PAMUKKALE ÜNİVERSİTESİ
Mühendislik Bilimleri Dergisi

Araştırma makalesi | Research paper

PAMUKKALE UNIVERSITY
Journal of Engineering Sciences

Cilt | Vol: XX • Sayı | No: X • ss | pp: XXXX-XXXX • Yıl | Year: 20XX

Federated aggregation strategies for credit card fraud detection under differential privacy

Murat Saran, Abdül Kadir Görür

ARTICLE INFO

Dates:
Received: 12.05.2026
Accepted: 27.08.2026

Doi:
10.65206/pajes.1949869

Corresponding author:
Murat Saran
(saran@cankaya.edu.tr)

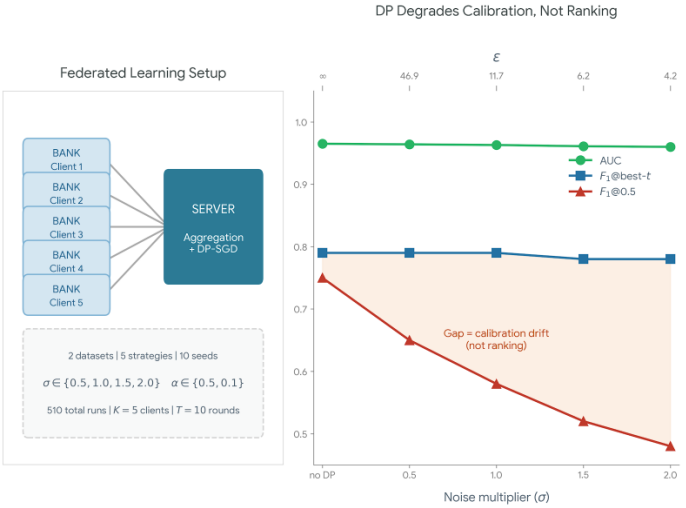
Author addresses:
Computer Engineering Department, Engineering Faculty, Çankaya University, Ankara, 06815, Türkiye
(saran@cankaya.edu.tr; agorur@cankaya.edu.tr)

ABSTRACT

Context—Card fraud cost the global financial system over $28 billion in 2019, and losses have risen every year since. Banks hold the transaction data needed for collaborative fraud detection, but privacy regulations such as GDPR and KVKK prevent cross-institutional data sharing. Federated Learning keeps raw data local: each institution trains on its own records and sends only model parameters to a shared coordinator. Differentially-Private Stochastic Gradient Descent (DP-SGD) counters gradient inversion attacks by clipping per-sample gradients and adding calibrated noise before parameters leave the client, yielding a record-level (ε, δ) guarantee whose strength depends on the accumulated privacy budget ε. How the aggregation strategy behaves under this noise has not yet been studied.

Objective—We compare five aggregation strategies—FedAvg, FedProx, cosine-similarity aggregation, FedAvg-DWA, and FedAdam—across two real-world benchmarks, two heterogeneity levels, four DP-SGD noise multipliers, and 10 random seeds per configuration. We ask whether the relative ranking of strategies survives DP noise, and what DP actually costs in deployed model behavior.

Method—510 federated training runs cover five aggregation configurations, two datasets (Kaggle ULB: n = 284807, 0.17% positive rate; IEEE-CIS: n = 590540, 3.50%), two Dirichlet non-IID levels ($\alpha \in \{0.5, 0.1\}$), and $\sigma \in \{0.5, 1.0, 1.5, 2.0\}$. The base learner is a three-layer MLP with LayerNorm. Privacy accounting uses the Opacus RDP accountant at $\delta = 10^{-5}$. Performance is reported at both the fixed 0.5 threshold and a validation-tuned F_1-maximizing threshold.



All pairwise comparisons carry Bonferroni-, Holm-, and Benjamini–Hochberg-corrected p-values, and all 200 realized client-shard partitions are characterized directly.

Results—On Kaggle ULB under IID partitioning, federated F_1@best-t (0.801–0.804) matches the centralized baseline (0.799 ± 0.036). Under mild non-IID without DP (α = 0.5), FedAvg-DWA narrowly leads (< 0.015 on F_1@best-t), though neither test survives correction. In the ULB privacy sweep, 7 of 40 Wilcoxon tests show nominal significance but none survive family-wise correction; performance consistently orders with FedAvg-DWA leading and FedAdam trailing. On IEEE-CIS, this ordering amplifies, and the extreme pair remains Holm-significant under DP (adjusted p = 0.020). While AUC holds near 0.95, calibration drift collapses precision at the fixed 0.5 threshold; a validation-tuned threshold recovers F_1 loss without privacy cost. Both calibration effects replicate on IEEE-CIS (ε = 8.10 ± 5.30 at σ = 1.0).

Conclusion—Aggregation-rule differences under DP-SGD are small and dataset-dependent in magnitude, consistent in direction, and on the ULB sweep none survives correction; outcome differences arise mainly from post-training threshold tuning. We also uncover a previously unreported interaction: Opacus's DP step quietly overrides the standard loss-augmented FedProx setup, making FedProx numerically equivalent to FedAvg, verified by bit-identical per-round global weight trajectories across seeds.

Key Words—Aggregation strategies, Credit card fraud detection, Differential privacy, Federated learning, Threshold calibration.

I. INTRODUCTION

In 2019, card fraud imposed a $28.65 billion loss on the global financial system, and this figure continues to rise [1], [2]. Banks hold rich transaction logs that could support accurate fraud-detection models, but privacy regulations, such as the GDPR in Europe [3], KVKK in Türkiye, and analogous laws in other jurisdictions, prevent them from pooling these records in one place.

Federated Learning (FL) removes the need for direct data sharing: each bank trains locally on its own transactions and transmits only model parameters to a central coordinator, which computes and redistributes an aggregated global model [4]. Parameter sharing alone does not settle the privacy question, however. Gradient inversion and membership inference attacks can partially reconstruct individual transactions from the shared updates [5]. Differentially Private Stochastic Gradient Descent (DP-SGD) [5], [6] counters this by clipping each per-sample gradient and adding calibrated Gaussian noise before any parameters leave the client, yielding a record-level (ε, δ) guarantee whose practical strength depends on the final privacy budget ε accumulated over training.

FL and DP-SGD have each been studied in the fraud-detection literature, but almost always separately. Work on aggregation rules compares strategies without privacy noise; work on private training fixes a single aggregation rule, usually FedAvg. Whether the choice of aggregation rule still matters once every client update carries DP noise is, to our knowledge, an open question, and it is the question a bank must answer before committing to a private federated deployment. This paper answers it empirically for credit card fraud detection.

A. Background and related work

The work related to this study falls into five areas: fraud-detection models, FL applied to credit card fraud detection (FL-CCFD), the design of aggregation rules, the interaction of class imbalance and differential privacy, and DP mechanisms in FL.

1. Fraud-detection models

Centralized deep-learning and ensemble methods accomplish strong results on the public benchmarks used in this field [1], [7]–[11]. Sequence models add value when temporal transaction features are available, and graph models can expose relational fraud patterns that tabular learners miss; Tang and Liang (2024), for example, detect fraud on transaction graphs with a federated graph learner [12]. Our study deliberately uses a compact multi-layer perceptron as the shared base learner. The aim is not to maximize absolute accuracy but to isolate the effect of the aggregation rule under DP noise. A small, fast learner also keeps the 510-run experimental grid tractable while remaining representative of the models banks actually deploy at the transaction-scoring stage. Whether our conclusions transfer to larger or sequential backbones is taken up in Section III.

2. FL for credit card fraud detection

Dang and Ha (2024) reached 97% accuracy with FedAvg and an MLP backbone [13]. Aurna et al. (2023) evaluated FedAvg with CNN, MLP, and LSTM backbones and several sampling methods [14]. Baabdullah et al. (2024) combined FL with a blockchain ledger for auditable aggregation [15], and blockchain has also been applied to secure card data at rest [16]. Alaklabi (2026) extends this line with PrivChain-AI, which layers differential privacy, homomorphic encryption, and smart-contract governance over federated training and reports 94.7% fraud-recognition accuracy at ε = 1.0 [17]. That result illustrates two things at once: the field is converging on DP as the privacy mechanism of choice for financial FL, and single-digit ε budgets are the level at which deployed systems aim to operate. None of these studies, however, vary the aggregation rule under DP, so they cannot say whether the rule matters once noise is present.

3. Aggregation strategies

Several alternatives to plain FedAvg target non-IID client data. Cosine-similarity aggregation [12] weights each client by the angle between its update and the consensus direction; FedAvg-DWA [18] uses inverse Euclidean distance; FedAdam [19] applies server-side Adam momentum; FedProx [20] adds a proximal term to the local objective. Nergiz (2023) compared aggregation algorithms, including momentum variants, and found measurable differences across heterogeneity levels, but in the non-private setting [21]. Comparisons within FL-CCFD typically rely on a single seed and a single benchmark and omit differential privacy entirely [13], [18], which leaves their rankings vulnerable to partition-draw luck and says nothing about the private regime.

4. Class imbalance and differential privacy

Extreme class imbalance is the defining difficulty of payment-fraud data: fraudulent transactions are typically well below 1% of observations [22], [23]. Resampling methods such as SMOTE [24], [25] remain the standard mitigation. DP-SGD makes the problem harder in a specific way: per-sample clipping and Gaussian noise suppress the gradient contribution of rare minority examples more than that of the majority class, so the fraud signal is exactly the part of the gradient most at risk of being drowned out. Farooq et al. (2025) respond by adapting clipping bounds and noise scales to protect informative minority gradients in federated fraud detection [26]. Our results bear on the same tension from a different direction: DP noise damages the model's calibration far more than its ranking ability. That finding moves the practical remedy from the gradient level to the decision-threshold level.

5. Differential privacy in FL

The Rényi DP accountant of Mironov [27] provides tight cumulative budget estimates, the Opacus library [28] implements DP-SGD for PyTorch, and McMahan et al. [29] formalized user-level DP-FedAvg. Record-level DP-SGD protects individual transactions against an honest-but-curious observer of the model updates; it does not, on its own, hide client updates from the coordinating server. Concealing updates from the server requires secure aggregation [30], which is orthogonal to DP and outside our experimental scope; Section II states the threat model this implies. Within that model, the interaction between the DP mechanism and the choice of aggregation rule in fraud detection has not been studied.

B. Contributions

Three gaps follow from the literature above: aggregation-rule comparisons in FL-CCFD are single-seed and single-benchmark, they exclude differential privacy, and when a configuration underperforms the cause is rarely diagnosed. This paper addresses all three. We run 510 experiments across five aggregation configurations, two benchmarks, two heterogeneity levels, and four DP-SGD noise multipliers, making this the first multi-seed, multi-dataset comparison of FL aggregation strategies in the differentially private CCFD setting. The results show that DP noise does not primarily degrade the model's ranking ability (AUC stays near 0.95); it drifts the model's calibration, collapsing F_1 at the fixed 0.5 threshold while leaving recall largely intact. A validation-tuned threshold recovers most of this loss at zero additional privacy cost. We also identify, verify at the level of global model weights, and correct a previously undocumented interaction: when FedProx is combined with Opacus using the standard loss-augmented recipe, the proximal term is silently overwritten before each DP-SGD step, reducing FedProx numerically to FedAvg.

The remainder of the paper is organized as follows. Section II presents the datasets, preprocessing, and the components of the experimental method. Section III reports and discusses the results in five phases, including the privacy sweep and the cross-dataset replication. Section IV concludes.

II. METHOD

The method has eight components, each described in a dedicated subsection: the two benchmark datasets (A) and their preprocessing (B), the base learner shared across all configurations (C), the federated training loop (D), the five aggregation strategies (E), the DP-SGD privacy mechanism and its threat model (F), the non-IID Dirichlet partitioning scheme with its realized per-client statistics (G), and the evaluation protocol including the two-threshold reporting design and the multiple-comparison policy (H). Two public benchmarks are used deliberately; findings that remain robust in the face of a 20× increase in fraud prevalence are less likely to be artifacts of the class imbalance of a single dataset.

A. Datasets

Table 1 summarizes both benchmarks. The Kaggle ULB creditcard dataset [31] contains 284807 transactions (492 fraudulent, 0.17%) from European cardholders in September 2013. Features V1–V28 are PCA-transformed for privacy; only Time and Amount remain raw. The IEEE-CIS dataset [32], contributed by Vesta Corporation to a 2019 Kaggle competition, contains 590540 transactions (20663 fraudulent, 3.50%) with 433 features spanning numeric, categorical, and obfuscated Vesta-engineered fields.

B. Preprocessing

Kaggle ULB needs only a StandardScaler fitted on the training fold. IEEE-CIS goes through a three-step pipeline. After merging the transaction and identity tables and dropping the identifier and timestamp columns, 431 candidate features remain; 9 columns exceed the 95% missingness threshold and are dropped, leaving 422 features (380 numeric, 42 categorical). Categorical columns are label-encoded against a training-fold vocabulary only, with cardinalities ranging from 3 to 12345 categories (the largest being card identifiers); numeric columns are median-imputed and scaled. Missingness in the retained columns is substantial before imputation—43% on average for numeric and 55% for categorical fields, driven largely by identity-table columns absent for the roughly 76% of transactions without an identity record—which is why the conservative impute-and-flag recipe matters for this dataset. The recipe avoids per-method choices that would complicate the cross-dataset comparison.

Splits are stratified 80/20 (test) and 90/10 (train/val), preserving class proportions throughout. Class imbalance is a well-documented challenge in payment fraud datasets [22], [23]. SMOTE [24], [25] (strategy 0.5; minority oversampled to 50% of the local majority) runs after Dirichlet partitioning, independently on each client's shard. GAN-based oversampling is another option [33], [34]; we opted for SMOTE because its behavior under Dirichlet partitioning is more predictable and it introduces fewer additional hyperparameters.

Applying SMOTE per client under Dirichlet heterogeneity raises a known failure mode: interpolating synthetic minority points from a very sparse neighborhood produces noise rather than signal. Our implementation guards the degenerate cases—oversampling is skipped when a shard holds fewer than 3 minority samples, and k_neighbors adapts downward to n_minority − 1—but the guard does not make sparse interpolation informative, it only makes it defined. Table 2 quantifies how often the regime occurs: even at α = 0.5, 13 of 50 ULB shards hold fewer than 10 fraud cases, and the most extreme shard in the whole grid synthesized 19766 points per real fraud case (an IEEE-CIS shard holding 6 fraud cases among 237213 transactions received 118598 synthetic ones); the ULB worst case amplified 4 real fraud cases into 62850 synthetic ones. At α = 0.1, SMOTE is skipped outright on 17 of 50 ULB shards, and 32 of the 200 shards overall invert past 50% fraud—18 of 50 on IEEE-CIS at α = 0.1—so SMOTE oversamples the legitimate class instead. We keep SMOTE for comparability with the FL-CCFD literature and treat these statistics as part of the explanation for the Phase 4 variance analyzed in Section III.

Table 1. Benchmark dataset statistics.

	Kaggle ULB	IEEE − CIS
Transactions	284807	590540
Fraud cases	492	20663
Positive rate	0.17%	3.50%
Raw features	30	433
After preprocessing	30	380num.+42 cat
Feature type	Numeric (PCA)	Mixed
Time span	2 days, Sept 2013	6 months, 2017

C. Base learner

The base learner is a three-layer MLP: hidden widths {64, 32, 16}, ReLU activations, LayerNorm after each hidden layer, 0.2 dropout, and a sigmoid output. LayerNorm is required because Opacus cannot handle BatchNorm—the running statistics couple samples within a batch, breaking per-sample gradient isolation. Training uses binary cross-entropy with Adam (lr = 10^{-3}, weight decay 10^{-4}) and batch size 256 in PyTorch [35]. The positive-class weight is w^{+}_{i} = n_neg,i / n_pos,i, computed from each client's post-SMOTE shard rather than from a global count.

D. Federated training loop

Each experiment runs K = 5 clients for T = 10 rounds. At round t, the server sends global parameters $w^{(t)}$ to every client. Each client trains for E = 2 local epochs and returns $w_i^{(t+1)}$; the server aggregates and broadcasts the new global. Clients run sequentially on a single GPU via a Flower [36] in-process simulator. Every fit/evaluate call sits inside a try/except boundary, and all strategies set accept_failures = True explicitly.

E. Aggregation strategies

Five configurations are compared, covering three distinct design axes found in the FL-CCFD literature. FedAvg [4] serves as the sample-weighted baseline. Cosine-similarity aggregation [12] and FedAvg-DWA [18] represent the geometry-based reweighting family. FedAdam [19] represents adaptive server-side optimization. FedProx [20] completes the comparison as the most-cited local-objective alternative.

1. FedAvg

Plain FedAvg [4] computes the sample-weighted mean by using (1):

$$w^{(t+1)} = \sum_i [n_i / \sum_j n_j]\, w_i^{(t+1)}. \quad (1)$$

2. Cosine-similarity aggregation

Per-client update deltas $\delta_i = w_i^{(t+1)} - w^{(t)}$ are compared to the sample-weighted mean delta $\bar{\delta}$. The aggregation weight combines sample size with non-negative cosine similarity $\alpha_i = \cos(\delta_i, \bar{\delta})$ as in (2):

$$wa_i = n_i \max(\alpha_i, 0)\,, w^{(t+1)}$$
$$= \sum_i a_i w_i^{(t+1)} / \sum_i a_i. \quad (2)$$

When all α_i = 1 (every client moves in the consensus direction), (2) reduces exactly to (1).

Table 3 outlines the step-by-step procedure for the cosine-similarity federated aggregation evaluated in the experiments.

3. FedAvg-DWA

FedAvg-DWA [18] replaces the cosine with inverse Euclidean distance $d_i = \|\delta_i\|_2$ as demonstrated in (3):

$$\beta_i = n_i/(d_i + \varepsilon_{dwa})\,,\ \varepsilon_{dwa} = 10^{-8}. \quad (3)$$

Clients that have drifted far from $w^{(t)}$ are down-weighted.

Table 2. Realized per-client partition statistics across 200 client-shards (10 seeds, K=5). Columns: fraud-rate min/median/max; shards below 1% and above 50% fraud; shards with <50 minority samples; shards where SMOTE's degeneracy guard skipped oversampling; maximum synthetic-to-real-minority ratio.

Dataset	α	Fraud rate (%), min/med/max	$< 1\%$	$> 50\%$	Min. < 50	SMOTE skip	Max syn:real
ULB	0.5	0.002 / 0.12 / 77.70	38	2	27	2	15713
ULB	0.1	0.001 / 0.19 / 95.20	37	6	36	17	2374
IEEE-CIS	0.5	0.003 / 5.43 / 99.20	16	6	4	0	19766
IEEE-CIS	0.1	0.002 / 3.30 / 99.99	19	18	21	7	18444

Table 3. Cosine-similarity federated aggregation.

	Input: $\{w_i^{(t+1)}\}, \{n_i\}, w^{(t)}$; **Output:** aggregated $w^{(t+1)}$
1	Compute $\delta_i \leftarrow w_i^{(t+1)} - w^{(t)}$ for all i
2	$N \leftarrow \sum_i n_i$; $\bar{\delta} \leftarrow (1/N) \sum_i n_i \delta_i$
3	For each i: $\alpha_i \leftarrow \delta_i \bar{\delta} / \left(\|\delta_i\|_2 \|\bar{\delta}\|_2 \right)$
4	$a_i \leftarrow n_i \max(\alpha_i, 0)$; $A \leftarrow \sum_i a_i$
5	if $A \leq 0$: return $(1/N) \sum_i n_i w_i^{(t+1)}$ [FedAvg fallback]
6	return $(1/A) \sum_i a_i w_i^{(t+1)}$

4. FedAdam

FedAdam [19] applies a server-side Adam step to the average update $\bar{\delta}^{(t)}$, as shown in (4)-(6):

$$m^{(t+1)} = \beta_1 m^{(t)} + (1 - \beta_1)\bar{\delta}^{(t)}, \tag{4}$$

$$v^{(t+1)} = \beta_2 v^{(t)} + (1 - \beta_2)\left[\bar{\delta}^{(t)}\right]^2, \tag{5}$$

$$w^{(t+1)} = w^{(t)} + \eta_s m^{(t+1)} / \sqrt{v^{(t+1)}} + \tau, \tag{6}$$

where hyperparameters are $\eta_s = 10^{-2}$, $\beta_1 = 0.9$, $\beta_2 = 0.99$, and $\tau = 10^{-3}$. Section III reports a sensitivity analysis over $\eta_s \in \{10^{-3}, 10^{-2}, 10^{-1}\}$ and $\tau \in \{10^{-3}, 10^{-2}\}$, and a rounds ablation at $T \in \{10, 25, 50\}$.

5. FedProx and the Opacus interaction

FedProx [20] keeps the FedAvg aggregation rule but adds a proximal term to each client's local objective, as defined in (7):

$$\min_{w_i} L_i(w_i) + (\mu/2)\left\|w_i - w^{(t)}\right\|_2^2 \tag{7}$$

Under DP-SGD as implemented in Opacus, the loss-augmented proximal term is silently neutralized. Opacus captures per-sample gradients via forward-pass hooks; the proximal gradient $\mu(w^{(t)} - w_i)$, computed outside the forward pass, is overwritten before the DP step is taken, so FedProx collapses numerically to FedAvg. To preserve FedProx semantics under DP, the proximal correction is applied as a deterministic post-step update in (8):

$$w_i \leftarrow w_i + \eta\mu(w^{(t)} - w_i) \tag{8}$$

where η is the local learning rate. The correction (8) is a deterministic function of public information and adds no privacy cost. Papers that follow the standard loss-augmented FedProx recipe with Opacus are numerically running plain FedAvg, whether they intend to or not. We verify this claim at the level of the global model weights rather than inferring it from privacy accounting alone. Three runs share one seed and therefore one initialization, one data order, one dropout-mask sequence, and one Gaussian noise stream: FedAvg, loss-augmented FedProx under Opacus, and the corrected post-step FedProx of (8). Since the proximal computation consumes no randomness and execution is deterministic on CPU, any weight difference between the first two runs could only come from the proximal gradient itself. The recorded global weight trajectories ($\sigma = 1.0$, $\alpha = 0.5$, $\mu = 0.01$, three partition draws) show that loss-augmented FedProx is bit-identical to FedAvg at every one of the eleven checkpoints (initialization and ten rounds) in every draw: the maximum per-round L_∞ distance is exactly zero, and the two runs produce identical test metrics. The corrected variant diverges from FedAvg from the first round (maximum L_∞ of 1.4–3.3 × 10^{-3}, four orders of magnitude above float32 resolution), and a control pair with DP disabled diverges by up to 0.85 in L_∞ while improving fixed-threshold F_1 by about 0.02, confirming that the loss-based proximal term acts, and helps, when Opacus is absent. Fig. 1 summarizes the three trajectories. Notably, the per-round training loss of the loss-augmented variant differs from FedAvg—the proximal penalty is computed and enters the reported loss—while the weights do not: the penalty's gradient is overwritten before the optimizer step and never reaches the model.

F. Differential privacy via DP-SGD

DP-SGD [5], [6] clips each per-sample gradient g_i to ℓ_2-norm $C = 1.0$, as in (9):

$$\tilde{g}_i = g_i \min(1, C / \|g_i\|_2), \tag{9}$$

then adds Gaussian noise as in (10):

$$\beta\hat{g} = (1/B)\left(\sum_i \tilde{g}_i + N(0, \sigma^2 C^2 I)\right). \tag{10}$$

Four noise multipliers $\sigma \in \{0.5, 1.0, 1.5, 2.0\}$ are swept. Cumulative $(\varepsilon, \delta = 10^{-5})$ is tracked by Opacus's RDP accountant, which persisted across federated rounds. The range is chosen to trace the transition from nominal to deployment-relevant protection rather than as an exploratory sweep: $\sigma = 0.5$ accumulates $\varepsilon \approx 46.9$ on ULB, which offers negligible practical protection by any established standard and serves as a no-privacy anchor for the degradation curve, while $\sigma = 2.0$ reaches $\varepsilon \approx 4.2$, approaching the $\varepsilon \leq 2$–8 range that deployed systems target—PrivChain-AI, for instance, reports operating at $\varepsilon = 1.0$

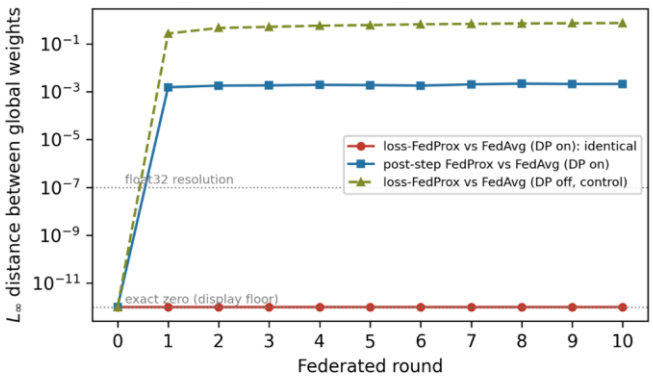


Fig. 1. Per-round L_∞ distance between the global weights of each FedProx variant and a FedAvg run sharing the same seed, data order, and noise stream (Kaggle ULB, $\alpha = 0.5$, $\sigma = 1.0$, $\mu = 0.01$, 3-seed mean, log scale).

[17]. We describe the reported budgets in these practical terms throughout and avoid presenting large-ε configurations as meaningfully private.

The threat model is as follows. The coordinating server is honest-but-curious, and record-level DP-SGD protects individual transactions against any observer of the shared model updates or the final model, including the server. It does not, on its own, conceal a client's updates from the server; that requires secure aggregation [30], which is orthogonal to the record-level mechanism and outside our experimental scope. All privacy statements in this paper are therefore per-client, record-level statements under a trusted-aggregation assumption.

The interaction between local oversampling and privacy accounting also deserves an explicit statement. Opacus accounts at the level of records in the post-SMOTE training set, and each synthetic sample is a deterministic interpolation of real minority records. A single real fraud case therefore influences many training records—in the most amplified shard we observed, six real fraud cases underlie 118598 synthetic training records—and by the group-privacy property the effective privacy loss for that original record exceeds the reported per-record ε. The ε values in Tables 7 and 9 are thus exact for post-SMOTE training records but constitute lower bounds for the original minority records that SMOTE amplified. A per-client accounting that composes SMOTE's amplification factor into the budget is, to our knowledge, an open problem, and we flag it as such rather than claiming the reported budgets transfer unchanged to the raw data. Majority-class records, which SMOTE never duplicates, are unaffected.

G. Non-IID partitioning

Dirichlet partitioning [37] with concentration parameter α controls data heterogeneity; we test α = 0.5 (mild) and α = 0.1 (strong). Because the realized partitions—not the α value—determine what each client actually trains on, Table 2 and Fig. 2 characterize all 200 client-shards the experiments used (10 seeds × 5 clients × 2 datasets × 2 α levels). Even at α = 0.5, the per-client fraud rate on Kaggle ULB spans 0.0015% to 77.7% (median 0.12%), 38 of 50 shards fall below 1% fraud, and 13 hold fewer than 10 fraud cases. At α = 0.1 the extremes widen: individual shards reach 95.2% fraud on ULB and 99.99% on IEEE-CIS, and on IEEE-CIS 18 of 50 shards invert past 50% fraud, so the minority class is the legitimate one. These realized distributions matter for two results reported later: they set the conditions under which per-client SMOTE operates (Section II.B), and they are the direct source of the partition-draw variance analyzed in Phase 4.

H. Evaluation and multi-seed protocol

Seven metrics are reported: accuracy, precision, recall, F_1, AUC, PR-AUC, and MCC. Recall emphasizes that fraud typically costs more than a false alarm. Two operating thresholds are used: the conventional 0.5 sigmoid cutoff and a validation-tuned F_1-maximizing threshold applied unchanged to the test fold (no test leakage). Each configuration is repeated across ten seeds; pairwise comparisons use the paired Wilcoxon signed-rank test with seed as the pairing axis. All tests are corrected within their phase family; for the privacy sweep, which involves C(5, 2) = 10 pairs across four σ levels, we pool the forty tests into one family—the conservative reading of one sweep, one question—and report Bonferroni-, Holm-, and Benjamini–Hochberg-adjusted p-values for every test in Appendix Tables A1 and A2, together with the per-noise-level (ten-test) Holm adjustment for the sweep, since the family choice matters at the margin. Table 4 presents all hyperparameters.

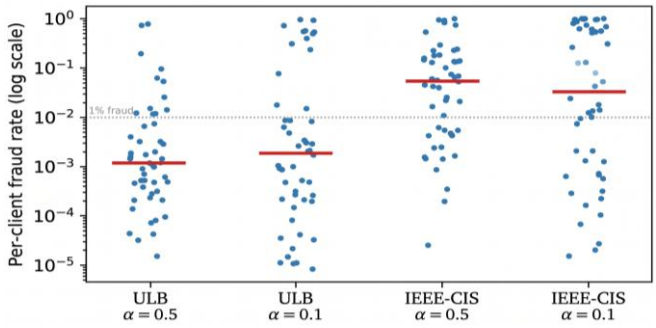


Fig. 2. Per-client fraud rates of 200 Dirichlet shards (log scale; red bar = median). At α = 0.1, extreme heterogeneity causes some shards to exceed 50% fraud, making the legitimate class the local minority.

III. RESULTS

In total, 510 runs are distributed across five phases: Phase 1a (centralized baseline, 10 runs); Phase 1b (federated IID, 50 runs); Phase 2 (α = 0.5, no DP, 50 runs); Phase 3 (DP σ sweep, 200 runs); Phase 4 (α = 0.1, no DP, 50 runs); and Phase 5 (IEEE-CIS replication, 150 runs). For each metric, we report the mean ± standard deviation over 10 random seeds and conduct pairwise comparisons using the paired Wilcoxon signed-rank test with the seed serving as the pairing dimension. Corrected p-values for every test appear in Appendix Tables A1 and A2.

A. Evaluation and multi-seed protocol

Table 5 contrasts centralized training (K = 1) with all five federated setups on IID splits. Centralized FedAvg attains F_1@best-t = 0.799 ± 0.036, while the five federated variants achieve values between 0.801 and 0.804. No pairwise Wilcoxon test yields p < 0.05. In this setting, federation comes at no performance cost.

B. Phase 2: Aggregation comparison without DP (α = 0.5)

Table 6 shows all five configurations bunched within a 0.014 spread on F_1@best-t—from 0.781 ± 0.030 (Cosine) to 0.795 ± 0.022 (FedAvg-DWA). Two pairs show nominally significant raw p-values—FedAvg-DWA vs. Cosine (p = 0.049, W = 8) and FedAvg-DWA vs. FedAdam (p = 0.037, W = 7)—but neither survives Bonferroni correction for ten simultaneous pairwise comparisons (corrected p = 0.49 and p = 0.37 respectively; Appendix Table A1). Phase 2 consequently yields no statistically significant pairwise differences once the family-wise error rate is controlled.

Table 4. Hyperparameter settings (uniform across all experiments).

Component	Parameter	Value
MLP	Hidden widths	(64, 32, 16)
	Activation	ReLU/LayerNorm/Dropout 0.2
	Optimizer	Adam, lr 10^{-3}, wd 10^{-4}
	Batch size	256
FL	K / E / T	5 / 2 / 10
SMOTE	Strategy	0.5, per client
DP-SGD	C / σ	1.0 / {0.5, 1, 1.5, 2}
	Δ	10^{-5}
FedProx	M	0.01
DWA	ε_{dwa}	10^{-8}
FedAdam	η_s/ β_1 / β_2 / τ	10^{-2} / 0.9 / 0.99 / 10^{-3}
Dirichlet	A	0.5 (mild) / 0.1 (strong)
Seeds	Per config.	10

Table 5. Phase 1 — Federation cost under IID. $K = 5, T = 10$, no DP. Mean ± std, 10 seeds.

Setting	Strategy	F_1@best - t	MCC†
Centralized	FedAvg	0.799 ± 0.036	0.747 ± 0.028
Federated IID	FedAvg	0.804 ± 0.026	0.771 ± 0.032
	FedProx	0.804 ± 0.036	0.765 ± 0.037
	Cosine	0.802 ± 0.028	0.767 ± 0.030
	FedAvg-DWA	0.801 ± 0.030	0.771 ± 0.029
	FedAdam	0.801 ± 0.027	0.717 ± 0.029

† MCC evaluated at the fixed 0.5 threshold.

Table 6. Phase 2 — Aggregation comparison, α = 0.5, no DP. Mean ± std, 10 seeds. bt = best-t; MCC at 0.5 threshold: FA 0.755, FP 0.776, CO 0.756, DWA 0.771, FAd 0.746.

Strategy	AUC	PR − AUC	F_1@0.5	F_1@bt
FedAvg	0.955 ± 0.040	0.742 ± 0.053	0.749 ± 0.064	0.786 ± 0.028
FedProx	0.958 ± 0.029	0.713 ± 0.071	0.774 ± 0.050	0.791 ± 0.028
Cosine	0.950 ± 0.033	0.752 ± 0.054	0.751 ± 0.065	0.781 ± 0.030
FedAvg-DWA	0.954 ± 0.035	0.731 ± 0.067	0.767 ± 0.062	0.795 ± 0.022
FedAdam	0.933 ± 0.023	0.708 ± 0.065	0.738 ± 0.102	0.783 ± 0.032

C. Phase 3: The leveling effect of DP-SGD

The central experiment sweeps $\sigma \in \{0.5, 1.0, 1.5, 2.0\}$ across all five configurations at α = 0.5, producing 200 runs. Table 7 reveals a consistent result across every noise level. AUC holds near 0.95 regardless of σ. What breaks is calibration: F_1@0.5 falls from roughly 0.75 to 0.48 as σ rises from none to 2.0, because precision collapses while recall is largely preserved. Switching to the validation-tuned threshold brings F_1 back to about 0.79 at every noise level, at no privacy cost.

Seven of the forty raw Wilcoxon p-values fall below 0.05, concentrated at σ = 1.0 and σ = 1.5. At σ = 1.0, Cosine vs. FedAdam (p = 0.039, W = 3) and FedAvg-DWA vs. FedAdam (p = 0.037, W = 7) are nominally significant. At σ = 1.5, five pairs flag: FedAvg-DWA vs. FedAdam (p = 0.004, W = 1, the strongest signal in the experiment), FedAvg vs. FedAvg-DWA (p = 0.023, W = 1.5), FedProx vs. FedAvg-DWA (p = 0.023, W = 1.5), FedAvg vs. FedAdam (p = 0.049, W = 8), and FedProx vs. FedAdam (p = 0.049, W = 8). None survives correction within the pooled forty-test family: the minimum corrected p is 0.156 under Bonferroni, Holm, and Benjamini–Hochberg alike (Appendix Table A1). The family choice matters at the margin, and we state it explicitly: under the per-noise-level alternative (ten tests per σ), the strongest pair, FedAvg-DWA vs. FedAdam at σ = 1.5, survives Holm correction at adjusted p = 0.039. We adopt the pooled family as primary and treat that pair as suggestive rather than established. Under the global null, roughly two of forty tests would fall below 0.05; observing seven, concentrated at $\sigma \in \{1.0, 1.5\}$ with a consistent FedAvg-DWA-ahead, FedAdam-behind ordering, is the expected signature of real effects below this design's confirmation threshold, a point the power analysis in Section III.F.4 quantifies. FedAdam's lag is analyzed further in Section III.F.1 and Fig. 3. After correction, aggregation-rule selection on the ULB sweep is statistically indistinguishable across all noise levels tested.

D. Phase 4: Strong non-IID stress test (α = 0.1)

Table 8 presents α = 0.5 and α = 0.1 side by side. Under increased heterogeneity, while the mean F_1@best-t decreases only slightly, the results become far more volatile; the standard deviation for FedAvg rises nearly fourfold, from 0.028 to 0.106. The distribution is bimodal across seeds: certain Dirichlet draws collapse every configuration to near-zero, while others produce results indistinguishable from the α = 0.5 regime. Table 2 and Fig. 2 quantify the realized partitions behind this variance: at α = 0.1, 17 of 50 ULB shards skip SMOTE entirely, individual shards reach 95.2% fraud, and seeds whose draws contain degenerate shards are exactly the seeds whose F_1 collapses.

E. Phase 5: IEEE-CIS cross-dataset replication

Table 9 replicates the headline phases on IEEE-CIS. Three sub-phases cover: IID federation (K = 5, no DP); α = 0.5, no DP; and α = 0.5 with DP at σ = 1.0 (ε = 8.10 ± 5.30). Despite the 20× prevalence shift, the calibration finding transfers unchanged. The strategy comparison transfers with one important difference: effects are larger, and some now survive correction. Under IID federation, FedAdam sits clearly behind the field (0.497 ± 0.004 vs. 0.551–0.562 for the others) and seven of ten pairwise tests survive all three corrections (Appendix Table A2). At α = 0.5 without DP, one pair is nominally significant, and none survives correction. Under DP at σ = 1.0, eight of ten pairs are nominally significant and FedAvg-DWA vs. FedAdam survives both Bonferroni and Holm (adjusted p = 0.020, d = 2.4). The ordering matches the ULB nominal flags—FedAvg-DWA ahead, FedAdam

Table 7. Phase 3 — Privacy sweep, Kaggle ULB, α = 0.5. ε identical across configurations at each σ; ε values are per-training-record on the post-SMOTE shards (Section II.F). Mean ± std, 10 seeds.

Strategy	σ	ε	AUC	F_1@0.5	F_1@best - t
FedAvg	none	∞	0.955 ± 0.040	0.749 ± 0.064	0.786 ± 0.028
	0.5	46.88 ± 19.30	0.951 ± 0.026	0.641 ± 0.134	0.793 ± 0.029
	1.0	11.72 ± 7.75	0.953 ± 0.026	0.563 ± 0.162	0.787 ± 0.040
	1.5	6.22 ± 4.38	0.955 ± 0.024	0.517 ± 0.167	0.788 ± 0.033
	2.0	4.18 ± 2.96	0.958 ± 0.023	0.480 ± 0.170	0.789 ± 0.035
FedProx	none	∞	0.958 ± 0.029	0.774 ± 0.050	0.791 ± 0.028
	0.5	46.88 ± 19.30	0.951 ± 0.026	0.641 ± 0.134	0.793 ± 0.029
	1.0	11.72 ± 7.75	0.953 ± 0.026	0.562 ± 0.162	0.790 ± 0.033
	1.5	6.22 ± 4.38	0.955 ± 0.024	0.518 ± 0.167	0.787 ± 0.034
	2.0	4.18 ± 2.96	0.958 ± 0.023	0.481 ± 0.170	0.789 ± 0.035
Cosine	none	∞	0.950 ± 0.033	0.751 ± 0.065	0.781 ± 0.030
	0.5	46.88 ± 19.30	0.944 ± 0.035	0.653 ± 0.137	0.789 ± 0.040
	1.0	11.72 ± 7.75	0.947 ± 0.032	0.580 ± 0.164	0.790 ± 0.032
	1.5	6.22 ± 4.38	0.952 ± 0.025	0.534 ± 0.175	0.783 ± 0.038
	2.0	4.18 ± 2.96	0.955 ± 0.023	0.499 ± 0.176	0.784 ± 0.032
FedAvg-DWA	none	∞	0.954 ± 0.035	0.767 ± 0.062	0.795 ± 0.022
	0.5	46.88 ± 19.30	0.954 ± 0.025	0.635 ± 0.131	0.794 ± 0.035
	1.0	11.72 ± 7.75	0.953 ± 0.027	0.552 ± 0.155	0.794 ± 0.029
	1.5	6.22 ± 4.38	0.957 ± 0.025	0.506 ± 0.165	0.795 ± 0.031
	2.0	4.18 ± 2.96	0.958 ± 0.025	0.470 ± 0.171	0.792 ± 0.033
FedAdam	none	∞	0.933 ± 0.023	0.738 ± 0.102	0.783 ± 0.032
	0.5	46.88 ± 19.30	0.933 ± 0.027	0.658 ± 0.156	0.780 ± 0.047
	1.0	11.72 ± 7.75	0.937 ± 0.018	0.596 ± 0.178	0.770 ± 0.044
	1.5	6.22 ± 4.38	0.945 ± 0.020	0.555 ± 0.192	0.740 ± 0.091
	2.0	4.18 ± 2.96	0.946 ± 0.021	0.528 ± 0.198	0.728 ± 0.117

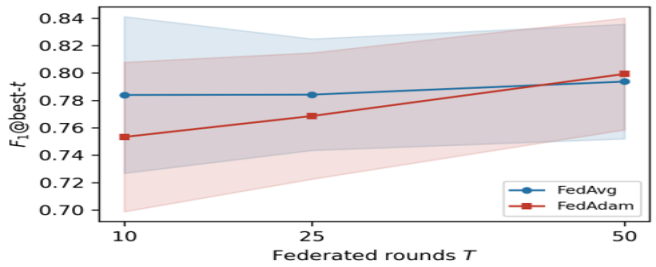


Fig. 3. FedAdam rounds ablation under DP (Kaggle ULB, α = 0.5, σ = 1.0, 5 seeds, mean ± std). At the default server learning rate, FedAdam's gap to FedAvg closes as rounds increase and vanishes by T = 50; cumulative ε grows with T at fixed σ (10.3, 17.6, 27.3), so comparisons are within each T level.

Table 8. Phase 4 — Strong non-IID stress test. F_1@best-t, no DP. Mean ± std, 10 seeds.

Strategy	α = 0.5	α = 0.1
FedAvg	0.786 ± 0.028	0.729 ± 0.106
FedProx	0.791 ± 0.028	0.759 ± 0.037
Cosine	0.781 ± 0.030	0.689 ± 0.236
FedAvg-DWA	0.795 ± 0.022	0.697 ± 0.233
FedAdam	0.783 ± 0.032	0.656 ± 0.270

Table 9. Phase 5 — IEEE-CIS cross-dataset replication. F_1@best-t, mean ± std, 10 seeds. Phase 5d: σ = 1.0, ε = 8.10 ± 5.30.

Strategy	5b: IID	5c: $\alpha = 0.5$	5d: DP
FedAvg	0.561 ± 0.005	0.406 ± 0.113	0.395 ± 0.047
FedProx	0.551 ± 0.006	0.429 ± 0.079	0.396 ± 0.048
Cosine	0.561 ± 0.006	0.393 ± 0.115	0.369 ± 0.076
FedAvg-DWA	0.562 ± 0.007	0.421 ± 0.091	0.409 ± 0.033
FedAdam	0.497 ± 0.004	0.407 ± 0.104	0.361 ± 0.026

behind—at magnitudes large enough to confirm. Aggregation choice under DP is therefore not irrelevant: its effect is dataset-dependent in size and consistent in direction.

IV. DISCUSSION

Four observations emerge from the results: why DP noise shrinks inter-configuration differences and what remains of them, why threshold calibration matters more than aggregation-rule choice, why strong heterogeneity makes comparisons unreliable, and what the principal threats to validity are.

A. Aggregation differences under DP-SGD: small on ULB, measurable on IEEE-CIS

In the non-private setting, the five strategies all act on something real: the geometry of client update vectors δ_i. Under DP-SGD, Gaussian noise at scale $C\sigma$ is added to every per-sample gradient before the local optimizer touches it. At $\sigma \geq 1$, this noise dominates the signal: cosine and DWA end up weighting noise geometry, FedAdam accumulates noise-corrupted pseudo-gradients, and FedProx's proximal pull becomes irrelevant. On the ULB sweep this shows up as seven nominal flags that do not survive family-wise correction; on IEEE-CIS, where the fraud signal per client is stronger, the same ordering produces effects large enough to survive it. The noise does not erase the differences between rules so much as compress them below what a ten-seed design can confirm on a hard, low-prevalence benchmark.

FedAdam's lag at T = 10 invited two explanations—insufficient tuning or insufficient rounds for server-side momentum to accumulate—and a dedicated ablation (Fig. 3) shows both contribute. First, the server learning rate matters far more under DP than its default suggests: at T = 10, $\eta_s = 10^{-3}$ collapses entirely (F_1@best - t of 0.26 and 0.06 depending on τ), the default $\eta_s = 10^{-2}$ reaches 0.753 ± 0.055, and $\eta_s = 10^{-1}$ reaches 0.786 ± 0.050, statistically indistinguishable from FedAvg's 0.784 ± 0.057 on the same seeds. A larger server step compensates for the short horizon. Second, at the default learning rate the gap closes with rounds exactly as the momentum explanation predicts: the FedAvg−FedAdam difference in mean F_1@best - t shrinks from +0.031 at T = 10 to +0.016 at T = 25 and −0.006 at T = 50. None of these per-round gaps are individually significant (paired Wilcoxon $p \geq 0.31$ at n = 5 seeds, where the minimum attainable p is 0.0625), and per-seed differences at T = 10 span −0.04 to +0.12, the same partition-draw variance documented in Phase 4. Two caveats bound the claim. The cumulative privacy budget grows with rounds at fixed σ (mean ε of 10.3, 17.6, and 27.3 at T = 10, 25, 50), so the ablation compares the FedAdam−FedAvg gap within each T level, where accounting is identical, rather than treating extra rounds as free. And in no configuration, tuned or extended, does FedAdam surpass FedAvg: adaptive server optimization recovers parity under DP noise, not an advantage, which leaves the practical recommendation unchanged. Phase 2/3 tables report the Reddi et al. [19] default hyperparameters, as stated in Section II.E.

B. Threshold calibration is the key factor to consider when working with differential privacy

AUC barely moves as σ increases. What degrades is calibration: the sigmoid output shifts so that the natural operating point no longer sits near 0.5. Precision collapses at the fixed threshold while recall holds, dragging F_1@0.5 down sharply. A threshold selected by maximizing F_1 on the validation set recovers most of that loss—for example, at σ = 1.0, FedAvg goes from F_1 = 0.563 ± 0.162 at 0.5 to F_1 = 0.787 ± 0.040 at the tuned threshold, with no additional privacy cost.

C. Partition variance dominates under strong heterogeneity

At α = 0.1, the seed-to-seed variance is so large that the performance distribution is bimodal. Table 2 shows why: the realized shards range from essentially fraud-free to fraud-majority, SMOTE is skipped or degenerate on a third of them, and which regime a seed lands in is decided by the partition draw. Papers that report a single seed in this regime do not measure which aggregation strategy is better—they are sampling the partition-draw lottery. Ten seeds are the minimum we would trust for the questions asked here.

D. Threats to validity

The Kaggle benchmark is over a decade old; its PCA-transformed features hide temporal transaction patterns. The IEEE-CIS replication partially addresses this by confirming the calibration findings on more recent data.

Statistical power. With n = 10 seeds, a two-sided paired Wilcoxon test at α = 0.05 reliably detects (80% power) standardized paired effects of $d \approx 1.0$, rising to $d \approx 1.5$ under the Bonferroni correction we apply. The effects actually present under DP are smaller: across the Phase 3 sweep the median observed $|d|$ ranges from 0.19 (σ = 0.5) to 0.60 (σ = 1.5), and the strongest flagged pair (FedAvg-DWA vs. FedAdam at σ = 1.5, raw p = 0.004) has d = 0.68. In metric units, the design's detection floor at σ = 1.0 is roughly 0.019 F_1 points at the tuned threshold; the between-strategy gaps we measure under DP on ULB sit below it. The study is therefore adequately powered for its main conclusion—no large aggregation-strategy differences survive under DP on ULB—but not for adjudicating the small effects the nominal flags hint at: doubling to n = 20 would lower the raw-α detection floor to exactly the size of the strongest observed effect ($d \approx 0.68$) while leaving it out of reach after correction, and confirming an effect of that size at corrected α with adequate power would require roughly 35 seeds. We report the seven nominal flags with this framing rather than claiming either their reality or their absence; the IEEE-CIS result, where the same ordering exceeds the detection floor and survives correction, is consistent with it.

Model architecture. All experiments share one compact MLP backbone. This is a deliberate control—varying the aggregation

rule while holding the learner fixed isolates the quantity under study, and a small learner keeps the 510-run grid tractable—but it bounds the claims: we have not shown that the rankings, or the calibration-drift finding, transfer to larger, sequential, or graph-based learners. Two considerations suggest the core mechanism is not architecture-specific: the drift originates in DP-SGD's per-sample clipping and noise, which perturb the decision boundary of any gradient-trained classifier, and both benchmarks are tabular (ULB's features are anonymized PCA components), leaving little for sequence or graph models to exploit here in any case. Verifying transfer on richer backbones remains future work.

On validation privacy: the threshold-calibration step uses held-out validation labels that never enter gradient computation; under the per-client record-level threat model this is a standard assumption. Finally, the trusted-server assumption stated in Section II.F means our DP guarantees are per-client record-level only; protecting against a curious server would require secure aggregation [37].

V. CONCLUSION

We compared five FL aggregation strategies for credit card fraud detection under per-client differential privacy, across 510 experiments, two benchmarks, four noise levels, and ten seeds per configuration. The clearest result is also the most practically useful one: under DP-SGD, differences between aggregation rules become small and hard to confirm. In the non-private regime, FedAvg-DWA edges ahead of the field at mild heterogeneity (α = 0.5), but the margin is narrow (0.014 on F_1@best-t) and does not survive correction. On the ULB privacy sweep, none of the forty pairwise Wilcoxon tests survives family-wise correction; seven are nominally significant with a consistent FedAvg-DWA-ahead, FedAdam-behind ordering, and a power analysis shows effects of the observed size ($d \approx 0.4$–0.7) sit below what a ten-seed design can confirm. On IEEE-CIS, with 20× higher fraud prevalence, the same ordering appears at larger magnitude, and its extreme pair survives Holm correction under DP. Aggregation choice under DP is dataset-dependent in size and consistent in direction—but in every regime we tested, it matters less than what comes next.

What DP reliably degrades is calibration. AUC stays near 0.95 even at σ = 2.0, but precision collapses at the fixed 0.5 threshold, pulling F_1@0.5 down from ≈0.75 to ≈0.48. A validation-tuned threshold recovers most of that loss at no additional privacy budget. The practical workflow is straightforward: choose the aggregation rule based on non-private performance or robustness requirements, train with DP-SGD at the required ε, then calibrate the threshold on a held-out validation set.

A secondary finding worth flagging for reproducibility: the standard loss-augmented FedProx recipe is silently neutralized when combined with Opacus, because the DP step overwrites the proximal gradient before it can act. We verify this at the weight level: under identical seeds and noise streams, the loss-augmented variant produces global models bit-identical to FedAvg at every round across three partition draws, while the corrected post-step formulation diverges from the first round. Any paper using FedProx with the Opacus loss-based API under DP-SGD is running plain FedAvg. The corrected formulation preserves FedProx behavior at no extra privacy cost. Future research directions include adding secure aggregation for the server setting, testing sequence-based fraud encoders that leverage temporal features, and seed budgets near 35 to resolve the small strategy effects that this design can only flag.

AUTHOR STATEMENT

Plagiarism Check—The article has been scanned with iThenticate and found to be compliant with the journal's plagiarism policy.
Conflict of Interest—There is no conflict of interest with any person/organization.
Ethics Committee Approval—Ethics committee approval is not required for this article.
Use of Artificial Intelligence Tools— In this study, Grammarly was used for grammar and spell checking. All content reflects the original contribution of the authors
Funding—No institutional/financial support was received for this study.
Data availability— Both benchmark datasets are publicly available on Kaggle [31], [32]. The per-run experimental results, the per-client partition statistics, the weight-trajectory validation data, the corrected pairwise test tables, and the analysis scripts are available from the corresponding author on reasonable request.
CRediT Author Contribution— Conceptualization, Methodology, Software, Writing - Original Draft, Formal analysis, Investigation, Data Curation (**Murat Saran**); Methodology, Writing - Original Draft, Resources, Data Curation, Visualization, Validation, Review & Editing (**Abdül Kadir Görür**).
Acknowledgment—The authors thank the anonymous reviewers for their constructive comments/improvements to the manuscript.

REFERENCES

[1] E. Ileberi, Y. Sun, Z. Wang, "Performance evaluation of machine learning methods for credit card fraud detection using SMOTE and AdaBoost", *IEEE Access*, 9, 165286–165294, 2021. https://doi.org/10.1109/ACCESS.2021.3134330.

[2] A. Ali, S. A. Razak, S. H. Othman, T. A. E. Eisa, A. Al-Dhaqm, M. Nasser, T. Elhassan, H. Elshafie, A. Saif, "Financial fraud detection based on machine learning: A systematic literature review", *Applied Sciences*, 12(19), 9637, 2022. https://doi.org/10.3390/app12199637.

[3] European Parliament and Council of the European Union, "Regulation (EU) 2016/679 (General Data Protection Regulation)", *Official Journal of the European Union*, L119, 1–88, 2016. https://gdpr-info.eu/.

[4] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data", *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)*, Fort Lauderdale, FL, USA, 20-22 April 2017, 1273–1282. https://doi.org/10.48550/arXiv.1602.05629.

[5] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, L. Zhang, "Deep learning with differential privacy", *Proceedings of the 23rd ACM Conference on Computer and Communications Security (CCS)*, Vienna, Austria, 24-28 October 2016, 308–318. https://doi.org/10.1145/2976749.2978318.

[6] C. Dwork, A. Roth, "The algorithmic foundations of differential privacy", *Foundations and Trends in Theoretical Computer Science*, 9(3–4), 211–487, 2014. https://doi.org/10.1561/0400000042.

[7] N. R. Shanbhog, K. S. Totad, A. R. Hanchinal, A. P. Bidargaddi, "Fraud detection in financial transactions using deep learning: A comparative study", *Proceedings of the 5th International Conference on Emerging Technologies (INCET)*, Belgaum, India, 24-26 May 2024. https://doi.org/10.1109/INCET61516.2024.10593486.

[8] F. Z. El Hlouli, J. Riffi, M. A. Mahraz, A. El Yahyaouy, H. Tairi, "Credit card fraud detection based on multilayer perceptron and extreme learning machine architectures", *Proceedings of the International Conference on Intelligent Systems and Computer Vision (ISCV)*, Fez, Morocco, 09-11 June 2020, 1–5. https://doi.org/10.1109/ISCV49265.2020.9204185.

[9] S. K. Hashemi, S. L. Mirtaheri, S. Greco, "Fraud detection in banking data by machine learning techniques", *IEEE Access*, 11, 3034–3043, 2023. https://doi.org/10.1109/ACCESS.2022.3232287.

[10] K. Randhawa, C. K. Loo, M. Seera, C. P. Lim, A. K. Nandi, "Credit card fraud detection using AdaBoost and majority voting", *IEEE Access*, 6, 14277–14284, 2018. https://doi.org/10.1109/ACCESS.2018.2806420.

[11] D. Varmedja, M. Karanovic, S. Sladojevic, M. Arsenovic, A. Anderla, "Credit card fraud detection — Machine learning methods", *Proceedings of the 18th International Symposium INFOTEH-JAHORINA*, East Sarajevo, Bosnia and Herzegovina, 20-22 March 2019, 1–5. https://doi.org/10.1109/INFOTEH.2019.8717766.

[12] Y. Tang, Y. Liang, "Credit card fraud detection based on federated graph learning", *Expert Systems with Applications*, 256, 124979, 2024. https://doi.org/10.1016/j.eswa.2024.124979.

[13] T. K. Dang, T. Ha, "A comprehensive fraud detection for credit card transactions in federated averaging", *SN Computer Science*, 5(5), 578, 2024. https://doi.org/10.1007/s42979-024-02898-y.

[14] N. F. Aurna, M. D. Hossain, Y. Taenaka, Y. Kadobayashi, "Federated learning-based credit card fraud detection: Performance analysis with sampling methods and deep learning algorithms", *Proceedings*

of the IEEE Cyber Security and Resilience (CSR), Venice, Italy, 31 July-02 August 2023, 180–186. https://doi.org/10.1109/CSR57506.2023.10224978.

[15] T. Baabdullah, A. Alzahrani, D. B. Rawat, C. Liu, "Efficiency of federated learning and blockchain in preserving privacy and enhancing the performance of credit card fraud detection systems", *Future Internet*, 16(6), 196, 2024. https://doi.org/10.3390/fi16060196.

[16] A. A. Süzen, B. Duman, "Blockchain-based secure credit card storage system for e-commerce", *Sakarya University Journal of Computer and Information Sciences*, 4(2), 204–215, 2021. https://doi.org/10.35377/saucis.04.02.895764.

[17] S. Alaklabi, "PrivChain-AI leveraging blockchain and federated learning for private financial reporting and access control", *Scientific Reports*, 16(1), 2786, 2026. https://doi.org/10.1038/s41598-025-32606-6.

[18] K. Bian, H. Zheng, "FedAvg-DWA: A novel algorithm for enhanced fraud detection in federated learning environment", *Proceedings of the 4th International Conference on Big Data, Artificial Intelligence and Internet of Things Engineering (ICBAIE)*, Hangzhou, China, 25-27 August 2023, 13–17. https://doi.org/10.1109/ICBAIE59714.2023.10281317.

[19] S. Reddi, Z. Charles, M. Zaheer, Z. Garrett, K. Rush, J. Konečný, S. Kumar, H. B. McMahan, "Adaptive federated optimization", *arXiv*, 2021, https://arxiv.org/abs/2003.00295.

[20] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, V. Smith, "Federated optimization in heterogeneous networks", *arXiv*, 2020. https://doi.org/10.48550/arXiv.1812.06127.

[21] M. Nergiz, "Federe Öğrenmede Birleştirme Algoritmalarının Model Performansına Etkisi", *Dicle Üniversitesi Mühendislik Fakültesi Mühendislik Dergisi*, 14(1), 65–73, 2023. https://doi.org/10.24012/dumf.1241947.

[22] R. A. Mohammed, K. W. Wong, M. F. Shiratuddin, X. Wang, "Scalable machine learning techniques for highly imbalanced credit card fraud detection: A comparative study", *Proceedings of the 15th Pacific Rim International Conference on Artificial Intelligence (PRICAI) / 15th Pacific Rim Knowledge Acquisition Workshop (PKAW)*, Nanjing, China, 28-31 August 2018, 237–246. https://doi.org/10.1007/978-3-319-97310-4_27.

[23] M. Alamri, M. Ykhlef, "Survey of credit card anomaly and fraud detection using sampling techniques", *Electronics*, 11(23), 4003, 2022. https://doi.org/10.3390/electronics11234003.

[24] N. V. Chawla, K. W. Bowyer, L. O. Hall, W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique", *Journal of Artificial Intelligence Research*, 16, 321–357, 2002. https://doi.org/10.1613/jair.953.

[25] G. Lemaître, F. Nogueira, C. K. Aridas, "Imbalanced-learn: A Python toolbox to tackle the curse of imbalanced datasets in machine learning", *Journal of Machine Learning Research*, 18, 17, 2017.

[26] M. S. Farooq, S. F. Munir, M. F. Manzoor, M. Shaheen, "AI-driven adaptive federated learning with privacy preservation and imbalance adjustment for financial credit card fraud detection", *Applied Computational Intelligence and Soft Computing*, 2025(1), 7116768, 2025. https://doi.org/10.1155/acis/7116768.

[27] I. Mironov, "Rényi differential privacy", *Proceedings of the 30th IEEE Computer Security Foundations Symposium (CSF)*, Santa Barbara, CA, USA, 21-25 August 2017, 263–275. https://doi.org/10.1109/CSF.2017.11.

[28] A. Yousefpour, et al., "Opacus: User-friendly differential privacy library in PyTorch", *arXiv*, 2021, https://arxiv.org/abs/2109.12298.

[29] H. B. McMahan, D. Ramage, K. Talwar, L. Zhang, "Learning differentially private recurrent language models", *arXiv*, 2018. https://doi.org/10.48550/arXiv.1710.06963.

[30] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, K. Seth, "Practical secure aggregation for privacy-preserving machine learning", *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*, Dallas, TX, USA, 30 October - 03 November 2017, 1175–1191. https://doi.org/10.1145/3133956.3133982.

[31] Machine Learning Group – ULB, "Credit card fraud detection", *Kaggle*, 2013, https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud, (15.04.2026).

[32] IEEE Computational Intelligence Society and Vesta Corporation, "IEEE-CIS Fraud Detection", *Kaggle*, 2019, https://www.kaggle.com/competitions/ieee-fraud-detection, (15.04.2026).

[33] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, "Generative adversarial networks", *Communications of the ACM*, 63(11), 139–144, 2020. https://doi.org/10.1145/3422622.

[34] A. Sharma, P. K. Singh, R. Chandra, "SMOTified-GAN for class imbalanced pattern classification problems", *IEEE Access*, 10, 30655–30665, 2022. https://doi.org/10.1109/ACCESS.2022.3158977.

[35] A. Paszke, et al., "PyTorch: An imperative style, high-performance deep learning library", *Proceedings of the 33rd Conference on Neural Information Processing Systems (NeurIPS)*, Vancouver, Canada, 08-14 December 2019.

[36] D. J. Beutel, et al., "Flower: A friendly federated learning research framework", *arXiv*, 2020, https://arxiv.org/abs/2007.14390.

[37] T.-M. H. Hsu, H. Qi, M. Brown, "Measuring the effects of non-identical data distribution for federated visual classification", *arXiv*, 2019, https://arxiv.org/abs/1909.06335.

APPENDIX-A

Appendix Tables A1 and A2 report every pairwise Wilcoxon signed-rank test on F_1@best-t (n = 10 paired seeds per test), with Bonferroni-, Holm-, and Benjamini–Hochberg-adjusted p-values computed within the stated family of each experimental phase. For the privacy sweep (Table A1, lower block), the pooled 40-test family is primary and the per-noise-level Holm adjustment (m = 10 per σ) is shown in the final column.

Table A1. Pairwise Wilcoxon tests: Phase 2 (ULB α = 0.5, no DP; family m = 10) and Phase 3 (ULB α = 0.5, DP; pooled family m = 40).

σ	Pair	Mean diff	d	p (raw)	Bonf.	Holm	BH	Holm (per σ)
-	DWA vs FedAdam	+0.0112	+0.63	0.0371	0.371	0.371	0.244	—
-	Cosine vs DWA	-0.0137	-0.64	0.0488	0.488	0.439	0.244	—
-	FedAvg vs DWA	-0.0090	-0.63	0.0977	0.977	0.781	0.290	—
-	FedProx vs FedAdam	+0.0080	+0.51	0.1309	1.000	0.916	0.290	—
-	FedProx vs Cosine	+0.0105	+0.50	0.1641	1.000	0.984	0.290	—
-	FedAvg vs Cosine	+0.0046	+0.37	0.1934	1.000	0.984	0.290	—
-	FedProx vs DWA	-0.0032	-0.44	0.2031	1.000	0.984	0.290	—
-	FedAvg vs FedProx	-0.0059	-0.41	0.2324	1.000	0.984	0.291	—
-	FedAvg vs FedAdam	+0.0022	+0.21	0.7344	1.000	1.000	0.770	—
-	Cosine vs FedAdam	-0.0025	-0.18	0.7695	1.000	1.000	0.770	—
0.5	DWA vs FedAdam	+0.0139	+0.47	0.3125	1.000	1.000	0.614	1.000
0.5	FedProx vs FedAdam	+0.0133	+0.40	0.3223	1.000	1.000	0.614	1.000
0.5	FedAvg vs FedAdam	+0.0131	+0.39	0.3750	1.000	1.000	0.625	1.000
0.5	Cosine vs DWA	-0.0042	-0.22	0.6250	1.000	1.000	0.833	1.000
0.5	FedAvg vs Cosine	+0.0033	+0.15	0.8125	1.000	1.000	0.964	1.000
0.5	Cosine vs FedAdam	+0.0098	+0.23	0.8203	1.000	1.000	0.964	1.000
0.5	FedProx vs Cosine	+0.0035	+0.16	0.8438	1.000	1.000	0.964	1.000
0.5	FedAvg vs DWA	-0.0008	-0.08	0.9766	1.000	1.000	1.000	1.000
0.5	FedProx vs DWA	-0.0006	-0.07	0.9805	1.000	1.000	1.000	1.000
0.5	FedAvg vs FedProx	-0.0002	-0.14	1.0000	1.000	1.000	1.000	1.000

Table A1. Continue.

σ	Pair	Mean diff	d	p (raw)	Bonf.	Holm	BH	Holm (per σ)
1	DWA vs FedAdam	+0.0240	+0.69	0.0371	1.000	1.000	0.279	0.371
1	Cosine vs FedAdam	+0.0201	+0.69	0.0391	1.000	1.000	0.279	0.371
1	FedProx vs FedAdam	+0.0207	+0.52	0.1602	1.000	1.000	0.377	1.000
1	FedAvg vs DWA	-0.0072	-0.41	0.3594	1.000	1.000	0.625	1.000
1	FedAvg vs FedAdam	+0.0168	+0.38	0.4258	1.000	1.000	0.655	1.000
1	Cosine vs DWA	-0.0040	-0.35	0.4258	1.000	1.000	0.655	1.000
1	FedAvg vs FedProx	-0.0040	-0.35	0.5000	1.000	1.000	0.690	1.000
1	FedProx vs Cosine	+0.0007	+0.04	0.7344	1.000	1.000	0.928	1.000
1	FedProx vs DWA	-0.0033	-0.21	0.7422	1.000	1.000	0.928	1.000
1	FedAvg vs Cosine	-0.0033	-0.17	0.9219	1.000	1.000	1.000	1.000
1.5	DWA vs FedAdam	+0.0544	+0.68	0.0039	0.156	0.156	0.156	0.039
1.5	FedAvg vs DWA	-0.0069	-0.73	0.0234	0.938	0.914	0.279	0.211
1.5	FedProx vs DWA	-0.0080	-0.72	0.0234	0.938	0.914	0.279	0.211
1.5	FedAvg vs FedAdam	+0.0475	+0.60	0.0488	1.000	1.000	0.279	0.342
1.5	FedProx vs FedAdam	+0.0464	+0.60	0.0488	1.000	1.000	0.279	0.342
1.5	Cosine vs DWA	-0.0121	-0.67	0.1094	1.000	1.000	0.327	0.547
1.5	Cosine vs FedAdam	+0.0423	+0.59	0.1309	1.000	1.000	0.327	0.547
1.5	FedAvg vs Cosine	+0.0052	+0.41	0.3125	1.000	1.000	0.614	0.938
1.5	FedProx vs Cosine	+0.0041	+0.36	0.4609	1.000	1.000	0.683	0.938
1.5	FedAvg vs FedProx	+0.0011	+0.44	0.5000	1.000	1.000	0.690	0.938
2	FedAvg vs FedAdam	+0.0610	+0.61	0.0645	1.000	1.000	0.280	0.645
2	FedProx vs FedAdam	+0.0608	+0.61	0.0645	1.000	1.000	0.280	0.645
2	Cosine vs DWA	-0.0087	-0.67	0.0840	1.000	1.000	0.280	0.672
2	Cosine vs FedAdam	+0.0551	+0.55	0.0840	1.000	1.000	0.280	0.672
2	DWA vs FedAdam	+0.0637	+0.59	0.0840	1.000	1.000	0.280	0.672
2	FedAvg vs Cosine	+0.0059	+0.58	0.1309	1.000	1.000	0.327	0.672
2	FedProx vs Cosine	+0.0057	+0.57	0.1309	1.000	1.000	0.327	0.672
2	FedProx vs DWA	-0.0030	-0.25	0.2969	1.000	1.000	0.614	0.891
2	FedAvg vs DWA	-0.0028	-0.23	0.3750	1.000	1.000	0.625	0.891
2	FedAvg vs FedProx	+0.0002	+0.32	1.0000	1.000	1.000	1.000	1.000

Table A2. Pairwise Wilcoxon tests on IEEE-CIS: IID no-DP, α = 0.5 no-DP, and α = 0.5 DP σ = 1.0 (family m = 10 each, in that order)

σ	Pair	Mean diff	d	p (raw)	Bonf.	Holm	BH
-	FedAvg vs FedProx	+0.0099	+1.88	0.0020	0.020	0.020	0.003
-	FedAvg vs FedAdam	+0.0635	+18.03	0.0020	0.020	0.020	0.003
-	FedProx vs FedAdam	+0.0536	+10.12	0.0020	0.020	0.020	0.003
-	FedProx vs DWA	-0.0110	-2.39	0.0020	0.020	0.020	0.003
-	DWA vs FedAdam	+0.0646	+11.97	0.0020	0.020	0.020	0.003
-	Cosine vs FedAdam	+0.0643	+17.13	0.0020	0.020	0.020	0.003
-	FedProx vs Cosine	-0.0107	-1.89	0.0039	0.039	0.020	0.006
-	FedAvg vs DWA	-0.0011	-0.26	0.5566	1.000	1.000	0.696
-	FedAvg vs Cosine	-0.0008	-0.24	0.7695	1.000	1.000	0.855
-	Cosine vs DWA	-0.0003	-0.08	0.9219	1.000	1.000	0.922
-	FedProx vs Cosine	+0.0359	+0.88	0.0371	0.371	0.371	0.210
-	FedAvg vs DWA	-0.0158	-0.63	0.0645	0.645	0.580	0.210
-	Cosine vs DWA	-0.0286	-0.91	0.0840	0.840	0.672	0.210
-	FedAvg vs FedProx	-0.0230	-0.62	0.0840	0.840	0.672	0.210
-	FedAvg vs Cosine	+0.0128	+0.52	0.1055	1.000	0.672	0.211
-	FedProx vs FedAdam	+0.0221	+0.46	0.1934	1.000	0.967	0.322
-	FedProx vs DWA	+0.0073	+0.43	0.2324	1.000	0.967	0.332
-	DWA vs FedAdam	+0.0149	+0.35	0.3223	1.000	0.967	0.403
-	Cosine vs FedAdam	-0.0138	-0.23	0.4922	1.000	0.984	0.547
-	FedAvg vs FedAdam	-0.0009	-0.02	1.0000	1.000	1.000	1.000
1	DWA vs FedAdam	+0.0478	+2.37	0.0020	0.020	0.020	0.020
1	FedAvg vs FedAdam	+0.0343	+1.22	0.0137	0.137	0.123	0.046
1	FedProx vs FedAdam	+0.0345	+1.17	0.0137	0.137	0.123	0.046
1	FedProx vs Cosine	+0.0264	+0.73	0.0273	0.273	0.191	0.055
1	FedAvg vs Cosine	+0.0262	+0.72	0.0273	0.273	0.191	0.055
1	FedProx vs DWA	-0.0132	-0.79	0.0488	0.488	0.244	0.061
1	Cosine vs DWA	-0.0396	-0.81	0.0488	0.488	0.244	0.061
1	FedAvg vs DWA	-0.0134	-0.85	0.0488	0.488	0.244	0.061
1	FedAvg vs FedProx	-0.0002	-0.12	0.3750	1.000	0.750	0.417
1	Cosine vs FedAdam	+0.0081	+0.14	0.5566	1.000	0.750	0.557