and degraded global performance [10, 22, 25].

Besides statistical heterogeneity, FL is also exposed to several structural fragilities. Communication constraints, client dropouts, and hardware heterogeneity can delay or destabilize training; non-IID or imbalanced data partitions can bias the global model. Among these challenges, adversarial or faulty clients pose a critical threat to the integrity of the aggregation phase. These factors underscore the need for aggregation strategies that are accurate under standard conditions and resilient to noise, manipulation, and other forms of attack. A key limitation of existing approaches is that the aggregation rule itself is predefined and static, typically relying on fixed arithmetic operations such as averaging or normalization. This rigidity prevents the model from adapting to dynamic or corrupted client behaviors, suggesting that aggregation should instead be treated as a learnable process—one capable of evolving functional forms that respond to changing conditions.

Genetic Programming (GP) has been widely used to automatically synthesize mathematical expressions and programs tailored to complex objectives [7, 16, 17]. When the aggregation step in FL is viewed as a symbolic regression problem, i.e., discovering a function that best combines heterogeneous client updates, GP becomes a natural candidate for obtaining adaptive, data-driven aggregation rules, thereby overcoming the rigidity of fixed averaging schemes. Following this idea, FedGP [18, 19] has been proposed as a GP-based aggregation method in which the server evolves the aggregation function itself, represented as a tree that operates directly on client model parameters, selecting, reweighting, or discarding them as needed. Such a mechanism is, by design, more suitable for adverse and non-IID scenarios, where some client contributions may be unreliable.

To date, FedGP has proven to be superior to traditional aggregation methods in FL [20]. In this paper, we present a comparative study of FedGP against three widely adopted federated aggregation methods: FedAVG, FedPROX [9, 10], and FedNOVA [25]. The comparison is carried out on two publicly available image classification datasets, PathMNIST [29] and FashionMNIST [26], under standard and adversarial conditions (i.e., including Gaussian noise, label flipping, and sign flipping), within a centralized and synchronous FL architecture. The results show that FedGP achieves the best overall performance and higher robustness in all settings. Nevertheless, FedNOVA remains competitive in some scenarios. FedAVG and FedPROX exhibit a higher sensitivity to corrupted or

heterogeneous updates. These findings confirm that FL resilience-oriented aggregation strategies are promising for real-world deployments.

The remainder of this paper is organized as follows. Section 2 reviews related work on FL aggregation and adversarial scenarios. Section 3 details the experimental methodology, including datasets, adversarial settings, baselines, and evaluation metrics. Section 4 reports and discusses the results obtained. Finally, Section 5 summarizes the main contributions, limitations, and outlines future research directions.

2 State of the Art

Since its introduction, FL has evolved from a privacy-preserving framework into a mature distributed learning paradigm encompassing diverse communication and optimization strategies [6, 14]. Beyond architectural variations—such as decentralized or hierarchical coordination [1] and asynchronous communication protocols [28]—a fundamental distinction in FL research concerns the statistical nature of the clients' data distributions. When data are independent and identically distributed (IID), clients contribute samples drawn from the same underlying distribution, and global convergence is typically stable. In contrast, under non-IID conditions, clients exhibit heterogeneous data distributions that differ in content, quantity, or both, leading to model divergence and biased global updates [6]. Two common forms of statistical heterogeneity are label skew, where clients possess disjoint or uneven class distributions, and quantity skew, where the number of samples per client varies significantly. These sources of non-IID settings make the aggregation phase particularly critical, as they amplify the imbalance among client contributions and challenge the stability of the global optimization process.

Consequently, the design of robust and adaptive aggregation mechanisms has become a major research focus [21], with particular attention to methods that can maintain stable convergence despite heterogeneous, noisy, or adversarial client behavior.

2.1 Aggregation Methods

Model aggregation defines how client updates are combined in each round. Among the numerous strategies proposed, three methods have gained particular prominence for their simplicity, robustness, and theoretical grounding: FedAVG, FedPROX, and FedNOVA.

FedAVG. Proposed by McMahan et al. [14], FedAVG represents the de facto standard for FL aggregation. It computes the weighted or unweighted average of client model weights, where the weights are proportional to the size of each client's dataset. Its simplicity, scalability, and widespread adoption have made it the baseline for nearly all subsequent research. However, its performance deteriorates in non-IID settings, where client data distributions differ significantly, leading to biased updates and slower convergence [10, 22].

FedPROX. FedPROX [9, 10] extends FedAVG by introducing a proximal term μ (which reduces to FedAVG when $\mu = 0$) into the local objective function to mitigate client drift. This regularization term anchors local updates toward the global model, stabilizing training under heterogeneous data and unbalanced participation. The algorithm preserves FedAVG's lightweight implementation while improving convergence in non-IID environments.

FedNOVA. FedNOVA [25] addresses the issue of objective inconsistency that arises when clients perform different numbers of local optimization steps. It normalizes each local update according to the number of local iterations, ensuring that the aggregated gradient remains unbiased and that convergence is not dominated by overactive clients. This approach is particularly effective in systems characterized by strong heterogeneity in computation.

Together, these methods represent the main aggregation paradigms currently adopted in FL: averaging (FedAVG), proximal regularization (FedPROX), and normalization (FedNOVA). Despite their practical success, they all rely on static aggregation rules, which may not adapt to dynamic or adversarial conditions.

2.2 GP for Adaptive Aggregation

To overcome the rigidity of fixed aggregation schemes, recent research has explored the use of evolutionary computation techniques to automatically design adaptive aggregation functions. GP [7, 16, 17] is particularly suited for this purpose, as it evolves symbolic expressions that optimize a given objective. In the FL context, the aggregation step can be framed as a symbolic regression problem—discovering a function that best combines heterogeneous client updates. This allows the aggregation mechanism to be learned rather than predefined, adapting dynamically to data distributions, noise, and client reliability.

FedGP [19] represents an implementation of this principle: an evolutionary aggregation method in which GP operates at the server level to evolve functional expressions that combine the clients' model parameters. By treating aggregation as an optimization process rather than a fixed rule, FedGP has the potential to enhance resilience and generalization in both standard and adverse scenarios.

2.3 Resilience in FL and Evolutionary Algorithms

Resilience has recently emerged as a key metric in evaluating FL systems [1, 24]. It encompasses the model's capacity to maintain stable performance despite data heterogeneity, communication failures, or adversarial perturbations. Common adversarial scenarios [4, 11, 27] include the injection of Gaussian noise, label flipping, or sign inversion, all of which can compromise the aggregation process and degrade the global model. Assessing how different aggregation methods react under such conditions provides insight into their robustness and practical reliability. In our case, we are interested in comparing the resilience of FedGP with other standard aggregators.

In the context of Evolutionary Algorithms (EAs), resilience describes the ability of the system to maintain functional performance under adverse conditions, adapting to perturbations such as noise in the data, hardware or software failures, malicious behavior, and dynamic changes in the optimization environment. It emerges as an intrinsic property of the population and adaptability that characterize EAs [3]. Miller and Goldberg show how population-based selection mechanisms can maintain stable performance even when the process is affected by noise [15]. It underlines the ability of EAs to filter out inaccurate information and converge towards promising regions of the solution space. Studies in distributed environments have shown that EAs maintain excellent tolerance to node losses and unpredictable variations in resource availability [13]. In particular, GP demonstrates a remarkable ability to continue evolutionary research despite high failure rates or intermittent participation [8]. Cotta analyzes EA behavior in the presence of Byzantine faults and intentionally manipulated fitness evaluations [2]. He shows how management mechanisms based on redundant evaluations and majority voting strategies can significantly mitigate the impact of adverse evaluations and improve the quality of the solutions obtained.

3 Methodology

This section outlines the experimental design adopted to evaluate the resilience of different aggregation strategies in FL. The workflow comprises five stages: (i) definition of adversarial scenarios, (ii) dataset(s) preparation, (iii) configuration of FL algorithms, (iv) experimental execution, and (v) evaluation and statistical analysis. All methods are tested under identical conditions to ensure fairness and reproducibility.

Adversarial Scenarios. To assess model robustness under heterogeneous and noisy environments, four distinct experimental conditions are considered: *(i)* a baseline scenario without attacks, *(ii)* Gaussian Noise, *(iii)* Label Flip, and *(iv)* Sign Flip. These scenarios represent increasingly adverse conditions that allow for the evaluation of each algorithm's stability and adaptability to client-level perturbations. The baseline test serves as a reference point for all subsequent comparisons.

3.1 FL Workflow

After introducing the general principles of FL, it is useful to briefly discuss how it works overall in order to clarify the operational context of this study. Figure 1 summarizes the typical workflow of the process, starting with the distribution of an initial model by the central server. Each client then performs local training on its own data. The updated models are then sent to the server, which combines them through an aggregation phase to obtain an updated global model. This model is then redistributed to the customers, and the cycle repeats itself.

Our work focuses precisely on this aggregation phase, exploring how the different strategies adopted in this step can influence the performance and robustness of the final model.

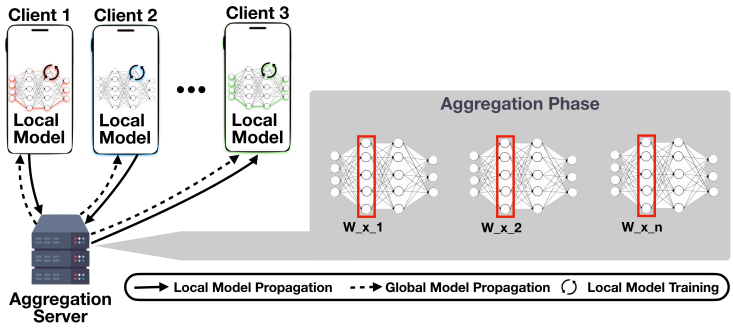


Fig. 1: FL Workflow

3.2 Dataset Selection and Characteristics

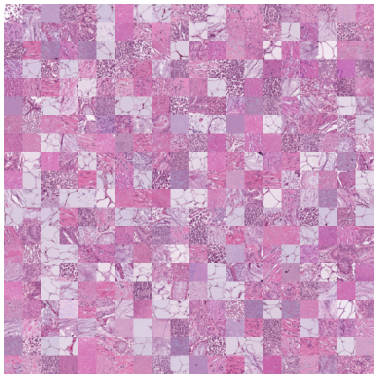
Two image-classification datasets with different levels of complexity and class distributions were used: *PathMNIST* and *FashionMNIST*. Both are well-established benchmarks for evaluating FL models under non-IID settings, as they allow the introduction of label and quantity skew among clients.

PathMNIST consists of 107,180 RGB histopathology images (28×28 pixels) derived from colorectal tissue slides (See Figure 2a) [29]. The dataset includes nine classes representing distinct tissue types and structures (e.g., adipose tissue, smooth muscle, mucus, adenocarcinoma epithelium, etc.). It exhibits a strong class imbalance, making it suitable to test aggregation robustness in complex multi-class scenarios.

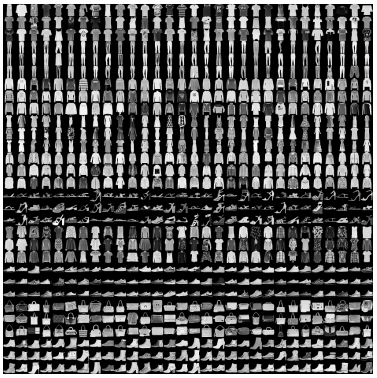
FashionMNIST is a grayscale dataset comprising 70,000 images (28×28 pixels) distributed across ten clothing categories (See Figure 2b) [26]. Although simpler than PathMNIST, it maintains sufficient variability to test resilience in lightweight yet still non-IID federated settings.

The combination of PathMNIST and FashionMNIST enables evaluation under both complex medical-like and standard computer-vision conditions, reflecting the contrasting challenges encountered in real-world FL environments.

Non-IID Data Partitioning To emulate realistic federated conditions, we adopted a shard-based non-IID partitioning strategy. For each dataset, samples were first grouped by class, and the indices of every class were randomly shuffled



(a) Sample images from the PathMNIST dataset (28×28 RGB histopathology tiles).



(b) Sample images from the FashionMNIST dataset (28×28 grayscale clothing items).

Fig. 2: Example samples from the two datasets used in this study, illustrating differences in visual complexity and domain characteristics.

and split into $3 \times N$ shards, where $N = 10$ is the number of clients. All resulting shards were then randomly permuted and distributed to the participants. A fixed fraction of shards was first assigned to the server according to the chosen, while the remaining shards were allocated to the clients in a non-uniform way using a Dirichlet distribution. The degree of heterogeneity was controlled through an *imbalance rate* parameter, set to 0.8 in all experiments. In the implementation, this value is mapped to a Dirichlet concentration parameter $\alpha = 1 - 0.8 = 0.2$, which generates highly skewed allocations, resulting in some clients receiving many shards (and therefore more samples from a few classes) while others receive only a small portion of the data. This mechanism simultaneously induces label skew (where clients observe only part of the classes) and quantity skew (where clients hold different amounts of data), closely matching non-IID federated scenarios. To avoid potential biases arising from a fixed data distribution, the non-IID partitioning process was repeated at the beginning of each experimental run using different random seeds. This ensures that every repetition of the experiment involves a distinct allocation of data across clients, allowing statistical variability to capture the effects of both model stochasticity and data heterogeneity.

3.3 Machine Learning Model

The adopted model is a feed-forward convolutional neural network (CNN). The architecture consists of five convolutional blocks followed by a fully connected classifier.

The first five layers (from layer 1 to layer 5) perform 2D convolutions, batch normalization, and ReLU activation; layers 2 and 5 also include Max Pooling operations to reduce the size of intermediate images and make the extracted

features more robust. As we move through the convolutional blocks, the number of filters increases (from 16 to 64). In this way, the network can progressively recognize more complex patterns, enabling it to learn increasingly sophisticated representations.

The final part of the network, the classifier, consists of three fully connected layers. The first two models use 128 neurons and ReLU activation functions, while the last one produces the final output, i.e., the probabilities of belonging to each class.

3.4 Federated Configuration and Training Parameters

All FL experiments adopt a centralized and synchronous architecture with 10 clients. Each client performed local training on its private data partition, and the server aggregated the resulting model updates at regular intervals. Specifically, aggregation was triggered every three local epochs, corresponding to five aggregation rounds per complete local training cycle of 15 epochs.

To ensure fair and reproducible comparisons, all aggregation algorithms shared the same global hyperparameter configuration. The batch size was fixed at 128, while the learning rate (LR) was systematically varied within a predefined values (0.0005, 0.0007, 0.002, 0.001, 0.003, 0.005, 0.01) to evaluate the sensitivity of each method under consistent conditions.

Physical configuration. Experiments were executed on a dedicated server running Ubuntu 20.04.6, equipped with two Intel® Xeon® Silver 4310 processors, 512 GB of RAM, and four NVIDIA A100 PCIe GPUs (40 GB memory each), using CUDA 12.4 and Python 3.11.10. The GP algorithm was executed on the CPU, while training, validation, and evaluation of FL models were performed on the GPU. To emulate the federated setup, client processes were executed in isolation on the same physical machine, ensuring a controlled yet realistic simulation of distributed training. All configurations were repeated 10 times to ensure statistical reliability and to account for stochastic variations introduced by both GP evolution and model initialization.

FedGP Configuration. FedGP evolves aggregation functions represented as symbolic expressions encoded in GP trees. The terminal set T (Equation 1) corresponds to the tensors received from the participating clients.

$$T = \{\text{CLIENT}_1, \text{CLIENT}_2, \ldots, \text{CLIENT}_n\}\,. \tag{1}$$

The function set F (Equation 2) includes protected arithmetic and aggregation primitives implemented with PyTorch tensor operations:

$$F = \left\{ \begin{array}{l} \text{torch.sum, torch.sub, torch.mul,} \\ \text{torch\_protected\_div, torch\_mean,} \\ \text{torch\_median, torch.abs, torch\_protected\_sqrt} \end{array} \right\} \tag{2}$$

Each individual in the population is a candidate aggregation applied to client updates. It is evaluated based on the validation accuracy of the resulting global

model. Evaluation uses a fixed validation subset that includes data from all classes. This ensures fair comparison across individuals. The evolutionary process was configured with 10 generations for each round of FL aggregation. The population is 50 individuals, elitism set to 1, a mutation rate of 0.4, a crossover rate of 0.6, and a tree depth limited between 1 and 5 levels to balance model expressiveness and generalization.

3.5 Hypothesis and Comparative Baselines

The core hypothesis driving this study is that FedGP offers higher resilience than conventional aggregation methods due to its ability to dynamically adapt aggregation functions and modulate the contribution of unreliable clients. This adaptability should enable the model to maintain stable performance under noise, label corruption, or parameter inversion, compared to the benchmarks used.

3.6 Evaluation and Statistical Analysis

Performance evaluation primarily relies on classification accuracy. To assess whether observed differences among aggregation methods are statistically significant, we employ non-parametric tests suitable for comparing multiple algorithms across repeated experiments. First, the Friedman chi-square test is used to detect global performance differences across methods. When the null hypothesis of equal performance is rejected, pairwise comparisons are conducted using the Nemenyi post-hoc test to identify significant differences between specific pairs of algorithms. Additionally, Wilcoxon signed-rank tests with Bonferroni correction are applied as complementary pairwise analyses. All tests adopt a significance level of $\alpha = 0.05$. Normality and variance homogeneity are preliminarily verified using Shapiro–Wilk and Levene's tests.

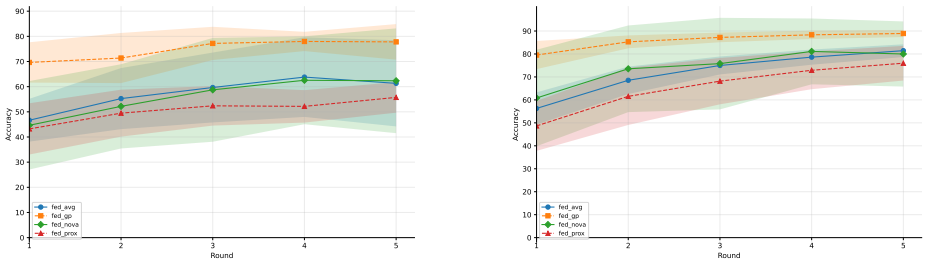
4 Results

Under standard conditions, without attacks or noise, the differences between the methods mainly reflect their ability to adapt to non-iid scenarios. On PathMNIST (Figure 3a), FedGP consistently ranks among the best-performing solutions, with average accuracy values higher than FedAvg and FedProx and significantly lower variability than FedNova, which, despite occasionally achieving good results, shows marked fluctuations between repetitions. The results are summarized in the Table 1. The Friedman test confirms the presence of significant overall differences ($\chi^2 = 9.00$, $p < 0.05$), with FedGP showing the best average rank and significant differences compared to FedProx (Wilcoxon $p \approx 0.002$). In contrast, FedGP's performance is statistically comparable to that of FedNova ($p = 0.23$), albeit with greater stability. On FashionMNIST (Figure 3b), the differences are amplified: FedGP maintains higher and more consistent performance than all traditional averaging strategies, with significant superiority over FedAvg and FedProx ($p < 0.01$) and only marginal superiority over FedNova

Table 1: Statistical comparison across all scenarios. The Friedman test results are shown in χ^2 and p_F. Columns *FedAvg*, *FedProx*, and *FedNova* report the Wilcoxon p-values comparing FedGP to each baseline method. *Best Rank (column BR)* indicates the lowest average rank in the Friedman analysis.

Dataset	**Scenario**	χ^2	p_F	FedAvg	FedProx	FedNova	**BR**
FashionMNIST	Baseline	16.560	0.00087	0.00195	0.00195	0.04883	FedGP
	Gaussian noise (1)	26.040	0.00001	0.00195	0.00195	0.00391	FedGP
	Gaussian noise (3)	18.600	0.00033	0.00195	0.00195	0.00391	FedGP
	Label flip (1)	15.600	0.00137	0.00195	0.00195	0.01953	FedGP
	Label flip (3)	13.800	0.00319	0.00391	0.00195	0.10547	FedGP
	Sign flip (1)	25.080	0.00001	0.00195	0.00195	0.04883	FedGP
	Sign flip (3)	27.231	0.00001	0.00195	0.00195	0.00195	FedGP
PathMNIST	Baseline	9.000	0.02929	0.04883	0.00195	0.23242	FedGP
	Gaussian noise (1)	23.538	0.00003	0.00195	0.00195	0.00195	FedGP
	Gaussian noise (3)	30.000	0.00000	0.00195	0.00195	0.00195	FedGP
	Label flip (1)	21.360	0.00009	0.00195	0.00195	0.00195	FedGP
	Label flip (3)	17.760	0.00049	0.00195	0.00195	0.01367	FedGP
	Sign flip (1)	20.040	0.00017	0.00195	0.00195	0.03711	FedGP
	Sign flip (3)	18.840	0.00030	0.00195	0.00195	0.00391	FedGP

($p \approx 0.049$). In this scenario, the evolutionary aggregation function is particularly effective in managing data heterogeneity, showing a significant advantage in terms of average accuracy and consistency between repetitions. Overall, even in the absence of adverse conditions, FedGP stands out for its ability to produce a more stable and consistent aggregation that is less sensitive to the non-iid nature of local distributions.

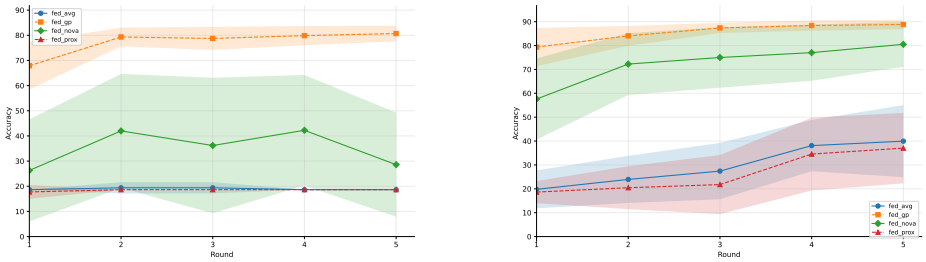


(a) PathMNIST (b) FashionMNIST

Fig. 3: Baseline comparison under normal conditions

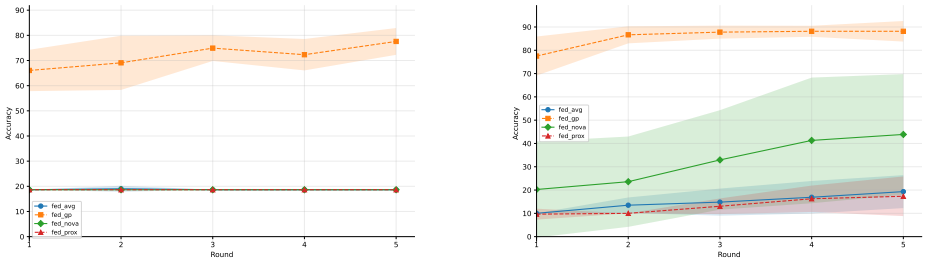
The behavior of the methods changes radically when perturbations are introduced. The injection of Gaussian noise (Figures 4, 5) causes an almost complete collapse of FedAvg and FedProx, which approach random classification values even with a single noisy client, and a significant degradation of FedNova, which maintains only marginally superior performance. In contrast, FedGP maintains

nearly unchanged accuracy levels compared to the baseline, both on PathMNIST and FashionMNIST, and shows no regressive trend as the number of malicious clients increases. The differences observed are supported by highly significant tests (PathMNIST: $\chi^2 = 23.54$–30.00, $p < 0.001$; FashionMNIST: $\chi^2 = 18.60$–26.04, $p < 0.001$), with FedGP consistently maintaining the lowest average rank and statistically confirmed differences compared to all other strategies (Wilcoxon $p < 0.005$). This behavior suggests that the aggregation function obtained from the evolutionary process effectively filters out distorted updates without penalizing the contribution of correct clients.



(a) PathMNIST (b) FashionMNIST

Fig. 4: Comparison using Gaussian noise using a malicious client.

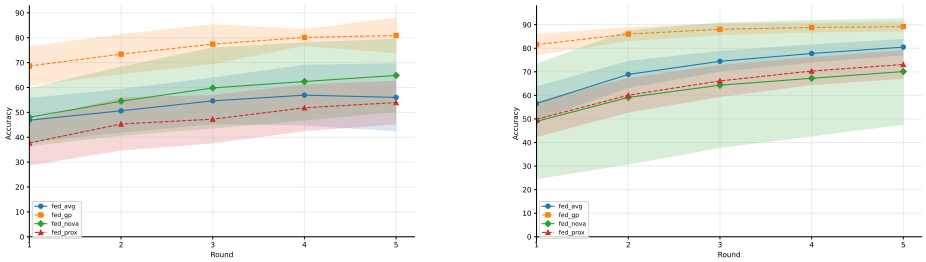


(a) PathMNIST (b) FashionMNIST

Fig. 5: Comparison using Gaussian noise through 3 malicious clients.

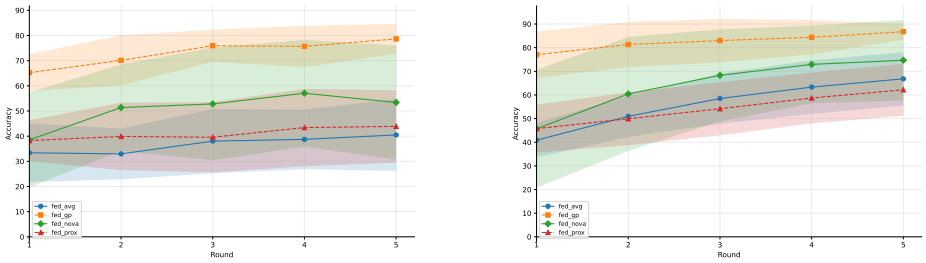
A similar trend can be observed in the Label Flip experiments (Figures 6, 7), where semantic corruption of labels progressively penalizes classical averages. FedAvg and FedProx show significant degradation as the number of corrupted clients increases, while FedNova, although partially resistant with a few malicious clients, shows a substantial reduction in stability. FedGP, on the other hand, maintains values close to the baseline even in the presence of 3 clients with inverted labels. Friedman's tests confirm significant differences in all cases

(PathMNIST: $\chi^2 = 21.36$ and 17.76; FashionMNIST: $\chi^2 = 15.60$ and 13.80; all $p < 0.005$), and post-hoc comparisons indicate FedGP as systematically superior to traditional averages ($p < 0.01$).



(a) PathMNIST (b) FashionMNIST

Fig. 6: Comparison with label flip and a malicious client.

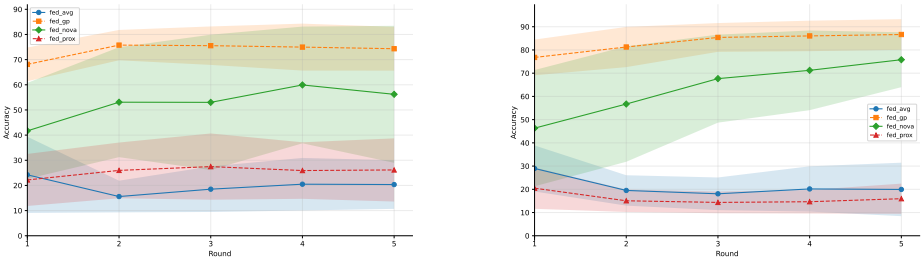


(a) PathMNIST (b) FashionMNIST

Fig. 7: Comparison with label flip and 3 malicious clients.

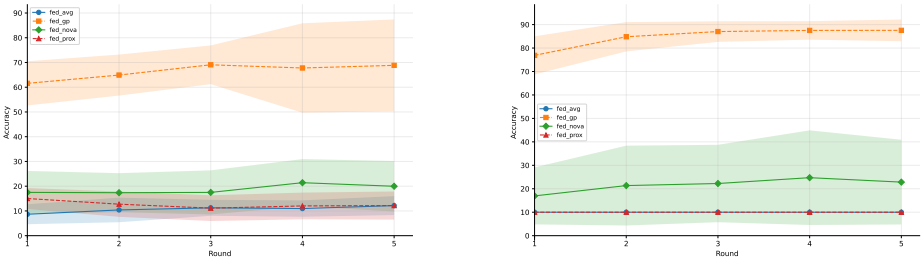
The difference becomes even more evident in the case of Sign Flip (Figures 8, 9), which represents the most destructive attack. Here, FedAvg and FedProx collapse completely with just one malicious client, and FedNova quickly loses its ability to converge as the share of inverted updates grows. Yet, FedGP is not affected, with high and stable performance regardless of the percentage of corrupted clients. Statistical tests confirm very significant overall differences in both datasets ($\chi^2 \approx 20$–27, $p < 0.001$) and pairwise differences consistently in favor of FedGP (Wilcoxon $p < 0.005$). This result confirms that evolutionary aggregation not only dampens noise but also discriminates between coherent and adverse signals, while maintaining the balance of the global update intact.

The graphs and statistical tests consistently confirm the superiority of FedGP in almost all scenarios. The observed differences are not attributable to random variations but rather reflect a structural advantage in the ability to adapt



(a) PathMNIST (b) FashionMNIST

Fig. 8: Sign Flip, 1 client malicious



(a) PathMNIST (b) FashionMNIST

Fig. 9: Sign Flip, 3 client malicious

and manage unreliable contributions, experimentally validating the hypothesis of greater resilience and stability of FedGP. The joint analysis of the different scenarios highlights two central aspects. On the one hand, cross-sectional robustness: FedGP maintains high and consistent performance on both datasets, regardless of the nature and intensity of the perturbations. On the other hand, stability with respect to the increase in malicious clients: while classical averaging strategies exhibit a degradation almost proportional to the presence of corrupt participants, FedGP maintains a regular and nearly unchanged trend, a sign of its intrinsic ability to isolate and compensate for anomalous updates without compromising global convergence.

Another aspect considered concerns the execution times of the different methods (Table 2). As expected, FedGP is significantly more computationally expensive, with average times about an order of magnitude higher than those of conventional strategies. This gap represents the primary limitation of the evolutionary approach, stemming from the design of EAs and the computational cost of the fitness function used to evaluate each individual in the population. However, this cost does not constitute an operational obstacle: the evolution of the aggregation functions occurs on the central server and does not require blocking client activities. Furthermore, aggregation is performed only at defined intervals (e.g., weekly or monthly), further reducing the overall impact on training time.

Overall, the results highlight a clear trade-off between computational cost and robustness. Although FedGP requires higher server-side resources, its stability and adaptability are unattainable by conventional aggregation strategies, confirming its potential as a foundation for resilient and adaptive FL systems.

Table 2: Average times by aggregation method.

(a) `pathmnist`

Method	Mean (s)	Std (s)
FedNova	299.45	8.00
FedAVG	304.74	4.85
FedProx	368.33	4.38
FedGP	4224.79	307.81

(b) `fashionmnist`

Method	Mean (s)	Std (s)
FedAVG	173.50	3.39
FedNova	174.00	3.60
FedProx	216.39	2.47
FedGP	5028.75	252.09

5 Conclusions

This paper compares FedGP and other standard FL aggregation methods in the context of resilience to adverse situations. The results provide evidence of FedGP's ability to maintain high accuracy and stability in both standard and adversarial settings. Unlike traditional aggregation strategies, FedGP's performance confirms its intrinsic resilience and adaptability. Even if the proportion of corrupted participants increases, FedGP maintains convergence and accuracy close to the baseline, highlighting its ability to isolate unreliable updates and preserve the integrity of the global model.

The primary limitation of FedGP is its computational cost. The evolutionary process required to identify optimal aggregation functions entails a significantly higher server-side workload compared to conventional methods. However, this overhead remains manageable, as the evolution step is executed centrally and intermittently, without disrupting client operations. The observed trade-off between computational effort and robustness suggests that the added complexity is a reasonable price for achieving enhanced resilience.

Future work will focus on improving the efficiency and scalability of FedGP. In particular, optimizing the evolutionary search through adaptive or parallel strategies and utilizing transfer learning mechanisms to reuse evolved aggregation functions across aggregation rounds, thereby reducing convergence time.

Acknowledgments. This work was partially supported by the Validate-H (PInter 14-2025), FASTER (140831) the Spanish Ministry of Economy and Competitiveness (PID2020-115570GB-C21, PID2023-147409NB-C22), funded by MCIN/AEI/10.13039/501100011033, and the Junta de Extremadura (GR24142).

References

1. Beltrán, E.T.M., Pérez, M.Q., Sánchez, P.M.S., Bernal, S.L., Bovet, G., Pérez, M.G., Pérez, G.M., Celdrán, A.H.: Decentralized federated learning: Fundamentals, state of the art, frameworks, trends, and challenges. IEEE Communications Surveys & Tutorials **25**(4), 2983–3013 (2023)
2. Cotta, C.: Tackling adversarial faults in panmictic evolutionary algorithms. In: Proceedings of the Companion Conference on Genetic and Evolutionary Computation. p. 59–60. GECCO '23 Companion, Association for Computing Machinery, New York, NY, USA (2023). https://doi.org/10.1145/3583133.3596426, https://doi.org/10.1145/3583133.3596426
3. Cotta, C., Olague, G.: Resilient bioinspired algorithms: A computer system design perspective. In: Applications of Evolutionary Computation. pp. 619–631. Springer International Publishing, Cham (2022)
4. Erdol, E.S., Erdol, H., Ustubioglu, B., Ulutas, G., Symeonidis, I.: Reducing defense vulnerabilities in federated learning: A neuron-centric approach. Applied Sciences **15**(11) (2025). https://doi.org/10.3390/app15116007, https://www.mdpi.com/2076-3417/15/11/6007
5. Hard, A., Rao, K., Mathews, R., Ramaswamy, S., Beaufays, F., Augenstein, S., Eichner, H., Kiddon, C., Ramage, D.: Federated learning for mobile keyboard prediction. arXiv preprint arXiv:1811.03604 (2019)
6. Kairouz, P., McMahan, H.B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A.N., Bonawitz, K., Charles, Z., Cormode, G., Cummings, R., et al.: Advances and open problems in federated learning. Foundations and trends® in machine learning **14**(1–2), 1–210 (2021)
7. Koza, J.R.: Genetic Programming: On the Programming of Computers by Means of Natural Selection. MIT Press (1992), http://mitpress.mit.edu/books/genetic-programming
8. Laredo, J.L.J., Bouvry, P., González, D.L., Fernández de Vega, F., Arenas, M.G., Merelo, J., Fernandes, C.M.: Designing robust volunteer-based evolutionary algorithms. Genetic Programming and Evolvable Machines **15**(3), 221–244 (2014)
9. Li, T., Sahu, A.K., Zaheer, M., Sanjabi, M., Talwalkar, A., Smith, V.: Federated optimization in heterogeneous networks. In: Dhillon, I., Papailiopoulos, D., Sze, V. (eds.) Proceedings of Machine Learning and Systems. vol. 2, pp. 429–450 (2020)
10. Li, X., Huang, K., Yang, W., Wang, S., Zhang, Z.: On the convergence of fedavg on non-iid data. arXiv preprint (2020), https://arxiv.org/abs/1907.02189
11. Liu, P., Xu, X., Wang, W.: Threats, attacks and defenses to federated learning: issues, taxonomy and perspectives. Cybersecurity **5**(1), 4 (2022)
12. Liu, Y., Ai, Z., Sun, S., Zhang, S., Liu, Z., Yu, H.: Fedcoin: A peer-to-peer payment system for federated learning. In: Federated learning: privacy and incentive. pp. 125–138. Springer (2020)
13. Lombraña González, D., Fernández de Vega, F., Casanova, H.: Characterizing fault tolerance in genetic programming. In: Proceedings of the 2009 workshop on Bio-inspired algorithms for distributed systems. pp. 1–10 (2009)
14. McMahan, B., Moore, E., Ramage, D., Hampson, S., y Arcas, B.A.: Communication-efficient learning of deep networks from decentralized data. In: Artificial intelligence and statistics. pp. 1273–1282. PMLR (2017)
15. Miller, B.L., Goldberg, D.E.: Genetic algorithms, selection schemes, and the varying effects of noise. Evolutionary computation **4**(2), 113–131 (1996)

16. Miller, J.F., Thomson, P.: Cartesian genetic programming. In: Poli, R., Banzhaf, W., Langdon, W.B., Miller, J., Nordin, P., Fogarty, T.C. (eds.) Genetic Programming. pp. 121–132. Springer Berlin Heidelberg, Berlin, Heidelberg (2000)
17. O'Neill, M., Ryan, C.: Grammatical evolution. IEEE Transactions on Evolutionary Computation **5**(4), 349–358 (2001). https://doi.org/10.1109/4235.942529
18. Pacioni, E., Fernández De Vega, F., Calvaresi, C.: Towards a meaningful communication and model aggregation in federated learning via genetic programming. In: In Proceedings of the 17th International Conference on Agents and Artificial Intelligence (ICAART 2025). vol. 3, pp. 1427–1431 (2025). https://doi.org/10.5220/0013380400003890
19. Pacioni, E., Fernández De Vega, F., Calvaresi, D.: Fedgp: Genetic programming for evolutionary aggregation in federated learning with non-iid data. In: International Conference on the Applications of Evolutionary Computation (Part of EvoStar). pp. 419–434. Springer (2025)
20. Pacioni, E., Muller, C., de Vega, F.F., Calvaresi, D.: Genetic programming in federated aggregation: A comparison of fedgp with state of the art methods. In: Proceedings of International Conference on Agents and Artificial Intelligence (ICAART), Springer LNAI (Dec 2025)
21. Qi, P., Chiaro, D., Guzzo, A., Ianni, M., Fortino, G., Piccialli, F.: Model aggregation techniques in federated learning: A comprehensive survey. Future Generation Computer Systems **150**, 272–293 (2024). https://doi.org/10.1016/j.future.2023.09.008
22. Reguieg, H., El Hanjri, M., El Kamili, M., Kobbane, A.: A comparative evaluation of fedavg and per-fedavg algorithms for dirichlet distributed heterogeneous data. In: 2023 10th International Conference on Wireless Networks and Mobile Communications (WINCOM). pp. 1–6. IEEE (2023). https://doi.org/10.1109/WINCOM59760.2023.10322899
23. Rieke, N., Hancox, J., Li, W., Milletari, F., Roth, H.R., Albarqouni, S., Bakas, S., Galtier, M.N., Landman, B.A., Maier-Hein, K., et al.: The future of digital health with federated learning. NPJ digital medicine **3**(1), 119 (2020)
24. Marin Machado de Souza, R., Holm, A., Biczyk, M., de Castro, L.N.: A systematic literature review on the use of federated learning and bioinspired computing. Electronics **13**(16), 3157 (2024)
25. Wang, J., Liu, Q., Liang, H., Joshi, G., Poor, H.V.: Tackling the objective inconsistency problem in heterogeneous federated optimization. In: Proceedings of the 34th International Conference on Neural Information Processing Systems. NIPS '20, vol. 33, pp. 7611–7623. Curran Associates Inc., Red Hook, NY, USA (2020)
26. Xiao, H., Rasul, K., Vollgraf, R.: Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms (2017)
27. Xie, X., Hu, C., Ren, H., Deng, J.: A survey on vulnerability of federated learning: A learning algorithm perspective. Neurocomputing **573**, 127225 (2024). https://doi.org/https://doi.org/10.1016/j.neucom.2023.127225, https://www.sciencedirect.com/science/article/pii/S0925231223013486
28. Xu, C., Qu, Y., Xiang, Y., Gao, L.: Asynchronous federated learning on heterogeneous devices: A survey. Computer Science Review **50**, 100595 (2023)
29. Yang, J., Shi, R., Wei, D., Liu, Z., Zhao, L., Ke, B., Pfister, H., Ni, B.: Medmnist v2-a large-scale lightweight benchmark for 2d and 3d biomedical image classification. Scientific Data **10**(1), 41 (2023)